

Mourad Elloumi · Michael Granitzer
Abdelkader Hameurlain
Christin Seifert · Benno Stein
A Min Tjoa · Roland Wagner (Eds.)

Communications in Computer and Information Science

903

Database and Expert Systems Applications

DEXA 2018 International Workshops
BDMICS, BLOKDD, and TIR
Regensburg, Germany, September 3–6, 2018
Proceedings

 Springer

DEXA 2018

Communications in Computer and Information Science

903

Commenced Publication in 2007

Founding and Former Series Editors:

Phoebe Chen, Alfredo Cuzzocrea, Xiaoyong Du, Orhun Kara, Ting Liu,
Dominik Ślęzak, and Xiaokang Yang

Editorial Board

Simone Diniz Junqueira Barbosa

*Pontifical Catholic University of Rio de Janeiro (PUC-Rio),
Rio de Janeiro, Brazil*

Joaquim Filipe

Polytechnic Institute of Setúbal, Setúbal, Portugal

Igor Kotenko

*St. Petersburg Institute for Informatics and Automation of the Russian
Academy of Sciences, St. Petersburg, Russia*

Krishna M. Sivalingam

Indian Institute of Technology Madras, Chennai, India

Takashi Washio

Osaka University, Osaka, Japan

Junsong Yuan

University at Buffalo, The State University of New York, Buffalo, USA

Lizhu Zhou

Tsinghua University, Beijing, China

More information about this series at <http://www.springer.com/series/7899>

Mourad Elloumi · Michael Granitzer
Abdelkader Hameurlain · Christin Seifert
Benno Stein · A Min Tjoa
Roland Wagner (Eds.)

Database and Expert Systems Applications

DEXA 2018 International Workshops
BDMICS, BIOKDD, and TIR
Regensburg, Germany, September 3–6, 2018
Proceedings

Editors

Mourad Elloumi
University of Tunis
Tunis
Tunisia

Michael Granitzer
MiCS, Media Computer Science
University of Passau
Passau, Bayern
Germany

Abdelkader Hameurlain
IRIT
Paul Sabatier University
Toulouse
France

Christin Seifert
University of Twente
Enschede, Overijssel
The Netherlands

Benno Stein
Fak. Medien
Bauhaus Universität Weimar
Weimar, Thüringen
Germany

A Min Tjoa
Inst. für Softwaretechnik
Vienna University of Technology
Vienna
Austria

Roland Wagner
FAW
Johannes Kepler University of Linz
Linz
Austria

ISSN 1865-0929 ISSN 1865-0937 (electronic)
Communications in Computer and Information Science
ISBN 978-3-319-99132-0 ISBN 978-3-319-99133-7 (eBook)
<https://doi.org/10.1007/978-3-319-99133-7>

Library of Congress Control Number: 2018950775

© Springer Nature Switzerland AG 2018

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, express or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Switzerland AG
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

The Database and Expert Systems Applications (DEXA) workshops are a platform for the exchange of ideas, experiences, and opinions among theoreticians and practitioners – those who are defining requirements for future systems in the areas of database and artificial technologies.

The DEXA workshop papers include papers that are primarily concerned with very specialized topics on applications of database and expert systems technology. We want to thank all workshop organizers for their excellent work.

This was the first time that DEXA workshop papers were published in the *Communications in Computer and Information Science* (CCIS) by Springer.

DEXA 2018 was the 29th annual scientific platform on Database and Expert Systems Applications after Vienna, Berlin, Valencia, Prague, Athens, London, Zurich, Toulouse, Vienna, Florence, Greenwich, Munich, Aix en Provence, Prague, Zaragoza, Copenhagen, Krakow, Regensburg, Turin, Linz, Bilbao, Toulouse, Vienna, Prague, Munich, Valencia, Porto, and Lyon. This year DEXA took place at the University of Regensburg, Germany. Special thanks to Günther Pernul and his local organizing team from the University of Regensburg for hosting DEXA 2018. We would like to express our thanks to all institutions actively supporting this event, namely:

- University of Regensburg
- City of Regensburg
- DEXA Association
- Institute for Application Oriented Knowledge Processing, University of Linz (FAW)
- FAW GmbH

The workshops took place in parallel to the DEXA conference and the program included the following three workshops:

- BDMICS 2018: Third International Workshop on Big Data Management in Cloud Systems
- BIODDD 2018: 9th International Workshop on Biological Knowledge Discovery from Data
- TIR 2018: 15th International Workshop on Technologies for Information Retrieval

June 2018

A Min Tjoa
Roland Wagner

Organization

General Chairs

Abdelkader Hameurlain	IRIT, Paul Sabatier University Toulouse, France
Günther Pernul	University of Regensburg, Germany
Roland R. Wagner	Johannes Kepler University Linz, Austria

Conference Program Chairs

Hui Ma	Victoria University of Wellington, New Zealand
Sven Hartmann	Clausthal University of Technology, Germany

Workshop Chairs

A Min Tjoa	Technical University of Vienna, Austria
Roland R. Wagner	FAW, University of Linz, Austria

Contents

Big Data Management in Cloud Systems (BDMICS)

Parallel Data Management Systems, Consistency and Privacy

A Survey on Parallel Database Systems from a Storage Perspective: Rows Versus Columns.	5
<i>Carlos Ordonez and Ladjel Bellatreche</i>	
ThespisDIIP: Distributed Integrity Invariant Preservation	21
<i>Carl Camilleri, Joseph G. Vella, and Vitezslav Nezval</i>	
Privacy Issues for Cloud Systems	38
<i>Christopher Horn and Marina Tropmann-Frick</i>	

Cloud Computing and Graph Queries

Script Based Migration Toolkit for Cloud Computing Architecture in Building Scalable Investment Platforms	46
<i>Rao Casturi and Rajshekhar Sunderraman</i>	
Space-Adaptive and Workload-Aware Replication and Partitioning for Distributed RDF Triple Stores	65
<i>Ahmed Al-Ghezi and Lena Wiese</i>	
Performance Comparison of Three Spark-Based Implementations of Parallel Entity Resolution.	76
<i>Xiao Chen, Kirity Rapuru, Gabriel Campero Durand, Eike Schallehn, and Gunter Saake</i>	
Big Data Analytics: Exploring Graphs with Optimized SQL Queries	88
<i>Sikder Tahsin Al-Amin, Carlos Ordonez, and Ladjel Bellatreche</i>	

Biological Knowledge Discovery from Big Data (BIOKDD)

New Modeling Ideas for the Exact Solution of the Closest String Problem. . .	105
<i>Marcello Dalpasso and Giuseppe Lancia</i>	
Ensemble Clustering Based Dimensional Reduction.	115
<i>Loai Abdallah and Malik Yousef</i>	

Detecting Low Back Pain from Clinical Narratives Using Machine Learning Approaches. 126
Michael Judd, Farhana Zulkernine, Brent Wolfrom, David Barber, and Akshay Rajaram

Classifying Big DNA Methylation Data: A Gene-Oriented Approach. 138
Emanuel Weitschek, Fabio Cumbo, Eleonora Cappelli, Giovanni Felici, and Paola Bertolazzi

Classifying Leukemia and Gout Patients with Neural Networks 150
Guryash Bahra and Lena Wiese

Incremental Wrapper Based Random Forest Gene Subset Selection for Tumor Discernment 161
Alia Fatima, Usman Qamar, Saad Rehman, and Aiman Khan Nazir

Protein Identification as a Suitable Application for Fast Data Architecture . . . 168
Roman Zoun, Gabriel Campero Durand, Kay Schallert, Apoorva Patrikar, David Broneske, Wolfram Fenske, Robert Heyer, Dirk Benndorf, and Gunter Saake

Mining Geometrical Motifs Co-occurrences in the CMS Dataset 179
Mirto Musci and Marco Ferretti

Suitable Overlapping Set Visualization Techniques and Their Application to Visualize Biclustering Results on Gene Expression Data 191
Haithem Aouabed, Rodrigo Santamaria, and Mourad Elloumi

Technologies for Information Retrieval (TIR)

Web and Domain Corpora

Can We Quantify Domainhood? Exploring Measures to Assess Domain-Specificity in Web Corpora 207
Marina Santini, Wiktor Strandqvist, Mikael Nyström, Marjan Alirezai, and Arne Jönsson

A Case Study of Closed-Domain Response Suggestion with Limited Training Data 218
Lukas Galke, Gunnar Gerstenkorn, and Ansgar Scherp

What to Read Next? Challenges and Preliminary Results in Selecting Representative Documents 230
Tilman Beck, Falk Böschen, and Ansgar Scherp

NLP Applications

Text-Based Annotation of Scientific Images Using Wikimedia Categories . . . 243
Frieda Josi, Christian Wartena, and Jean Charbonnier

Detecting Link and Landing Page Misalignment in Marketing Emails 254
Nedim Lipka, Tak Yeon Lee, and Eunyee Koh

Toward Validation of Textual Information Retrieval Techniques
for Software Weaknesses 265
Jukka Ruohonen and Ville Leppänen

Social Media and Personalization

Investigating the Effect of Attributes on User Trust in Social Media 278
Jamal Al Qundus and Adrian Paschke

Analysing Author Self-citations in Computer Science Publications 289
Tobias Milz and Christin Seifert

A Semantic-Based Personalized Information Retrieval Approach Using
a Geo-Social User Profile. 301
Tahar Rafa and Samir Kechid

Author Index 315

Big Data Management in Cloud Systems (BDMICS)

3rd International Workshop on Big Data Management in Cloud Systems, BDMICS 2018

Preface

<http://www.dexa.org> or <http://www.irit.fr/BDMICS>

The main objective of this third annual international workshop on Big Data Management in cloud systems is to present and exchange the latest results in research and Big Data applications, and to shape future directions. In this perspective, the reviewing process led to the acceptance of 7 full revised papers for presentation at the workshop and inclusion in this CCIS volume. The selected papers focus mainly on parallel data management systems, cloud systems, graph queries, consistency, and privacy. The workshop would not have been possible without the support of the Program Committee members and members of the DEXA Conference Organizing Committee, as well as the authors. In particular, we would like to thank Gabriela Wagner and Roland Wagner (FAW, University of Linz) for their help in the realization of this workshop.

June 2018

Abdelkader Hameurlain

Organization

Chair

Abdelkader Hameurlain IRIT and Paul Sabatier University, France

Program Committee

Reza Akbarinia	Inria and LIRMM, France
Djamal Benslimane	LIRIS and University of Lyon, France
Elizabeth Chang	University of New South Wales, Australia
Shahram Ghandeharizadeh	University of Southern California, USA
Tasos Gounaris	Aristotle University of Thessaloniki, Greece
Omar Khadeer Hussain	University of New South Wales, Australia
Maria Indrawan-Santiago	Monash University, Australia
Sergio Ilarri	University of Zaragoza, Spain
Harald Kosch	University of Passau, Germany
Kjetil Norvag	Norwegian University, Norway
Philippe Lamarre	LIRIS and INSA of Lyon, France
Riad Mokadem	IRIT and Paul Sabatier University, France
Franck Morvan	IRIT and Paul Sabatier University, France
Carlos Ordonez	University of Houston, USA
Claudia Roncancio	LIG and Grenoble University, France
Viera Rozinajova	Slovak University, Slovakia
Soror Sahri	Descartes Paris University, France
Hala Skaf-Molli	LINA and Nantes University, France
Joseph G. Vella	University of Malta, Malta
Shaoyi Yin	IRIT and Paul Sabatier University, France



A Survey on Parallel Database Systems from a Storage Perspective: Rows Versus Columns

Carlos Ordonez¹(✉) and Ladjel Bellatreche²

¹ University of Houston, Houston, USA
carlos@central.uh.edu

² LIAS/ISAE-ENSMA, Poitiers, France

Abstract. Big data requirements have revolutionized database technology, bringing many innovative and revamped DBMSs to process transactional (OLTP) or demanding query workloads (cubes, exploration, pre-processing). Parallel and main memory processing have become important features to exploit new hardware and cope with data volume. With such landscape in mind, we present a survey comparing modern row and columnar DBMSs, contrasting their ability to write data (storage mechanisms, transaction processing, batch loading, enforcing ACID) and their ability to read data (query processing, physical operators, sequential vs parallel). We provide a unifying view of alternative storage mechanisms, database algorithms and query optimizations used across diverse DBMSs. We contrast the architecture and processing of a parallel DBMS with an HPC system. We cover the full spectrum of subsystems going from storage to query processing. We consider parallel processing and the impact of much larger RAM, which brings back main-memory databases. We then discuss important parallel aspects including speedup, sequential bottlenecks, data redistribution, high speed networks, main memory processing with larger RAM and fault-tolerance at query processing time. We outline an agenda for future research.

1 Introduction

Parallel processing is central in big data due to large data volume and the need to process data faster. Parallel DBMSs [13, 15] and the Hadoop eco-system [30] are currently two competing technologies to analyze big data, both based on automatic data-based parallelism on a shared-nothing architecture. On the other hand, larger memory capacity has triggered rearchitecting systems to push more processing to main memory. Nowadays the major DBMS storage technologies are rows and columns. Rows are the most common storage mechanism, researched for decades. They provide great performance for common OLTP queries, but can be slow to process complex queries combining joins and aggregations. Yet during the

C. Ordonez—Work partially conducted while the first author was visiting MIT.

past decade columnar database systems (i.e. DBMSs using column-based storage) [1, 18, 31] have become a major alternative to compute complex SQL queries on large databases, providing at least an order of magnitude in performance improvement compared to row-based DBMSs [31, 33], and two orders of magnitude compared to the Hadoop ecosystem (MapReduce [9], Spark [36]). Unfortunately, column storage requires rewriting many subsystems from the ground up. With that motivation in mind, we present a survey on parallel DBMSs, covering the state of the art and issues for future research. We consider the implications of row and columnar storage, highlighting which processing tasks are easier or more difficult on either storage mechanism. Since it is a major benchmarking effort to compare all systems on loading speed, ACID compliance, query expressiveness and query processing speed we do not include any experiments. Instead we aim to identify main system architecture characteristics, storage mechanisms, and time complexity of algorithms evaluating physical operators.

We provide a unifying view and classification of storage mechanisms, query processing algorithms, novel optimizations and parallel processing across different systems. We cover the full spectrum of subsystems going from storage to analytic processing, contrasting columnar and row DBMSs. Based on the state of the art, we outline a vision of research issues. Due to lack of space we omit many references, especially commercial systems (e.g. white papers, web sites).

2 Preliminaries

Notation: To characterize time and space complexity, we use n to denote table size (number of records), p as a variable for the number of columns (of diverse data types) and P for the number of nodes/processors in a parallel system.

DBMS: We focus on parallel DBMSs, which we classify as row or columnar. A DBMS has the following distinguishing features compared to other large software systems: storage is available only for structured data: relational tables (most common), but also disk-based arrays (new trend); a table schema must be defined before storing data (inserting new records or loading them in batch), querying (in general with SQL, but also with alternative languages like Datalog [2], or XML [13]) and maintaining ACID properties (atomic, consistent, isolated, durable) to maintain data in a complete & consistent state [13].

Landscape: Row DBMSs represent a mature technology dating back several decades [13], pioneered by IBM System R, then followed by Oracle, SQL Server and Teradata. Columnar DBMSs came one decade ago to tackle complex SQL queries. Influential columnar DBMSs include Sybase [24] (a pioneer in columnar storage), MonetDB [18, 25] (industrialized as VectorWise) and C-store [1, 31] (rewritten and commercialized as Vertica [22, 34]). Other columnar DBMSs worth mentioning are SAP Hana [12] (main memory columnar DBMS, integrated with SAP), Virtuoso. Finally, columnar storage was grafted into older row-based

DBMSs like SQL Server (in the form of indexes), Teradata (as user-defined table or precomputed join index) and Oracle (tables in main memory). There is a third class of DBMS with innovative storage: arrays, enabling unlimited size multidimensional arrays and matrices, fundamental in machine learning (SciDB [32], Rasdaman [5]). Due to big data requirements and open-source trends, DBMSs now feature automated schema requirements and enable fast data transfer with Big Data systems. But at the same time “Big Data” Hadoop systems (working on HDFS) have evolved to support SQL queries and more recently transactions with varying ACID guarantees. That is, they have evolved, getting closer to relational DBMSs. A major difference is that DBMSs are generally monolithic, whereas open-source Hadoop systems tend to be modular allowing different subsystems being assembled together (the so-called Hadoop stack). There exist alternative Big Data database systems and subsystems with weaker OLTP features (weaker no ACID properties), noSQL (alternative query languages), generally built on top of the Hadoop distributed file system (HDFS). Figure 1 shows a taxonomy of Big Data Analytics systems.

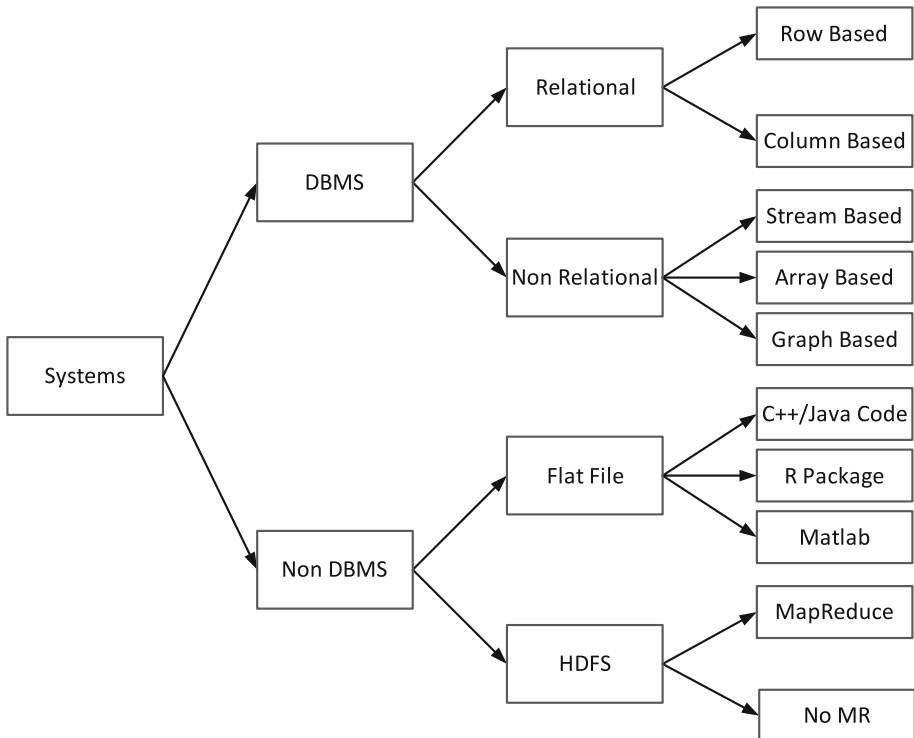


Fig. 1. Classification of parallel DBMSs and Big Data systems based on storage.

3 Parallel System Architectures

There exist diverse parallel system architectures, going from a multicore CPU to multiple CPUs interconnected with a high speed network. The common goal is to exploit as many processing units as possible; such processing units can be multiple threads running on one CPU, several cores on multicore CPUs, many cores in a GPU or a set of servers running on a cluster. Most DBMSs have a shared-nothing parallel architecture with independent secondary storage, whereas HPC systems allow more interconnectivity between processors and secondary storage. Another major difference is the inter-unit communication: HPC systems favor MPI, whereas parallel DBMSs prefer plain UDP sockets. The main reason is intensive I/O on a shared disk (or disk array) or shared memory accessed by multiple threads become a bottleneck. These days there is a divide in HPC between having one machine with many cores and large shared RAM or a cluster with independent RAM and a fast node interconnect. In short, we assume a shared-nothing parallel DBMS architecture with P processing nodes, each with its own memory and disk [10, 30, 33]. A second major difference is main memory access. In general, HPC systems load as much data as possible into main memory across the cluster in one phase, whereas parallel DBMSs use a buffer (a.k.a. cache) with clever page eviction. Given larger RAM there is a renewed trend to avoid I/O and perform most transaction and query processing in main memory.

4 Writing: Storage and Updating Database

We contrast row and columnar DBMSs, highlighting internal system changes. We justify the need to define a schema. For each subsystem we will first present a unifying view of current research, identifying common mechanisms, algorithms, data structures and optimizations, emphasizing tradeoffs among them and pointing out optimizations not used in practice.

Table 1 provides a summary of popular parallel systems. It shows DBMSs supporting SQL, HDFS systems and alternative systems that offer some DBMS functionality. This table is by no means complete as it omits many smaller systems with more specific functionality, or tailored to specific analytic problems.

4.1 Storage Mechanisms

Files: In a row DBMS running on one machine each table is generally stored on one file, partitioned into blocks of rows (unless it is “sharded”). Thus reading a row implies reading all columns. In contrast, with column-based storage queries read only the few columns they need from a wide denormalized table [31]. At the logical level columnar DBMSs exploit relational projections [18, 22, 23] of the form $\pi_{i, A_1, A_2, \dots, A_p}(T)$ on table T , where i is an internal row identifier. In general, such projections do not represent materialized views because they do not involve joins and aggregations [18, 22, 24, 31]. The main optimization is to process queries on such projections instead of T . A well chosen set of projections

Table 1. DBMSs & Big Data systems: storage, query language, parallel architecture.

Name	Storage	Query language	Shared-nothing P nodes
DataDepot	Row	SQL	Y
DB2	Row	SQL:2011	Y
Greenplum	Row	SQL:2008	Y
MemSQL	Row	SQL	Y
MySQL	Row	SQL:1999	Y, sharding
Oracle	Row	SQL:2008	Y
SQL Server	Row	Transact SQL	N, Y in PDW
PostgreSQL	Row	SQL:2008	N
SQLite	Row	SQL	N
Teradata	Row	SQL:1999 and T-SQL	Y
VoltDB	Row	SQL	Y
InfiniDB	Columnar	SQL:92	Y
MonetDB	Columnar	SQL:2008, SciSQL	N
SAP Hana	Columnar	SQL:92	Y
Vertica	Columnar	SQL:1999	Y
Accumulo	HDFS	Thrift	Y
Hbase	HDFS	(J)Ruby's IRB	Y
Impala	HDFS	SQL:92	Y
Shark	HDFS	SQL	
PostGIS	Array/block	SQL:2008	N
Rasdaman	Array/raster	RaSQL	N
SciDB	Array/chunk	AQL	Y

can substitute T [31], although it is preferable to materialize T anyway [22]. At the physical level each projection is further partitioned into columns, where values for each column A_j are stored in a separate file. An essential internal reference mechanism is that the row id i is not stored, but it is an implicit array subscript based on some value ordering [1]. Such approach has two fundamental benefits: (1) it decreases storage space [1] and it reduces I/O cost [18]; (2) it allows using i as an array subscript, bypassing indexing mechanisms [1, 18]. As a consequence, the boundary between the logical and physical storage levels gets blurred, in contrast to row DBMSs. In general, the row identifier i is derived from the column value position inside a block. Column values are generally stored in compressed form; in such case the system determines an offset by multiplying the value position by the value frequency. Storing columns in separate files allows multithreaded parallel reads and it yields high compression ratios [18, 31], but it requires assembling rows (materialization) at the end of query processing. Array DBMSs have many similarities with columnar DBMSs: they store each attribute in separate storage units, they partition arrays by column, they feature

improved locality, larger blocks and compression. The main differences are that multidimensional arrays have uniform content across their dimensions and they need mathematical and physical operators incompatible with relational algebra.

I/O Unit: In row DBMSs a block of rows remains the I/O standard. Compared to row OLTP systems a larger I/O unit is more common in columnar systems. This generally results in a big logical block: segment in a columnar DBMS [22]. The rationale is that a larger block favors long scans and minimizes seeks for query processing. Blocks may have non uniform size depending on value distributions and compression. Grouping column values into blocks requires considering two aspects [18, 22, 31]: improving locality of access and parallel processing. The system improves access locality by maintaining values sorted. Fixed size BLOBS (pure byte strings) enable direct loading and array access [18, 23, 37], but they are not compatible with compression.

Hybrid Storage Mixing Rows and Columns: Can rows and columns coexist in the same storage system? yes, but it is difficult because the physical operator executor requires significant changes. Can we store row and columns on the same block? There is not a clear answer. The net result: performance not good enough compared to a pure column store [1]. Since full support for column storage requires rewriting the storage manager from scratch, legacy row DBMSs have opted for limited column optimizations [23]. Currently, the fastest columnar DBMSs have two separate internal storage managers [12, 22], as we explain later.

4.2 Fast Access: Ordering Rows Versus Indexes

Ordered Value Storage to Avoid Indexing: There are two major alternatives to improve access time to a subset of rows based on a search key: adding indexes to improve search time (row DBMS) or maintaining sorted projections (columnar DBMS). In row DBMSs the most common data structures to index column values are B-trees and hash tables with $O(\log(n))$ and $O(1)$, average time respectively. In a columnar DBMS since query processing requires searching values there are two major alternatives to decrease search time: maintaining values sorted or using an auxiliary data structure to search values. If values are sorted search takes time $O(\log(n))$. But since an index is a projection of a table most approaches favor maintaining values sorted, which can be done in an efficient manner when insertions come in batches. Notice ordering is crucial for time series, spatial data and streams: queries tend to access value ranges.

Compression: Both row DBMSs and columnar DBMSs exploit compression, but they sharply differ on how they process queries. Two reasons motivate compression: the existence of a few frequent column values and maintaining column values physically sorted. We emphasize compression of columns with repeated values is not a new idea, since compression is commonly applied on row DBMSs

and compression is a common mechanism to accelerate data transfer in networking. Moreover, efficiently storing long sequences of repeated symbols is well known in text processing. Columnar storage provide much better compression than row stores [1]. In fact, a subtle aspect that is not well known is that columnar DBMSs actually provide higher compression rates than popular compressing utilities [22] (e.g. gzip, winzip). Thus compression significantly reduces I/O cost and network traffic in a parallel system. Yet the overhead to decompress values at query time must be considered. Since storage is column based, the chosen compression granularity is per column, which creates blocks of compressed data. The two most common compression mechanisms in columnar DBMSs are run-length encoding (RLE) and dictionary encoding (DE), with RLE being the most popular. RLE can only be applied when values are sorted and then counted. In an alternative manner, DE is preferred on row DBMSs and it is useful when there are few frequent column values; this scheme is more effective if values take many bytes (i.e. long strings). Hash tables and arrays are two common mechanisms to program dictionaries with a code of a fixed bit length. There exist other compression mechanisms like delta encoding, with less general applicability. Compression mechanisms that produce variable length codes or that encrypt data are slower at query time. So LZW & Huffman codes or arithmetic compression are a bad idea in a DBMS [1, 31]. In short, lightweight compression is preferable to enable faster decompression [22, 37] or working directly on a compressed data representation [22, 31].

Indexing: In a row DBMS B-trees are preferred for point/range queries and transaction processing; hashing is preferred to distribute rows for parallel processing and accelerate join computation; bitmaps are preferred to accelerate “count” aggregations. In contrast to row DBMSs, a columnar DBMS [22] does not use row-level indexing; sparse indexes (trees on min/max values per compressed block [31]) are eagerly created when the user defines projections [22] or dynamically by the DBMS during query processing (database cracking [18]). Some row DBMSs expose columnar projections as special indexes [23].

4.3 Inserting and Updating Database Records

We consider two major mechanisms to update the database: inserting/updating a few records, most commonly done by transactions and loading a batch of records, generally done in a data warehouse or analytical database. Such mechanisms lead to two prominent concurrent processing scenarios: (1) OLTP: Updating or deleting few records concurrently with multi-threaded processing in shared memory or (2) OLAP/cubes: doing batch loading and processing complex queries at the same time. We discuss each mechanism below.

Transactions: Transactions (small SQL programs updating the database) enable concurrent updates and provide fault tolerance. Row DBMSs remain the best technology to process transactions, where locking (2PL, optimistic [13]) and

multi-version concurrency control (MVCC+timestamping) [13] remain the most common mechanisms to enforce ACID properties (i.e. serializable and recoverable schedules). It is generally agreed locking is slower but provides strongest consistency guarantees, whereas MVCC is faster but creates multiple inconsistent views of the database. Recently row DBMSs have started moving towards lock-free approaches using timestamping [33] exploiting main memory processing with single threaded processing (very fast, significantly simplifying OLTP processing and recovery, defying old assumptions). A main reason behind such acceleration is that they exploit servers with much larger RAM. When performing parallel processing in a cluster timestamping can be used on multiple nodes if the transaction data records can be partitioned and routed for local processing. For small and medium size OLTP databases (relative to large RAM) it is feasible to maintain the database in main memory (e.g. VoltDB) across all servers in the parallel cluster. On the other hand, transaction processing is considered infeasible in a columnar DBMS, because to update a row columns should be decompressed, rows dynamically assembled and then compressed again and written back. Moreover, updating values on multiple rows (especially non-key columns) in a columnar DBMS is significantly slower than row DBMSs due to the need to decompress and compress and potential data re-partitioning [31] (a.k.a. shuffling). As explained below, a common solution is to cache recent updates in batch in a data structure in RAM [12, 22]. In general, columnar DBMSs provide minimal transaction processing features, but provide reasonable (but weaker) ACID guarantees [12, 22]. On the other hand, to enable fast bulk insertions (no updates or deletes) and concurrent querying most systems provide snapshot isolation, internally managed with record versioning. That is, records are inserted into blocks that cannot be accessed by the user until loading is complete.

Loading Data: Loading data from row OLTP databases into a central row database (data warehouse) is fast and in some systems (Teradata, MemSQL) is done in near real time (e.g. active data warehousing). In a columnar DBMS, loading data in batch is generally a bottleneck because row to column transformation is required [18, 22], which in turn requires sorting each column. Thus transforming row by row to columns is slow [22, 31] (akin to matrix transposition). A solution is having two internal storage managers with a batched tuple mover between a write-optimized row store and a write-optimized column store. In general, the write-optimized store works in RAM [12, 22], where refreshing data with δ tuples is done as a background process to avoid query interruption [22]. Sybase IQ [24] explicitly forces a tuple order, by time, to avoid sorts.

5 Reading: Query Processing

We first discuss how sequential processing happens on one machine or one thread. Based on such foundation, we then discuss how sequential processing is generalized and extended to work in parallel.

5.1 Sequential Query Processing

Physical Operators: In a row DBMS, there are scan (read all rows), join (merge-sort, hash), sort and search (indexed or sequential) algorithms. On the other hand, in a columnar DBMS, there are newer versions of scan, join, sort and search algorithms acting at the column level [20]. A scan operator brings blocks directly into RAM. These are the major scan variations: reading small blocks that can fit in the CPU cache memory [18], reading medium blocks with compressed column values [22], reading large blocks with uncompressed column values. Scans on multiple columns are optimal when such columns are aligned on the same order, enabling direct manipulation with arrays. There are two preferred kinds of join algorithms: merge joins and hash joins [18, 22, 31], where merge joins avoid the sort phase in the classical sort-merge join. In general, most DBMSs avoid nested loop joins. In contrast to row DBMSs, merge joins are preferred over hash joins, when input tables have their projections already sorted by the joining columns. Otherwise, DBMSs use hash joins as default, creating a hash table on the (inner) smaller participating table [22]. When both tables are compressed on the same key and merge join is feasible time complexity is $O(k)$, where $k = |\pi(K)|$ for the join key K . Sorting is necessary when no projection satisfies a join or aggregation GROUP BY required order or when the cardinality of the column is large. Sorting (including external sort) generally uses the traditional merge sort algorithm in time $O(n \log(n))$ to create a sorted projection. Sequential searches are avoided. Whenever there exists a projection physically sorted by a key the default search algorithm is binary search, which can be as efficient as $O(\log(k))$ on a compressed column, or $O(\log(n))$ otherwise. Otherwise, time is $O(n)$, but assumed a rare occurrence. In a row DBMS projection π takes time $O(pn)$, regardless of how many columns are projected. In contrast, in a columnar DBMS projection takes time $O(mk)$ for m projected columns and k distinct values, and in most cases $m \ll p$ and $k \ll n$.

Query Executor: To evaluate each operator a row DBMS takes as input rows from one or two tables (for unary and binary operators, perhaps pipelined) and produces rows on one output table. A columnar DBMS is significantly different, having these main steps: determine required columns, read column values by block, search column blocks (exploiting some indexing data structure), transfer blocks to buffer in RAM, process blocks in RAM with column physical operators, and assemble the rows at some point. Processing algorithms across DBMSs vary significantly depending on compression. If blocks are compressed it is preferable to process them in compressed form, delaying decompression until results are returned [1]. Dictionary encoding does not require decompressing in intermediate evaluation steps, whereas run-length encoding helps query evaluation, by directly processing the repeated value and its frequency. Queries require eventually assembling rows from column values. There are two major approaches to assemble rows: early materialization [18] and late materialization [22, 31], leading to significantly different query processing. Late materialization provides best performance when compression is heavily used [1]. The level of integration of the

query processor with the hardware varies significantly going from a small query optimizer/processor kernel, tightly integrated with the operating system [18] to a fully fledged big optimizer [22], comparable to legacy DBMSs.

Query Plan: Most DBMSs provide ANSI SQL compatibility. Most row DBMSs represent the query plan in a similar manner, following IBM System R [13]. In contrast, columnar DBMSs differ significantly on how queries are internally represented and optimized. If a traditional relational algebra approach is used, relational query transformation rules and cost-based query optimization are applied together [22, 31]. On the other hand, other DBMSs define a simpler algebra on narrow projections (e.g. binary association tables [18, 37]). to enable tight integration with the CPU and operating system and a smaller space for potential query plans (operator commutativity is more limited). Query transformation favors pushing projection instead of pushing selection in SPJA queries [1, 22]. In contrast to row DBMSs, when projections share ordering by the same key, join operators are evaluated first since they act as a filter. In a query optimizer with full place space exploration, like [22], query trees tend to be bushier [22] instead of left deep [13]. Thus plan space exploration becomes harder with bushy trees. Some DBMSs use traditional cost models combining CPU, disk I/O and network transfer cost [22], with more upfront optimization, whereas other DBMSs delay such optimization, combining query transformation with dynamic storage re-organization and adaptive indexing [18, 37].

Non-relational Operators and Functions: It is no surprise there is work on adapting DBMSs to analyze non-relational data [18]. The two most outstanding novel data types are streams and arrays. Stream data records arrive row by row [16]: they require a different query processor combining main memory and secondary storage. Also, streams arrive (mostly) in time order which complicates reorganizing storage when accessing and ordering by a different column. In order to keep up with stream speed column stores require a fast incremental converter of new rows. A common form of stream operator has a stream in RAM and a table on disk (i.e. a join between a table and a stream). On the array angle, unfortunately arrays and relational tables are incompatible with each other. Therefore, it is necessary to develop new operators (sometimes extending SQL) and new languages that can manipulate arrays.

5.2 Parallel Query Processing

Data Partitioning: This is statically done at table loading or dynamically during query processing [6, 13]. Sharding [30] creates subsets of tables and can work across new servers. To provide 24/7 availability, it is a requirement to add/remove hardware (including nodes), without disruption. Such dynamic hardware expansion is particularly popular with MapReduce (MR) [9, 21], Spark [36] and specialized systems on top of HDFS [3]), which exploit heartbeats, data replication and daemon processes. Data redistribution (shuffling) happens when

required rows or values are not available on the same node, transferring data to different nodes. There are two major approaches to partition tables across P nodes: hashing or range-based, like row DBMSs [13, 22]. Parallel DBMSs provide three basic operators to send and receive data blocks [13, 22]: (1) broadcast a block to all P nodes (scatter). (2) converge-cast from P nodes to 1 node (gather). (3) node to node 1-1. There are two partitioning levels: inter-node horizontal partitioning and intra-node partitioning (which extends horizontal partitioning). Data redistribution happens after some physical operators and it cannot be guaranteed results will remain on the same node. So load re-balancing is required. Sharding [30] is another important partitioning technique, which horizontally partitions tables into “focused” subsets to avoid full table scans when analyzing data.

Parallel Speedup: We compare merits and limitations of each approach with respect to parallel processing theory [11]. Currently, there are two parallelism dimensions: scale-up and scale-out. Scale-up is represented by multicore CPUs, GPUs using multi-threaded processing on a single node [18], where main memory may be shared or partitioned. On the other hand, scale-out is represented by multi-node parallel processing. Multi-core CPUs and GPUs keep growing the number of cores, but their speed has reached a threshold. GPUs represent an alternative with a much higher number of cores, but with separate main memory and different API. RAM keeps growing, but multiple CPUs compete for it (i.e. contention for a shared resource). Network transfer remains slower than the transfer from CPU to RAM, although fast CPU interconnections are changing the landscape. Therefore, in general network communication is the bottleneck, followed by RAM (i.e. the memory wall [18]). All systems aim to achieve linear speedup, minimizing the sequential bottleneck (Amdahl’s law [11]) and communication overhead. The parallel cluster topology (seen as a graph connecting P nodes) can be: complete graph ($O(P^2)$ connections), tree (balanced binary tree $O(P)$ connections), grid (connecting CPUs and disks). Most DBMSs assume a complete graph to enable point-to-point data transfer, but synchronized transfer. Network communication is commonly done with sockets (medium speed), with MPI (slower, but with higher reliability, compatible with HPC applications) or with high-speed interconnect hardware (e.g. Infiniband). MonetDB [18] (commercially VectorWise) is the best DBMS exploiting hardware: multicore CPUs, increasing cache hits; exploit locality of reference, exploiting faster seek on flash, fast long scans on disk and flash.

Parallel Physical Operators: There are parallel versions of scan, join and sort [18, 22, 31, 37]. Scan works by transferring compressed blocks to the requesting node, decompressing at the end. The fastest join algorithms (highlight differences) include hash joins or merge joins. Nested loop joins are rare, but they are used by replicating a small table at all the P nodes to be joined with a much larger table. The default sorting algorithm is merge-sort. In general, sorting an entire table happens only during query processing because insertion maintains

table columns sorted separately. Big blocks minimize network traffic as long as all values are relevant (i.e. pre-filtered and with a useful value ordering).

Fault Tolerance: Parallel processing at massive scale requires fault tolerance to process long lasting queries and graph/ML analytics, to avoid restarting jobs [22, 30]. Thus, this is similar to MapReduce and Spark. Some DBMSs provide K -safety [22, 31] in which $K + 1$ copies must be maintained to tolerate up to K node failures. To guarantee safe operation, at any time A , the number of active nodes, must be $A > P/2$.

6 Conclusions: A Tentative Agenda for Future Research

Row DBMSs are best for transaction processing, very fast for batch loading and provide good performance for complex queries, whereas columnar DBMSs are bad for transaction processing, reasonably fast for batch loading and fastest for complex queries. Even so there are many opportunities for future research. Big Data Analytics is an emerging area, going beyond database systems. Here we categorize important research problems into core database research, where DBMS technology prevails and where problems are well established and big data analytics systems, where the norm today is cloud computing and a weak enforcement of a database schema. Due to space limitations, we omit discussion on database modeling (ER model, process management), data pre-processing (data cleaning, ETL) and database integration.

6.1 Core Database Systems Research

Storage: It is necessary to investigate the coexistence of two different storage mechanisms: row and column: currently two internal DBMSs seem the most efficient approach [12, 22]. More efficient row to column converters are needed to process deltas on streams and high velocity OLTP transactions. Hardware can be exploited, especially RAM and SSDs. For instance, Hana [12] can efficiently query an entire database in RAM (say hundreds of GBs). So a given a query workload a carefully chosen subset of the database could be processed completely in RAM, like Shark [35]. Compressed blocks can be more efficiently read and written on SSDs. Support for diverse and complex data types is not well understood yet: fixed length storage helps address computation, whereas compression leads to variable length storage. Text data with a large number of keywords across a document collection represents a sparse data set with many opportunities for optimization. BLOBs are used to enable arrays and exploiting cache memory [18], but it is unclear if they can be combined with RLE compression. UDTs mixing simple data types and text need careful storage layout optimization.

Updating Database: Processing transactions is a bad fit for columnar DBMSs, but they generally incorporate an internal row store in RAM to convert rows

to columns. Timestamping has proven more effective for fast multicore CPUs [18, 22], leaving locking for legacy DBMSs. Given the speed of columnar DBMSs and their storage flexibility it may be better to do ELT, instead of ETL to integrate databases. Another potential improvement, enabled by new hardware, is to provide transaction processing and ad-hoc querying on the same DBMS when the database fits in RAM.

Indexing: Since ordering by every projection is infeasible some form of indexing is needed to complement ordering. Sparse B-trees are a solution [31], but it is unclear if they can benefit more general query processing. Hashing is used in main memory processing [22], but not on disk. Adaptive indexing can accelerate query processing as the workload varies, like database cracking.

Query Processing: Row and column stores are competing and influencing each other. So it is necessary to investigate guidelines to add row DBMS features to columnar DBMSs and vice-versa. Bushy plans [22] versus shallow/simpler query trees [18]. We need new algorithms for automated physical design to compete with NoSQL systems. Fast long writes and faster seeks on SSDs can accelerate scans and even hash-based joins. Streams need revisiting joins (band joins, stream+table lookup join), and complex aggregations (especially approximate histograms). We need to revisit well-known analytic query problems with hybrid row and columnar processing: cubes (iceberg queries [13], sparse cube storage [29], horizontal aggregations [28]), recursive queries (transitive closure [4, 26]), skylines [16], and spatio-temporal data [13]. For ever-growing data sets approximation and sampling are needed (sample for plan cost estimation [18], view materialization, stream analysis over a window [37], dynamically reindexing continuously changing data [19]), but sampling requires materializing rows. Development of new database languages is needed: going beyond SQL, XML, Datalog, although adoption is a practical problem.

Parallel Processing: Parallel merge/hash joins and GROUP BY aggregations remain challenging. New data partitioning and alignment algorithms are needed to improve data balancing. It is necessary to study the tradeoffs between traditional partitioning versus sharding [30]. Query workload optimization will become more important: query scheduling, like Hadoop/MR job scheduling [9]. Fault tolerance with massive parallelism for slow queries is needed [22].

6.2 Big Data Management and Machine Learning

Storage and Querying: The main characteristics of Big Data are the three “Vs”: Volume, Velocity and Variety [7, 14, 30], although this is being expanded with two additional Vs: veracity (many data sources, possibly with conflicting information) and value (usefulness). Volume is now represented by two data worlds: data warehouses and search engines. Variety is the hardest challenge

[7]: data management issues are compounded by text (documents, files, web data, logs), matrices (vectors, arrays) spatio-temporal data (location, historical) and graphs (describing general relationships). Velocity is represented by streams: sensors and user log data. Currently, columnar DBMSs are very fast for wide denormalized tables [1, 22], slightly slower than row DBMSs for large narrow tables (i.e. normalized tables) [1], reasonably fast for data coming periodically in batches (data warehousing), OK for high velocity data (exploiting cache and multicore CPUs [37]), fair, but better than row DBMSs, for “variety” data with complex structure (graphs) or little or no structure (text, no schema). Since columnar DBMSs have more flexible storage than row DBMSs, they can be adapted to process text relaxing or automating the schema definition requirement using key-value pairs like Hadoop systems, thereby allowing storage of columns with varied text content. Further research is needed to query large graphs, especially stored by edge [26], analyzing tradeoffs between dense (e.g. quadratic number of edges, highly connected) and sparse graphs (e.g. linear number of edges, trees, disconnected). Columnar DBMSs show promise to store variable length text data with inconsistent information: innovative research is needed to automate schema definition on text data.

Machine Learning: Considering previous research on row DBMSs [8, 17, 27] and popular Hadoop/MapReduce/Spark [8, 36], this is a categorization of problems in descending importance order: (1) Processing alternatives: is it better to build data mining tools working outside the DBMS realm? this approach affords flexibility and overcomes DBMS limitations to develop fast algorithms, but it leads to data management problems and I/O bottlenecks to transfer data [17, 27]. Therefore, it is necessary to determine a complexity/cost boundary of problems that cannot be computed efficiently even inside a columnar DBMS. (2) Scalable computation of statistical models with large data sets (hard) and high dimensionality (harder). The three major solutions are: internal integration with DBMS source code via cursors UDFs, SQL queries. Given their speed, columnar DBMSs look promising to compute models and graphs entirely with SQL queries, followed by UDFs exploiting denormalization and fast joins/aggregations. However, most algorithms require all columns, not a projection. Reducing the number of passes over the data set and data summarization seem orthogonal to storage by row or column: tradeoffs between both storage mechanisms need study. On the other hand, sampling requires row materialization. (3) Pattern discovery: association rules, sequences, Such algorithms need traversing the dimensions lattice, which generally requires pointers, but may be faster with columns than rows. (4) Supporting broader mathematical processing, enabled by arrays: graph mining, matrix operators, linear algebra and key numerical methods like gradient descent. The integration of mathematical packages and libraries (e.g. R, Matlab, BLAS/LAPACK) with the DBMS is difficult and time consuming. Is it hacking? No, we believe there are indeed research issues: There is a programming language impedance mismatch with SQL, efficient data transfer and conversion in RAM are needed, memory and disk management is difficult when running on

the same hardware. (5) Text and web data analytics are a better fit for Hadoop (MapReduce and Spark), but the following tasks need further study in modern DBMSs when documents and databases are inter-related: text preprocessing, ranking, ontologies, document classification and text summarization.

Acknowledgments. The first author thanks the guidance from Michael Stonebraker to understand query processing based on columnar storage, arrays of unlimited size to support mathematical analytics and lock-free transaction processing in main memory.

References

1. Abadi, D.J., Madden, S., Hachem, N.: Column-stores vs. row-stores: how different are they really? In: Proceedings of ACM SIGMOD Conference, pp. 967–980 (2008)
2. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases: The Logical Level, Facsimile edn. Pearson Education POD, London (1994)
3. Abouzied, A., Bajda-Pawlikowski, K., Huang, J., Abadi, D.J., Silberschatz, A.: HadoopDB in action: building real world applications. In: Proceedings of ACM SIGMOD Conference, pp. 1111–1114. ACM (2010)
4. Bancilhon, F., Ramakrishnan, R.: An Amateur’s introduction to recursive query processing strategies. In: Proceedings of ACM SIGMOD Conference, pp. 16–52 (1986)
5. Baumann, P., Dumitru, A.M., Merticariu, V.: The array database that is not a database: file based array query answering in Rasdaman. In: Nascimento, M.A., et al. (eds.) SSTD 2013. LNCS, vol. 8098, pp. 478–483. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40235-7_32
6. Bellatreche, L., Benkrid, S., Ghazal, A., Crolotte, A., Cuzzocrea, A.: Verification of partitioning and allocation techniques on teradata DBMS. In: Xiang, Y., Cuzzocrea, A., Hobbs, M., Zhou, W. (eds.) ICA3PP 2011. LNCS, vol. 7016, pp. 158–169. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24650-0_14
7. Ceri, S., Della Valle, E., Pedreschi, D., Trasarti, R.: Mega-modeling for big data analytics. In: Atzeni, P., Cheung, D., Ram, S. (eds.) ER 2012. LNCS, vol. 7532, pp. 1–15. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34002-4_1
8. Cohen, J., Dolan, B., Dunlap, M., Hellerstein, J., Welton, C.: MAD skills: new analysis practices for big data. In: Proceedings of VLDB Conference, pp. 1481–1492 (2009)
9. Dean, J., Ghemawat, S.: MapReduce: simplified data processing on large clusters. *Commun. ACM* **51**(1), 107–113 (2008)
10. DeWitt, D., Gray, J.: Parallel database systems: the future of high performance database systems. *Commun. ACM* **35**(6), 85–98 (1992)
11. Dongarra, J., Duff, I.S., Sorensen, D.C., van der Vost, H.A.: Numerical Linear Algebra for High-Performance Computers. SIAM (1998)
12. Färber, F., et al.: The SAP HANA database: an architecture overview. *IEEE Data Eng. Bull.* **35**(1), 28–33 (2012)
13. Garcia-Molina, H., Ullman, J.D., Widom, J.: Database Systems: The Complete Book, 2nd edn. Prentice Hall, Upper Saddle River (2008)
14. Ghazal, A., et al.: BigBench: towards an industry standard benchmark for big data analytics. In: Proceedings of ACM SIGMOD Conference, pp. 1197–1208. ACM (2013)

15. Hameurlain, A., Morvan, F.: Parallel relational database systems: why, how and beyond. In: Wagner, R.R., Thoma, H. (eds.) DEXA 1996. LNCS, vol. 1134, pp. 302–312. Springer, Heidelberg (1996). <https://doi.org/10.1007/BFb0034690>
16. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann, San Francisco (2006)
17. Hellerstein, J., et al.: The MADlib analytics library or MAD skills, the SQL. Proc. VLDB **5**(12), 1700–1711 (2012)
18. Idreos, S., Groffen, F., Nes, N., Manegold, S., Mullender, K.S., Kersten, M.L.: MonetDB: two decades of research in column-oriented database architectures. IEEE Data Eng. Bull. **35**(1), 40–45 (2012)
19. Idreos, S., Kersten, M.L., Manegold, S.: Self-organizing tuple reconstruction in column stores. In: Proceedings of ACM SIGMOD Conference, pp. 297–308 (2009)
20. Jacobs, A.: The pathologies of big data. Commun. ACM **52**(8), 36–44 (2009)
21. Jemal, D., Faiz, R., Boukorca, A., Bellatreche, L.: MapReduce-DBMS: an integration model for big data management and optimization. In: Chen, Q., Hameurlain, A., Toumani, F., Wagner, R., Decker, H. (eds.) DEXA 2015. LNCS, vol. 9262, pp. 430–439. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-22852-5_36
22. Lamb, A., et al.: The Vertica analytic database: C-store 7 years later. PVLDB **5**(12), 1790–1801 (2012)
23. Larson, P.A., Hanson, E.N., Price, S.L.: Columnar storage in SQL server 2012. IEEE Data Eng. Bull. **35**(1), 15–20 (2012)
24. MacNicol, R., French, B.: Sybase IQ multiplex - designed for analytics. In: Proceedings of VLDB Conference, pp. 1227–1230 (2004)
25. Manegold, S., Boncz, P.A., Kersten, M.L.: Optimizing main-memory join on modern hardware. IEEE Trans. Knowl. Data Eng. (TKDE) **14**(4), 709–730 (2002)
26. Ordonez, C.: Optimization of linear recursive queries in SQL. IEEE Trans. Knowl. Data Eng. (TKDE) **22**(2), 264–277 (2010)
27. Ordonez, C.: Statistical model computation with UDFs. IEEE Trans. Knowl. Data Eng. (TKDE) **22**(12), 1752–1765 (2010)
28. Ordonez, C., Chen, Z.: Horizontal aggregations in SQL to prepare data sets for data mining analysis. IEEE Trans. Knowl. Data Eng. (TKDE) **24**(4), 678–691 (2012)
29. Sismanis, Y., Deligiannakis, A., Roussopoulos, N., Kotidis, Y.: Dwarf: shrinking the petacube. In: ACM SIGMOD Conference, pp. 464–475 (2002)
30. Stonebraker, M., et al.: MapReduce and parallel DBMSs: friends or foes? Commun. ACM **53**(1), 64–71 (2010)
31. Stonebraker, M., et al.: C-Store: a column-oriented DBMS. In: Proceedings of VLDB Conference, pp. 553–564 (2005)
32. Stonebraker, M., Brown, P., Zhang, D., Becla, J.: SciDB: a database management system for applications with complex analytics. Comput. Sci. Eng. **15**(3), 54–62 (2013)
33. Stonebraker, M., Madden, S., Abadi, D.J., Harizopoulos, S., Hachem, N., Helland, P.: The end of an architectural era: (it’s time for a complete rewrite). In: VLDB, pp. 1150–1160 (2007)
34. Tran, N., Bodagala, S., Dave, J.: Designing query optimizers for big data problems of the future. PVLDB **11**(6), 1168–1169 (2013)
35. Xin, R.S., Rosen, J., Zaharia, M., Franklin, M.J., Shenker, S., Stoica, I.: Shark: SQL and rich analytics at scale. In: Proceedings of ACM SIGMOD Conference, pp. 13–24 (2013)
36. Zaharia, M., Chowdhury, M., Franklin, M.J., Shenker, S., Stoica, I.: Spark: cluster computing with working sets. In: HotCloud USENIX Workshop (2010)
37. Zukowski, M., Boncz, P.: Vectorwise: beyond column stores. IEEE Data Eng. Bull. **35**(1), 21–27 (2012)



ThespiDIIP: Distributed Integrity Invariant Preservation

Carl Camilleri^(✉), Joseph G. Vella, and Vitezslav Nezval

Department of Computer Information Systems, University of Malta, Msida, Malta
carl.camilleri.04@um.edu.mt

Abstract. Thespi is a distributed database middleware that leverages the Actor model to implement causal consistency over an industry-standard DBMS, whilst abstracting complexities for application developers behind a REST open-protocol interface. ThespiDIIP is an extension that treats the concept of integrity invariance preservation for the class of problems where value changes must be satisfied according to a Linear Arithmetic Inequality constraint. An example of this constraint is a system enforcing a constraint that a transaction is only accepted if there are sufficient funds in a bank account. Our evaluation considers correctness, performance and scalability aspects of ThespiDIIP. We also run empirical experiments using YCSB to show the efficacy of the approach for a variety of workloads and a number of conditions, determining that integrity invariants are preserved in a causally-consistent distributed database, whilst minimising latency in the user's critical path.

Keywords: Distributed integrity invariant preservation
Causal consistency · Distributed databases · Actor model · Middleware

1 Introduction

The CAP theorem [9, 15] proves that having both Availability and Partition Tolerance within databases (DBs) that guarantee Consistency [19] is not possible. Strong Consistency (SC) is the strongest type of consistency offered by traditional DBMSs that favour consistency over availability in the event of network partitions.

Distributed datacentres (DCs) have led to a wide adoption of DBs that forego strong data consistency in favour of availability and partition tolerance to provide the scalability and high-availability properties sought by enterprise-scale online applications. Popular DBMSs in this area offer Eventual Consistency (EC) [31], a weak consistency model which guarantees that given no new WRITE operations, all DCs (i.e. distributed partitions) of the DB eventually converge to the same state.

EC is relatively easy to achieve, and does not suffer from the performance limitations of distributed algorithms, such as Paxos [24], that attempt to achieve a degree of availability and SC in a distributed environment. However, EC shifts

data safety and consistency responsibilities to the application layer, giving rise to a new set of problems [14].

Causal Consistency (CC) [1] is weaker than SC, but stronger than EC, and has been proven to be the strongest type of consistency that can be achieved in a fault-tolerant, distributed system [25]. Informally, CC implies that readers cannot find a version of a data element before all the operations that led to that version are visible [5].

In our previous work, we presented Thespis [10], a middleware approach that offers causal consistency encapsulated in a layer of REST services, such that its integration within enterprise applications is straightforward.

We now consider *Integrity Invariance Preservation*. *Integrity Invariants* are defined as application-specific operation pre-conditions, or rules that determine whether an operation on a data element should be accepted or not. In a distributed DBMS designed for low latency and high availability, such as Thespis [10], operations can be accepted at any data centre (DC), and propagated to other DCs asynchronously. The result of an operation accepted at any replica can be propagated to a remote replica at a time when the operation's pre-condition no longer holds [6], leading to an anomaly in the integrity invariant.

Therefore, although DBMSs that implement CC tackle an important class of problems for enterprise applications (as noted in [10]), a class of applications require even stronger guarantees from their DBMS to preserve business-specific integrity invariants.

Specifically, in this paper we focus on integrity invariants for data values that must be satisfied according to a Linear Arithmetic Inequality (LAI) constraint [7]. These are a set of problems that involve resource allocation [23], such as operations on bank accounts (integrity invariants define that withdrawals cannot request more than the available funds) and order fulfillment operations (an order can be accepted only if there is enough stock). Although important in real-world applications [4], these types of integrity invariants are not *I-confluent* [3], meaning they cannot be preserved by concurrent transactions without coordination.

Three main aspects are of interest for distributed integrity invariant preservation (DIIP): (1) **Safety**, i.e. that the integrity of the data, as defined by a set of constraints, is preserved (2) **Liveness**, i.e. that the system can still operate in the face of site failures or network partitioning (typically indistinguishable) (3) **Performance**, including operation completion time and system throughput. These three aspects can be orthogonal. Safety can be guaranteed by consensus-based algorithms, such as the Two-Phase Commit Protocol (2PC) [17], but these fail to guarantee liveness under some failure conditions. Other protocols such as the Three-Phase Commit Protocol (3PC) improve liveness [8], at the cost of performance.

Requirements for a distributed DBMS include: (1) Guaranteeing safety by ensuring that integrity invariants are preserved at all times (2) Maximising Liveness, through securing availability in the face of non-Byzantine failures such as

network partitions and (3) Maximising performance by optimising the costs of distributed transaction processing and maximise application throughput.

We review related work on distributed data processing in a failure-prone environment, and then introduce a model, based on distributed value processing, that is suited for preserving integrity invariants. As an informal proof of correctness, this model is also empirically evaluated against specific problem domains treating fungible entities (i.e. an entity whose individual units are equivalent can be mutually substituted [21]). Finally, we present our evaluation in terms of scalability and performance through empirical experiments using YCSB [12], which show that our objectives are possible to achieve with minimal performance overheads, and in a manner that integrity invariants can be preserved in a causally-consistent distributed database without impacting system latency for the end-user.

2 Literature Survey

2.1 Distributed Integrity Invariant Preservation Strategies

A number of approaches to DIIP for fungible entities have been described, including the Data-Value Partitioning (DVP) approach [29], the Demarcation Protocol [7] and the Generalised Site Escrow (GSE) [23]. In this section we give a brief description overview of DVP and GSE.

Data-Value Partitioning. The concept of DVP and Virtual Messages (VMs) was introduced in [29]. The authors describe problems with blocking protocols in distributed transaction processing i.e. protocols that are known to *block*, or not reach a decision to commit or abort a transaction, in a (bounded) number of steps.

A transaction can also only be considered to be truly non-blocking if it can execute at only oneDC, without co-ordination with other DCs [29]. The authors then define an approach where fungible data items are partitioned across the DCs that participate in a distributed database cluster. For a distributed database, each DC j in a cluster of J DCs, is assigned a part of the whole value N (a value of a business entity instance) such that, at all times, an integrity invariant $\sum_1^J N_j \leq N$ is observed. In [29], algorithms that enable distributed transactions are described. DVP does not require global synchronisation and, as all transactions execute locally and autonomously in a DC, the algorithm is also non-blocking and guarantees liveness.

Generalised Site Escrow. The Generalised Site Escrow (GSE) [23] generalises Transaction Escrow (TE) [27] for a distributed database scenario using quorum locking [18]. With TE, transactions put resources pessimistically *in escrow*. An escrow reservation operation is successful if the total amount of resource available, minus the amounts that have already been escrowed by other transactions

(both committed and un-committed), is sufficient. If the amount is not sufficient, a transaction is blocked waiting for sufficient resources, and eventually it aborts after a pre-defined timeout. In GSE, DCs in the cluster exchange gossip messages in order to propagate a *timetable* TT_i , as well as a set of updates u . TT_i is defined as a two-dimensional array: each element $[i, i]$ is incremented once a transaction starts at DC i , and each element $[i, k]$ indicates the last transaction started at DC k , known at on a DC i . GSE, in contrast to DVP, is a replication-based approach [11]: tokens available are replicated across all DCs, such that each DC can execute local escrow transactions.

2.2 Implementations and Approaches

The problem of satisfying integrity invariants for partitionable, fungible entities has been widely explored in the literature. In [11], a number of approaches to token-based re-distribution are evaluated. The main focus is on an evaluation of different re-distribution strategies when implementing DVP as described by [29] through simulated scenarios. A secondary objective is to compare and contrast partitioning strategies (such as DVP) and replication strategies (such as GSE) when handling tokenisable data in a distributed environment. The authors conclude that DVP achieves better throughput than GSE, with a lower amount of messages exchanged across DCs in the cluster. In [30], a multi-agent-based value partitioning and allocation strategy is proposed to improve response time and find an optimal token allocation strategy. The Quantity Transfer approach is also discussed in [2], which focuses on averaging the quantity of tokens of a partitionable entity safely across a distributed cluster.

3 ThespiDIIP Approach

Our approach to DIIP for LAI constraints is modeled as a cost-optimisation problem. As noted by [16], the cost is the communication between DCs in a cluster, i.e. communication between two sites to exchange one token has the same cost as exchanging hundreds of tokens between the same two sites. More specifically, we model the problem as a cost optimisation problem *in the critical path*. Although network access is still costly, it does not impact application throughput if it happens asynchronously and in the background. We also base our approach on the assumption that DCs participating in a cluster are connected via broadband connections. From the literature review, we can summarise that an approach for resource allocation of an entity instance is concerned by four major aspects, namely **(1) Initial allocation** - how many tokens will each DC in the cluster start with? **(2) Borrowing strategy** - when should a particular DC in the cluster request more tokens from its peers? **(3) Borrowing quantity forecast** - how many tokens will be borrowed? and **(4) Borrowing clique formation** - from which DC in the cluster will a borrower request tokens?

Although works in the literature compare and contrast different strategies, as in [11], we argue that it is difficult to propose a one-size-fits-all solution

to scenarios involving fungible entities. For example, in the class of problems where resources are implicitly related (as in a bank account implicitly related to only one user) then, even in the face of skewed workloads (majority of bank users are being served by a particular DC in a cluster), we cannot consistently conclude that a user U (i.e. an account holder), connected to an under-worked DC in a cluster, will not perform a large withdrawal and therefore require a large amount of funds to be available at the local DC. Also, the amount of transactions happening at a particular DC (i.e. a transaction rate) cannot accurately pre-empt the amount of funds needed at a local site, since it is very likely that the transactions are being carried out by different users. We propose a Hierarchical Pre-Emptive Relocation (HPR) approach, which aims to optimise the resource allocation cost in the critical path for workloads over implicitly-related entities. To the best of our knowledge, this is a novel approach to resource allocation optimisation.

3.1 Hierarchical Pre-emptive Distribution

HPR aims to exploit the structure of the underlying RDBMS, as well as application-specific configuration, to optimise data relocation costs in the critical path. A relational database (RD) can be modeled as a Directed Graph [28], with vertices representing tables and foreign keys (FKs) forming directed edges between vertices. A class of applications can benefit from data relocation strategies that exploit the nature of RDs. For example, in a banking scenario, a hypothetical RD model is illustrated in Fig. 1, showing a One-to-Many relationship between users and bank accounts. In this domain, if user U logs in at site S , then it is desirable that the system in the background starts re-allocating any partitionable resources linked to that user record to site S . This improves the probability that, once user U comes to interact with the data (e.g. to withdraw funds), the transaction can be executed locally. This is similar to the *Primary* re-distribution strategy proposed by [11], but our approach infers this from the application’s RD model.

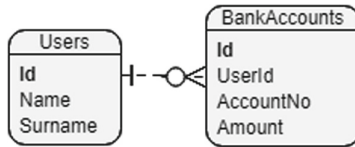


Fig. 1. Bank account RDBMS: one user having many bank accounts

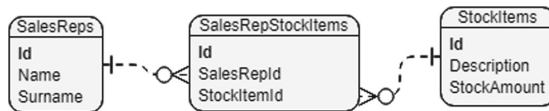


Fig. 2. Sales representatives RDBMS

This approach is applicable also to other problem domains, such as stock of a particular commodity that can be sold by a number of sales persons. A hypothetical RD model for this domain, showing a Many-to-Many relationship, is illustrated in Fig. 2. Given a sales rep U that logs in at site S , we can then infer that the *Primary* location of X amount of a particular stock item can be allocated immediately to site S . This can also be extended with inherent information captured from application logic and database information (e.g. historic sales information can influence the allocation amount). We formulate a HPR approach in-line with the four aspects outlined in this section.

For Initial Allocation, each DC in the cluster starts with an equivalent number of tokens.

As a Borrowing Strategy, a DC in the cluster will start requesting tokens for partitionable resources once a READ operation for any entity linked to the resource is served.

For Borrowing Quantity Forecast, we use a weighted strategy to forecast how many tokens does a DC require, based on the cardinality of relations. Specifically: Let $amt_b \equiv$ the amount of tokens to be borrowed, $amt_g \equiv$ the amount of global tokens available in the cluster, $T_e \equiv$ the database table storing instances of an entity e , $T_t \equiv$ the database table storing instances of a partitionable entity, $F_x^m = \{T_e, T_t\}$, in other words a link between the two tables, of cardinality M . Then $amt_b = \frac{amt_g}{M}$. Therefore, the amount to be borrowed is a ratio of the entities that share it. Intuitively, in a 1-Many relationship (as illustrated in Fig. 1), $M = 1$, so the borrower DC will attempt to borrow all available tokens as soon as a user entity instance is read. This is similarly applicable to Many-Many relationships, (such as is illustrated in Fig. 2), but we not considering Many-1 relationships i.e. we assume that the entity that has a partitionable attribute is on the right side of a directed relationship.

For the **Borrowing Clique Formation**, we adopt a simple clique formation tactic: a borrower DC can contact its lender peers in descending order of the amount of available tokens. This approach is used since: (1) For 1-Many relationships, a borrower needs to relocate all available tokens, so it needs to contact all its peers, (2) In Many-Many relationships, there is a high probability that M is larger than the number of DCs in the cluster, and so the local amount (as defined by the initial allocation strategy) is already sufficient and (3) Borrowing happens in the background, outside of the critical path.

3.2 System Model

Thespis DIIP is implemented as an extension to Thespis [10]. DVP logic is modeled as a hierarchy of distributed Actors [20], that are responsible to manage partitionable tokens across the cluster. Here, we focus on the *DVP Manager* component of the middleware, as illustrated in Fig. 3.

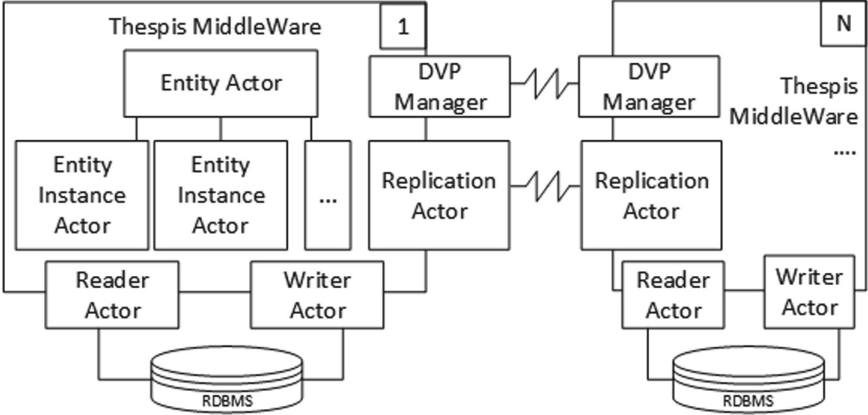


Fig. 3. System model

4 Methodology

4.1 Definitions

We define a few concepts that are used in our design, namely (1) **Tokenisable property (TP)**: a property of an entity that can be split into tokens. Referring to Fig. 1, an example of a tokenisable property is *BankAccounts.Amount*, (2) **Tokenisable Entity (TE)**: an entity that has at least one tokenisable property. Referring to Fig. 1, an example is *BankAccounts*, (3) **Parent Entity (PE)**: the entity that owns a tokenisable entity. Referring to Fig. 1, an example is *Users*, which owns *BankAccounts*, and (4) **Hierarchical Relationship (HR)**: a description of the relationship between a PE and a TE, defined by a triplet composed of $\langle PE, TP, Cardinality \rangle$.

4.2 Application-Specific Configuration

A small amount of application-specific metadata is required to be provided at design-time in order to define HRs. This is required since: (1) The middleware cannot assume that the data definition language (DDL) semantics of the underlying RDBMS can be used to natively identify TPs and TEs, (2) Deterministically inferring the PE given a TP participating in a Many-to-Many relationship, using only FK information from the underlying database, is non-trivial and can yield an HR which is not relevant for the application’s use case and (3) Application data might influence the cardinality of an HR, and optimise token allocation. HRs are defined in a simple structure that identify the PE, TE, TP and a Cardinality. The Cardinality can be expressed either as a fixed numeric value, or as a numeric-returning function that can be executed against the underlying RDBMS. Table 1 illustrates the metadata that defines the HRs for the structures in Figs. 1 and 2.

Table 1. Example HR metadata configuration

PE	TE	TP	Cardinality
Users	BankAccounts	Amount	1
SalesReps	StockItems	StockAmount	SELECT COUNT(DISTINCT SalesRepId) FROM SalesRepStockItems WHERE StockItemId=TE.Id

4.3 Event Concurrency and Parallelism

In each DC, the middleware maintains a cluster of actors that manage the DVP functionality, as illustrated in Fig. 4. The DVP Manager can receive 5 types of messages: *AcquireTokens*, *GetStatus*, *LendTokens*, *BorrowTokens* and *ClearTokens*. The DVP Manager’s behaviour handles the different messages depending on their type. *AcquireTokens* and *GetStatus* messages are forwarded to a child actor identified by the key *ParentEntityType-ParentEntityId*, whilst *LendTokens*, *BorrowTokens* and *ClearTokens* messages are forwarded to a relevant child actor. By the nature of the actor model [20], where each actor can process one message at a time, this approach ensures that, within a DC, messages relevant to each PE data item can be processed in parallel and independently, but messages for the same PE data item are handled sequentially. However, control messages (*LendTokens*, *BorrowTokens*, *ClearTokens*) can be processed in parallel and independently, but only one of each type of control process can be invoked at any point in time in any DC.

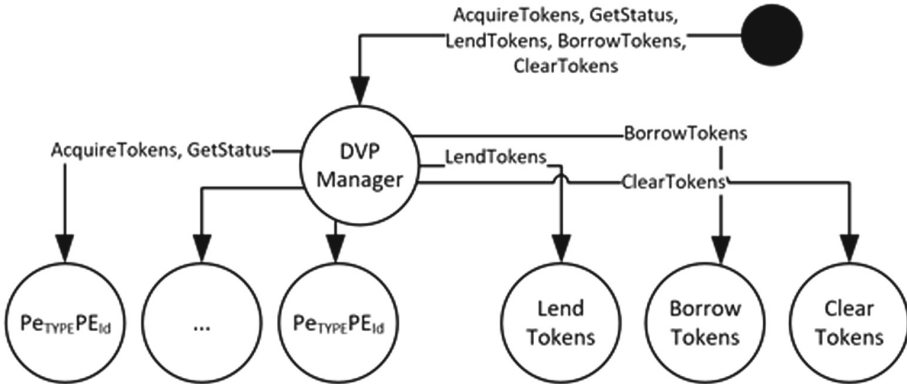


Fig. 4. Actor cluster

4.4 Token Event Persistence

Partitionable tokens in the cluster are stored in a structure as illustrated in Table 2.

Table 2. Structure of the token event store

PETYP	The type of the Parent Entity (e.g. <i>Users</i>)
PEID	The identifier of the Parent Entity for which tokens are allocated
TETYP	The type (e.g. table name) of Tokenisable Entity (e.g. <i>BankAccounts</i>)
TPNAME	The name of the tokenisable property (e.g. <i>BankAccounts.Amount</i>)
TEID	The identifier of the Tokenisable Entity
AMT	The amount of tokens. A positive value indicates that the tokens reside in the local cluster, a negative value indicates that the tokens have been lent to a remote DC
SID	The identifier of the server where the tokens reside
TIMESTAMP	The event timestamp

Algorithm 1. BorrowTokens/LendTokens

BorrowTokens (DVPTokens)/LendTokens (DVPTokens):

1. For each element in the set of DVPTokens:
 - a) Insert a record in the Token Event Store, ensuring that AMT is a positive value (for BorrowTokens) or negative value (for LendTokens)
-

4.5 Event Handling

Behaviour logic is encapsulated within an Actor cluster to handle each type of message. Algorithms 1, 2 and 3 summarise the salient logic executed for the most important messages.

4.6 Implementation

We implement ThespiDIIP on top of Thespi, using Akka.NET¹, a port of the original Akka framework² for JVM. PostgreSQL 9.5 is used as a DBMS. The ThespiDIIP extension is exposed to an end-user application via a REST interface with one endpoint, *acquiretokens*, that initiates Algorithm 3 for a given TE.

5 Performance Evaluation

We implement ThespiDIIP and evaluate our approach in the context of **(a) Correctness**: does ThespiDIIP preserve integrity invariants?; **(b) Scalability**: how does ThespiDIIP scale as the number of DCs grow, and the number of partitionable and fungible entities grow?; and **(c) Applicability**: as the cluster and data set grow, would a waiting state in the hot path be minimised or eliminated?

¹ Akka.NET - <http://getakka.net/>.

² Akka - <http://akka.io/>.

Algorithm 2. GetStatus

 GetStatus (PE_Type, PE_Id, TE_Type, TP_Name, RequestorServerName):

1. Let TE denote the set of TEs of type TE_Type, related to the PE identified by PE_Id
 2. For each $te \in TE$:
 - a) Let AMT = the value of the TP
 - b) Let AMT_{LND} = the amount of the TP that has been borrowed by other servers in the cluster from this DC i.e. the sum of AMT from the Token Event Store where AMT is negative
 - c) Let AMT_{BRW} = the amount of the TP that has been transferred from the other servers in the cluster to this DC, and not allocated to a PE i.e. the sum of AMT from the Token Event Store where AMT is positive and PEID is NIL
 - d) Let $AMT_{AVAIL} = AMT_{BRW} - AMT_{LND}$
 - e) $DVPStatus = \{te_{ID}, AMT, AMT_{BRW}, AMT_{LND}, AMT_{AVAIL}, SID\}$
 - f) Return $DVPStatus$
-

5.1 Correctness

Our design is in-line with DVP approaches in the literature (e.g. [11]) that have been shown to be correct in preserving integrity invariants and solving issues such as the “double-spending” problem.

By Algorithm 3, tokens for a TE are gathered in the local DC. Operations that need to manipulate the value of any TP in the given TE can consequently safely execute locally, given that the local DC holds enough tokens to fulfill the operation. In parallel, a DC can only acquire tokens for a TE if they have been borrowed from a remote DC. In other words, a concurrent operation on the same TE in a remote DC that has lent tokens to the local DC cannot be satisfied unless tokens are requested again. Hence, an integrity invariant such as $BankAccount.Amount > 0$ is preserved across the distributed data cluster.

In order to illustrate with an example, we focus on the “Bank Account” scenario, having the data model illustrated in Fig. 1. For our example, we assume that there exists a user U that owns one bank account B , with a balance of 100. We also assume that the system is hosted in four DCs (DC_1, DC_2, DC_3, DC_4), configured with the metadata defined in the first row of Table 1, and that user U has been logged in to DC_1 . This initiates Algorithm 3 in DC_1 .

By Algorithm 3 running in DC_1 , therefore:

- $DC_L = DC_1$
- $L = 1$
- $DCS = \{DC_1, DC_2, DC_3, DC_4\}$
- $RDCS = \{DC_2, DC_3, DC_4\}$
- At the very beginning, each DC owns 0 tokens of bank account instance B .
- Hence, by Step 7(c) of Algorithm 3, tokens need to be initially distributed.
- In-line with the “Initial Allocation” strategy defined in Sect. 3.1, at Step 8, each DC receives a “Borrow” message, instructing it to borrows $\frac{100}{4} = 25$ tokens from “System”.

Algorithm 3. AcquireTokens

AcquireTokens (PE_Type, PE_Id, TE_Type, TP_Name, AMT=NIL):

1. Let DC_L denote the local DC, L the identifier of the local DC, DCS the set of all DCs in the cluster, $P_i \in DCS$ denote the DVP Manager actor in DC_i and $RDCS = DCS - DC_L$ (i.e. the set of all remote DCs in the cluster)
 2. For each $P_i \in DCS$, send a *GetStatus* message
 3. Let $DVPStatus$ denote the set, with size M , of statuses received as a response to each *GetStatus* message
 4. Let TE denote the set of Tokenisable Entities of type TE_Type identified across $DVPStatus$
 5. Let DVP_{BORROW}^i denote an empty set of $DVPToken$, holding a list of tokens that can initiate a BORROW operation in DC i
 6. Let DVP_{LEND}^i denote an empty set of $DVPToken$, holding a list of tokens that can initiate a LEND operation in DC i
 7. For each $te \in TE$
 - a) $AMT_{LND} = \sum_{i=1}^m DVPStatus_i.AMT_{LND} [DVPStatus_i.id = te.id]$
 - b) $AMT_{BRW} = \sum_{i=1}^m DVPStatus_i.AMT_{BRW} [DVPStatus_i.id = te.id]$
 - c) IF $AMT_{LND} = AMT_{BRW} = 0$:
 - i. $AMT_{TOBORROW} = \left\lfloor \frac{AMT_{AVAIL}}{\sum_{i=1}^m [DVPStatus_i.id=te.id]} \right\rfloor$
 - ii. For each $P_i \in DCS$, formulate $DVPToken = \{NIL, NIL, TE_Type, TEID, TP_Name, AMT_{TOBORROW}, \text{"System"}\}$ and add it to DVP_{BORROW}^i
 - d) ELSE
 - i. Let CRD denote the relationship cardinality of te as calculated by the Cardinality Function
 - ii. Let AMT_{RQD} denote the amount of tokens required at DC_L
 - iii. IF $AMT \neq NIL$, THEN $AMT_{RQD} = AMT$ ELSE $AMT_{RQD} = \frac{AMT_{AVAIL}}{CRD}$
 - iii. Let AMT_{AVAIL} denote the amount of tokens available of te for PE_Id in DC_L i.e. the sum of AMT from the Token Event Store WHERE AMT is positive AND (PEID is NIL or PEID=PE_Id)
 - iv. Allocate $MIN(AMT_{AVAIL}, AMT_{RQD})$ in DC_L
 - v. $AMT_{TOBORROW} = AMT_{RQD} - AMT_{AVAIL}$
 - vi. IF $AMT_{TOBORROW} > 0$
 - I. $DVP_{AVAIL} = \forall s \in DVPStatus$ where $s.AMT_{AVAIL} > 0$ AND $s.SID \neq L$.
 - II. For each $s \in DVP_{AVAIL}$ ordered in descending order of AMT_{AVAIL} , formulate $DVPToken = \{PE_Type, PE_Id, TE_Type, TEID, TP_Name, MAX(s.AMT_{AVAIL}, AMT_{TOBORROW}), L\}$ until $AMT_{TOBORROW}$ is satisfied, and add it to DVP_{LEND}^i
 8. For each DVP_{BORROW}^i where $DVP_{BORROW}^i \neq \emptyset$, for $P_i \in PEERS$, send a *BorrowToken* message with parameter DVP_{BORROW}^i
 9. For each DVP_{LEND}^i where $DVP_{LEND}^i \neq \emptyset$ for $P_i \in RDCS$, send a *LendToken* message with parameter DVP_{LEND}^i
 10. IF $\exists DVP_{BORROW}^i$ where $DVP_{BORROW}^i \neq \emptyset$, go to Step 1
-

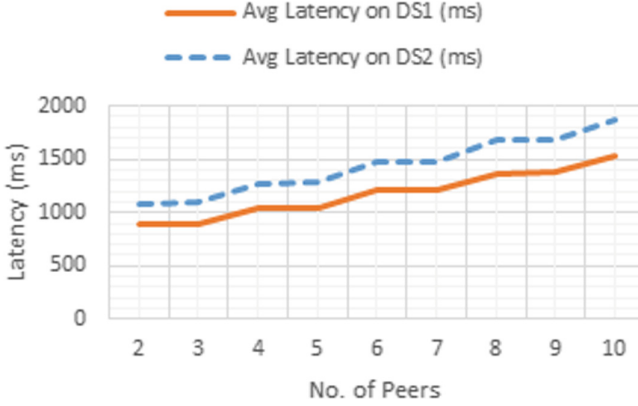


Fig. 5. Benchmark: operation latency *vs.* cluster growth on DS1 (small data set) and DS2 (large data set)

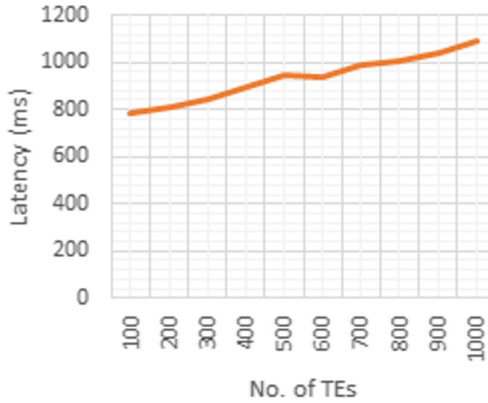


Fig. 6. Operation latency *vs.* data set growth (small cluster)

- This causes the condition in Step 10 of Algorithm 3 to match, and another iteration of the algorithm starts.
- At this stage, each DC owns 25 tokens of bank account B .
- By Step 7(d) of Algorithm 3, the amount of tokens required in DC_1 , given the cardinality function configured in the metadata, is calculated as $\frac{100}{1} = 100$, of which 25 already reside in DC_1 .
- Therefore, at Step 8, each DC receives a “Lend” message, instructing it to lend all of its 25 tokens to DC_1 , finally arriving at a state where DC_1 holds all the 100 tokens of bank account B .

By the end of the execution, user U is allowed to perform a transaction of any amount in DC_1 on bank account B , without requiring any further co-ordination with remote DCs. More importantly, the integrity invariant $B.balance \geq 0$ is preserved: the remote DCs own zero tokens of bank account B , and therefore no remote transaction on bank account B can execute concurrently to violate the integrity invariant.

The state of the tokens is persisted in the Token Event Store in each DC, as defined in Sect. 4.4. Table 3 further illustrates a summary of the status of the Token Event Store, at the Local and Remote DCs, with this progression of Algorithm 3.

Table 3. Token event store progression

Stage	Local DC (DC_1)	Remote DC ($DC_{2,3,4}$)
1: Each DC owns 0 tokens of bank account B	AMT SID	AMT SID
	- -	- -
2: Each DC owns 25 tokens of bank account B	AMT SID	AMT SID
	25 SYSTEM	25 SYSTEM
3: Each DC lends all of its tokens to DC_1	AMT SID	
	25 SYSTEM	AMT SID
	25 DC2	25 SYSTEM
	25 DC3	-25 DC1
	25 DC4	

5.2 Performance Characteristics

We test our implementation on Google Cloud Platform³, with two DCs, “europe-west1-d” and “us-west1-b”. Each DC hosts multiple installations, where each installation consists of a middleware and a database server. One Load server is hosted in “europe-west1-d”. Infrastructure specifications are as those in [10]. The evaluation is performed on a data set which mimics a Bank situation, in other words on a 1-Many data schema, with *User* as the PE and *BankAccount* as the TE.

Cluster Growth. We first focus on the performance in the context of cluster growth i.e. how the performance of an *AcquireTokens* operation varies as the number of DCs grow. Tests execute on a small data set (DS1) of 1 PE linked to 2 TEs, using YCSB to benchmark the *AcquireTokens* REST endpoint on ThespiDIIP. Each *AcquireTokens* operation: (1) Assumes no tokens have been

³ Google Cloud Platform - <https://cloud.google.com/>.

previously distributed and first re-distributes tokens for both TEs across the cluster. (2) Gets the tokens to the local cluster for both TEs. This ensures that, subsequently, any operation on each TE owned by the PE can succeed locally without waiting time, whilst any operation on remote clusters will fail due to 0 available tokens, hence preserving the integrity constraint, (3) Each benchmark executes the same operation 10 times, and is repeated 3 times, with the average operation latency recorded for each run, and (4) Cluster is grown from 2 to 10 peers. At each phase, a peer in the “next” DC is added e.g. 2 peers = 1 peer in EU, 1 in US; 3 peers = 2 peers in EU, 1 in US; 4 peers = 2 peers in EU, 2 peers in US; and so on. Results are illustrated in Fig. 5.

Data-Set Growth. Next we focus on data-set growth, measuring how the performance of an *AcquireTokens* operation varies as the number of TEs grow. The benchmark is similar to Sect. 5.2, but the number of peers is kept constant at 2 (1 peer in the US DC and 1 peer in the EU DC) and the number of TEs linked to 1 PE is varied from 100 to 1000. Results are illustrated in Fig. 6.

Cluster Growth with Data-Set Growth. Our last set of benchmarks focus on cluster growth with a large data-set (DS2). This benchmark is similar to the one in Sect. 5.2, but using a data set of 1000 TEs. Results are also illustrated in Fig. 5.

5.3 Observations

A number of observations are elicited from the results in Sect. 5.2. From Fig. 5, we note that operation latency is primarily impacted by inter-DC communication. In other words, scaling the distributed database as multiple installations within the same DC has negligible impact on the latency of the *AcquireTokens* operation. This corroborates the conclusions by [16], and emphasises the impact of network latency on operation latency. Moreover, taking the case of a DC acquiring the tokens for a PE linked to many TEs, in the context of a large cluster size, the maximum operation response time remains below 2s, which is significant in the context of system usability studies. A number of studies [13, 22] indicate a common page *dwell time* of 30s on user-facing interfaces, with aggressive dwell times going down to 10s. On the other hand, *tolerable wait time* for the same interfaces is considered much less, going even down to 2s [26, 32]. Considering a Banking application as a common application usage for Thespi-DIIP, a typical UI/UX workflow is illustrated in Fig. 7. Given our results, and even assuming aggressive dwell times together with a worst-case scenario on the asynchronous DVP operation would allow the *Request Withdrawal* operation to complete locally without waiting.

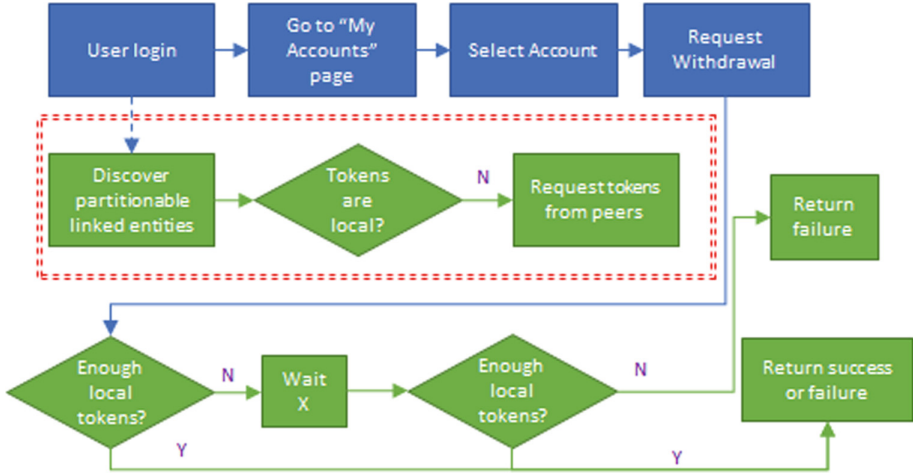


Fig. 7. DVP process in a typical UI/UX workflow

6 Conclusions

ThespisDIIP improves Thespis with DIIP, over and above the original CC guarantees, tackling an important class of problems that are outside of the scope of CC. The implementation of ThespisDIIP, a middleware offering both CC and DIIP over a commercial RDBMS is a novel idea.

The first contribution of this paper is the Hierarchical Pre-Emptive Relocation (HPR), a theoretical model that defines an approach to DVP, taking into consideration the relational structure of the business data model.

Our second contribution is the implementation of ThespisDIIP, a system that offers both CC and integrity invariant preservation. Our implementation also gives details as to how HPR can be easily implemented using the Actor model to manage operation concurrency.

The final contribution is the observation, based on empirical performance analysis, that our approach and implementation allows asynchronous token distribution outside of the critical path of a regular user journey, allowing transactions to safely execute locally, eliminating waiting time and optimising the user's experience.

Our design is unique in exploiting the structure of the underlying RDBMS to trigger background DVP operations when needed. In contrast to, for example, the approaches defined by [11], we treat the scenario of an important class of applications that do not benefit from token redistribution based on global transaction rate observations. Secondly, by integrating our solution in the Thespis middleware, our approach does not require any additional client libraries, and the complexities of DVP are abstracted: with simple design-time identification of TP's, the middleware has enough knowledge to fire asynchronous DVP operations when a PE is requested.

Acknowledgement. This work is partly funded by the ENDEAVOUR Scholarship Scheme (Malta), part-financed by the European Union – European Social Fund (ESF) under Operational Programme II – Cohesion Policy 2014–2020.

References

1. Ahamad, M., Neiger, G., Burns, J.E., Kohli, P., Hutto, P.W.: Causal memory: definitions, implementation, and programming. *Distrib. Comput.* **9**(1), 37–49 (1995)
2. Almeida, P.S., Shoker, A., Moreno, C.B.: Exactly-once quantity transfer (2015)
3. Bailis, P., Fekete, A., Franklin, M.J., Ghodsi, A.: Coordination avoidance in database systems. *Proc. VLDB Endow.* **8**(3), 185–196 (2014)
4. Bailis, P., Fekete, A., Franklin, M.J., Ghodsi, A., Hellerstein, J.M., Stoica, I.: Feral concurrency control: an empirical investigation of modern application integrity. In: *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data*, pp. 1327–1342. ACM (2015)
5. Bailis, P., Ghodsi, A., Hellerstein, J.M., Stoica, I.: Bolt-on causal consistency. In: *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pp. 761–772. ACM (2013)
6. Balegas, V., Prego, N., Duarte, S., Ferreira, C., Rodrigues, R.: IPA: invariant-preserving applications for weakly-consistent replicated databases. *arXiv preprint arXiv:1802.08474* (2018)
7. Barbará-Millá, D., García-Molina, H.: The demarcation protocol: a technique for maintaining constraints in distributed database systems. *VLDB J. - Int. J. Very Large Data Bases* **3**(3), 325–353 (1994)
8. Borr, A.: Transaction monitoring in encompass. In: *Proceedings of 7th VLDB* (1981)
9. Brewer, E.A.: Towards robust distributed systems. In: *PODC*, vol. 7 (2000)
10. Camilleri, C., Vella, J.G., Nezval, V.: Thespis: actor-based causal consistency. In: *2017 28th International Workshop on Database and Expert Systems Applications (DEXA)*, pp. 42–46. IEEE (2017)
11. Cetintemel, U., Ozden, B., Franklin, M.J., Silberschatz, A.: Design and evaluation of redistribution strategies for wide-area commodity distribution. In: *2001 21st International Conference on Distributed Computing Systems*, pp. 154–161. IEEE (2001)
12. Cooper, B.F., Silberstein, A., Tam, E., Ramakrishnan, R., Sears, R.: Benchmarking cloud serving systems with YCSB. In: *Proceedings of the 1st ACM Symposium on Cloud Computing*, pp. 143–154. ACM (2010)
13. Danaher, P.J., Mullarkey, G.W., Essegai, S.: Factors affecting web site visit duration: a cross-domain analysis. *J. Market. Res.* **43**(2), 182–194 (2006)
14. Elbushra, M.M., Lindström, J.: Eventual consistent databases: state of the art. *Open J. Databases (OJDB)* **1**(1), 26–41 (2014)
15. Gilbert, S., Lynch, N.: Brewer’s conjecture and the feasibility of consistent, available, partition-tolerant web services. *ACM SIGACT News* **33**(2), 51–59 (2002)
16. Golubchik, L., Thomasian, A.: Token allocation in distributed systems. In: *Proceedings of the 12th International Conference on Distributed Computing Systems*, pp. 64–71. IEEE (1992)
17. Gray, J.N.: Notes on data base operating systems. In: Bayer, R., Graham, R.M., Seegmüller, G. (eds.) *Operating Systems. LNCS*, vol. 60, pp. 393–481. Springer, Heidelberg (1978). https://doi.org/10.1007/3-540-08755-9_9

18. Herlihy, M.: Concurrency versus availability: atomicity mechanisms for replicated data. *ACM Trans. Comput. Syst. (TOCS)* **5**(3), 249–274 (1987)
19. Herlihy, M.P., Wing, J.M.: Linearizability: a correctness condition for concurrent objects. *ACM Trans. Program. Lang. Syst. (TOPLAS)* **12**(3), 463–492 (1990)
20. Hewitt, C., Bishop, P., Steiger, R.: A universal modular actor formalism for artificial intelligence. In: *Proceedings of the 3rd International Joint Conference on Artificial Intelligence*, pp. 235–245. Morgan Kaufmann Publishers Inc. (1973)
21. Ivica, S.B., Aleksandar, M.R., Radomir, M.A.: Crypto-currency and e-financials. *J. Econ. Law* **132** (2014)
22. Jansen, B.J., Spink, A.: An analysis of web documents retrieved and viewed. In: *International Conference on Internet Computing*, pp. 65–69. Citeseer (2003)
23. Krishnakumar, N., Bernstein, A.J.: High throughput escrow algorithms for replicated databases. In: *VLDB*, vol. 1992, pp. 175–186 (1992)
24. Lamport, L.: The part-time parliament. *ACM Trans. Comput. Syst. (TOCS)* **16**(2), 133–169 (1998)
25. Mahajan, P., Alvisi, L., Dahlin, M.: Consistency, availability, and convergence. University of Texas at Austin Technical report, 11 (2011)
26. Nah, F.F.-H.: A study on tolerable waiting time: how long are web users willing to wait? *Behav. Inf. Technol.* **23**(3), 153–163 (2004)
27. O’Neil, P.E.: The Escrow transactional model. *ACM Trans. Database Syst.* **4**(11), 405–430 (1986)
28. Radev, R.: Representing a relational database as a directed graph and some applications. In: *Balkan Conference in Informatics*, p. 1 (2013)
29. Soparkar, N., Silberschatz, A.: Data-valued partitioning and virtual messages. In: *Proceedings of the Ninth ACM Symposium on Principles of Database Systems*, pp. 357–367. ACM (1990)
30. Takaishi, M., Leguizamo, C.P., Kimura, S., Takanuki, R.: Autonomous multi-agent-based data allocation technology in decentralized database systems for timeliness. In: *Proceedings of Autonomous Decentralized Systems, ISADS 2005*, pp. 25–32. IEEE (2005)
31. Vogels, W.: Eventually consistent. *Commun. ACM* **52**(1), 40–44 (2009)
32. Zona, R.: The economic impacts of unacceptable web site download speeds. Technical report, Research report (1999). <http://www.zonaresearch.com>



Privacy Issues for Cloud Systems

Christopher Horn¹ and Marina Tropmann-Frick²(✉)

¹ Lufthansa Industry Solutions, Schützenwall 1, 22844 Norderstedt, Germany
christopher.horn@lhind.dlh.de

² Hamburg University of Applied Sciences, Berliner Tor 5, 20099 Hamburg, Germany
marina.tropmann-frick@haw-hamburg.de
<https://www.lufthansa-industry-solutions.com>
<https://www.haw-hamburg.de/ti-i>

Abstract. In this paper we discuss the issue of privacy in the cloud. In the area of privacy in the cloud we take a first look, which components take part in a cloud privacy system. This starts with influencing factors in hardware production, customer and provider privacy. With the cloud privacy it is valuable to handle Privacy as a Service (PaaS) with its security protocols. Not only is it essential to include trusted third parties, the cloud providers itself have to be strict with their code of conduct in the cloud (cloud of conduct). The fast paste economy has brought up ethic issues over the years. There are different kinds of cloud types which more or less harmonize with privacy ethics. These topics need to be viewed in the context of a cloud privacy system.

Keywords: Cloud-computing · Code of conduct · Cloud of conduct
Cloud privacy · Privacy as a Service · Economic ethics
Privacy ethics · Cloud ethics

1 Introduction

In our fast paced globalised world it is not easy to know where our privacy is offended. Especially when many companies work together for a customer product. When we want to discuss the privacy in the cloud, we have to go to the start of cloud production. This paper is inspired by the ‘General Data Protection Regulation’ (GDPR), which started on the 25.05.18. This new European regulation mainly strengthens the users privacy with e.g. Cloud-Systems. Users can ask for a detailed profile of their stored data. Further, companies need to ask users, if they are allowed to give the user data to third parties. The GDPR is a huge workload for every company in Europe, because they need to log every piece of user data, where it is used, where it will be send etc. The Europeans are one step nearer to transparency of today’s massive data collection. This transparency should be a fundamental point for cloud systems and the main issue for this paper and its view on the frame of a cloud privacy trust system.

1.1 Privacy Chain

Many people nowadays mix data privacy with data security. That is why it is necessary to get a clear distinction of privacy and security.

Privacy. Means the protection of personal information against the access of not privileged third parties. It means, in the context of e.g. healthcare data stored in the cloud, a patient decides how a party controls her/his patient details.

Security. Is defined about the three characteristics confidentiality, availability and integrity. It includes various aspects e.g. defend personal data against exposure, modification or destruction. Again in the respect of healthcare data in the cloud, it means a hospital needs to guarantee the protection of patient details.

All in all, privacy and security have different objectives and are not necessarily addressing each other. If a system has a security concept, it does not mean it implements a good privacy. Especially if the confidentiality and integrity is broken. But to ensure good privacy it is better to have a security concept [2].

Cloud providers like Microsoft with Azure, give a lot of documentation and source code, to let their customers make the best out of the Cloud-System. Customers can use the Github code and variate the code to protect their own data or they use Azure data protection services. Providers of Cloud-Systems have also privacy rules, which are included in the company contracts. These are good features, but it is not enough.

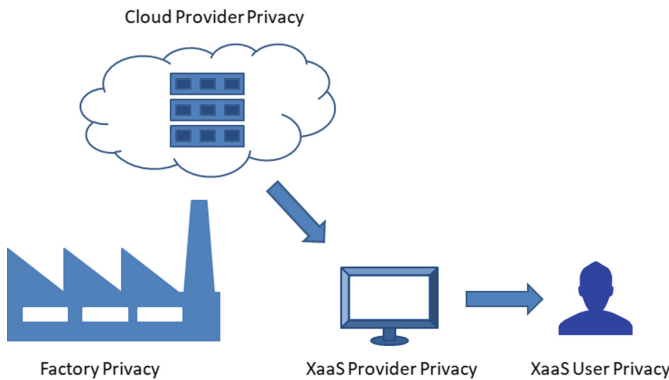


Fig. 1. Cloud privacy chain

Figure 1 shows the whole chain, where privacy needs to be protected. The three roles cloud provider, XaaS Provider and XaaS User are discussed in the following [8,9]. Not only cloud providers have to follow privacy rules. It is necessary for providers to buy infrastructure parts with the same data secrecy as the cloud provider. It is important that no infrastructure parts are manipulated in

the production facilities. There are several ways to take measures for security. The security measures go from security controls to facility monitoring until hardware production line. Important is that the production hardware e.g. hard disk, memory, CPU and more, have the highest priority. An example for protection could be a special gateway coating, where external attacks are not possible. This means that no person can install infested code or other infested hardware on the important hardware. If the gateway gets in contact with infected code/hardware, the e.g. hard disk would not work or the broken gateway part could be decoupled in the safe building of the cloud provider and there would be no harm to the actual hardware. What should be done is, that independent third party companies, specialised in testing manipulated hardware, certify the products for cloud providers. This rating should be visible in a trust system for customers, too. So that the customers know which cloud hardware they are using.

Cloud providers like AWS, Microsoft or Google inform the customers about data and user privacy through the companies sites or services. All these information are not enough and the real deal lays behind the scene. That is why it is important to bring more transparency to the users. Again this can be handled with trusted third parties who secure the data in the cloud providers infrastructure, so that even the cloud providers themselves can not use analytical systems to gain information about their customers. This privacy needs to be starting from the data upload into the cloud. Cryptographic streams with trusted third party protocols and cryptographic data storage. There are many cases where not only the cloud providers want the uploaded company data. Nowadays nearly every country in the world try to influence the cloud chain to steal or monitor user (company) data. The first address where countries ask, are the cloud providers. These providers could bend and give access to the privacy of massive datasets. These worst cases show, that the privacy in the cloud needs to be more protected, so that only the user himself can view his data.

XaaS Provider means a provider of any as a Service. The main services are Software, Platform and Infrastructure as a Service through utility computing. Of course AWS, Google, Microsoft etc. are all XaaS Provider, but for example the Azure Marketplace has many SaaS Provider. These SaaS Provider should also, like factory and cloud provider, have a strict and transparent privacy system. At the end the XaaS User should see and know exactly, from which factory to which cloud data storage, with which privacy protection level his data streams.

XaaS Users as companies, are using more and more cloud based XaaS solutions through Web applications. They see the cost efficiency and work reduction, what is a normal economic pull effect. For the private XaaS Users it is the other way around. Companies like Samsung, Apple, Google etc. use XaaS solutions to bring the data of their smartphone customers automatically into the cloud. People are indirectly forced, because the settings are mostly set to cloud storage etc. On the private sector, users are pushed into the XaaS privacy problem.

The main problem of this cloud privacy problem is, that companies always want to know their customers better, to bring out the right product at the right

time. On the other hand customers want to know their privacy in safe hand, but also want to have a good product when they need it.

1.2 Privacy as a Service

These before discussed cloud privacy issues can be seen in the context of a Privacy as a Service (PaaS) concept. This PaaS has the main priority on defending the involved information from unauthorized access, disclosure, disruption, modification, inspection, and annihilation [5,6]. The PaaS needs to be independent of the cloud chain system, country regulation and economic system itself. The best way to get such a system is an alliance of countries, who built a privacy service organisation and strengthen it with a new law. This organisation could be seen as a worldwide trusted third party, who secures the privacy of the cloud chain.

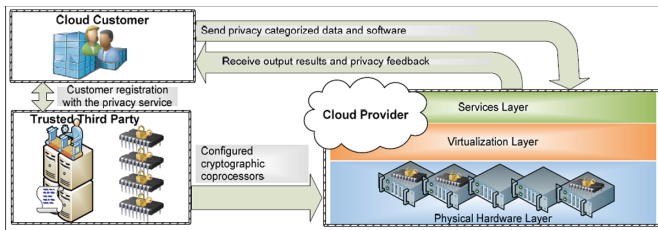


Fig. 2. PaaS cloud-based system model [4]

In Fig. 2 is the PaaS cloud-based system model [4]. This model has three actors. The trusted third party, the cloud customer and the cloud provider. This three actor model can be applied on the factory privacy, the cloud provider privacy, the XaaS provider and the XaaS User privacy case, discussed in the section before. In the following we will discuss the cloud provider privacy until the XaaS User privacy case.

In the PaaS are three trust level included.

Full Trust. Full trust level is used for insensitive data, which can be safely stored and processed without any form of encryption on the computing cloud side. The cloud provider has the full trust for data storage and processing.

Compliance-Based Trust. The compliance-based trust level is for customer data, which needs to be to be stored encrypted, because of legal compliance regulations. These legal compliance regulations can be the Health Insurance Portability and Accountability Act (HIPAA), which states how medical records and patient’s information needs to be secured. Another legal compliance regulation example is the Leach-Bliley Act. It defines the confidentiality of financial

records and banking transaction for any institution providing a financial service. Through the compliance-based trust level the customer trusts in encrypted stored data, that the cloud provider ensures with a provider-specific cryptographic key.

No Trust. The no trust level is used for highly-sensitive customer data. This data needs to be concealed from the cloud provider. The sensitive data, else than in the compliance-based trust level, needs to be stored encrypted with a customer-trusted cryptographic key. Furthermore this sensitive data should be processed in isolated cryptographic containers in the cloud. A third-party (e.g. the world trusted organisation mentioned before) configure, distribute, and maintain these isolated containers. The third-party needs to be trusted by the cloud customer and provider.

These three trust levels are one part of a so called trust system. For a XaaS User without IT-knowledge it could be shown in e.g. a web application interface, where the trust level system shows the user, if his user data would be not offended in privacy issues. He could then pin point the problem in the cloud chain and take pre-measures to secure his data privacy.

In the context of a trust level system, the encrypted storage of data is a main issue. Through the years of cloud-computing different approaches of encryption were used. One of these approaches is the Homomorphic Encryption.

Generally encryption provides a strong protection of the data against third party's insights. With the base idea of Homomorphic Encryption (HE), it is possible to preserve privacy and achieve security at the same time. HE is the expert among the encryption algorithms. For instance in a Public Key Infrastructure (PKI), it is necessary to have a public and private key for encrypting and decrypting (personal) data in the cloud. Is there a third party which wants to manipulate the data or wants to search in it, it needs the key pair to get access on the data. In this case, privacy and security are ignored one hundred percent. Homomorphic Encryption provides computations - like addition and multiplication operations - on encrypted data. It prevents a decryption of the encrypted data on any access. From the beginning of the uploaded encrypted data into a cloud, until the requested manipulation of the data, it stays encrypted and nobody gets access on the decrypted data. It has the same effect if the request manipulates the decrypted data [3]. Homomorphic Encryption can be categorised in two types of schemes based on its computation possibilities on encrypted data. On the one hand Fully Homomorphic Encryption (FHE) and on the other hand Somewhat Homomorphic Encryption (SHE). In other sources SHE is called Partially Homomorphic Encryption (PHE), which has the same meaning.

Fully Homomorphic Encryption (FHE). FHE is defined to have unlimited computations on encrypted data. The problem is, that FHE has a big efficiency problem. The more computations are executed the more noise is produced. This means after every manipulation on data, they become more and more corrupted.

Somewhat Homomorphic Encryption (SHE). She does not allow unlimited computations on the encrypted data. That is why SHE has a better performance than FHE.

These encryptions can prevent untrusted cloud providers from viewing data. There would be no untrusted cloud providers or cloud third parties if the cloud-computing companies would have a trust system through a code of conduct. In the next chapter will be discussed an approach of a cloud of conduct.

1.3 Cloud of Conduct

In the fast paste IT-Business, including Cloud-Systems, it will become necessary to have a code of conduct model for the companies themselves, for among themselves, for the government trust and the customer trust. Such a code of conduct or cloud of conduct model, can be a pillar for a trust system, which takes part in the cloud chain. This can be seen with the newest case, of privacy offence, of Facebook with Cambridge Analytica. Schwartz used different code of conduct models to find a solution, which he called universal moral values [1]. These universal moral values are built through the ‘Companies Codes of Ethics’, ‘Global Ethics Codes’ and the ‘Business Ethics Literature’. He defines a total of six universal moral values:

Trustworthiness. This includes honesty, integrity, transparency, reliability, and loyalty. To gain more trust in the cloud and XaaS provider it is necessary to fulfil the aspects. If the trustworthiness of the cloud product is not high, then the customer does not accept e.g. the price to integrate his system into the cloud and use cloud services.

Respect. This can be seen as respect for human rights. These human rights can be also referred to the privacy of the user. Furthermore, respect of human rights can also be seen in the context of our cloud chain. If the cloud hardware is produced under inhuman conditions the XaaS User would not use the cloud product.

Responsibility. This includes accountability, excellence, and self-restraint. This point is very important, because cloud provider or XaaS Provider are mostly not from the country, where they sell their product. It is important not to use loopholes in the law to escape unpleasant country laws. An always discussed issue is the storage of data in an other country or the tax system. Companies have to take the responsibility to really apply the law and not use their power to bend the law.

Fairness. This includes process, impartiality, and equity. Cloud and XaaS provider have to process the user data in the best way for the user and themselves. This even means, that the privacy has to be always the main aspect and economic issue never get the upper hand.

Caring. This includes avoiding unnecessary harm. Cloud and XaaS provider mostly avoid harm, but they just look one step ahead and not more. This means they define the rules for third parties, so that they do not have to be blamed for incursions on user privacy. When they get unlawful data from third parties or they give their data to third parties, who process the data unlawful, most companies would blame the third party. Caring means, that for example the cloud data, has not to harm any user from the start to the end of the cloud chain.

Citizenship. This can be seen as obeying laws and protecting the environment. In the case of the factory for cloud hardware, the cloud provider and the XaaS provider have to think always in the best way, for obeying the laws and protecting the environment of the country where the factory produces.

All of these ethical values can be seen as a cloud of conduct, which not necessary refers to cloud-computing, but also the whole IT-Business. Of course this cloud of conduct can be extended, but it should be seen as a starting point and the pillar of a trust system.

2 Conclusion

This paper explained the different privacy problems, which come up in the context of cloud-computing. The idea is not, to just only take a closer look at the cloud providers and tighten the privacy through a trusted third party, with a certification system. Privacy starts before the cloud providers. The starting point takes place in the factory, where cloud hardware is produced. When this production is secured, a high privacy will also take place in the later chain. It is important to categorise the trust level of cloud providers, to find out if the data needs to be encrypted or not. This encryption can be done by homomorphic encryption. XaaS provider should give the XaaS User these privacy measures. Furthermore, the XaaS User should himself see, without IT-knowledge, if his user privacy is offended. These offences should be seen, for example through a trust system, which contains every piece of privacy concerned, from the factory production, cloud provider, XaaS provider to the XaaS Users web application. One way to build such a trust system, is to bring in a basic value system like a cloud of conduct. This cloud of conduct can be seen as the trust system pillar and further go onwards with this basic system to higher level layers. The pillar for a trust system was defined as a cloud of conduct, which was gained through six universal moral values. More likely to happen, the cloud privacy will get more and more important in this cloud era. With the components of security protocols, encryption, third parties and a cloud of conduct, a cloud privacy trust system gets a first framework. For the outlook of this work, it will be necessary to make a guideline for an implementation of this framework. This means that specific parts need to be prioritised for implementation. Further, test scenarios on privacy matters need to be defined in the context of the cloud chain with e.g. performance and cost.

References

1. Schwartz, M.S.: Universal moral values for corporate codes of ethics. *J. Bus. Eth.* **59**, 27–44 (2005)
2. Jain, P., Gyanchandani, M., Khare, N.: Big data privacy: a technological perspective and review. *J. Big Data* **3**(1), 25 (2016)
3. Acquisto, G.D., Domingo-Ferrer, J., Kikiras, P., Torra, V., de Montjoye, Y.-A., Bourka, A.: Privacy by design in big data - an overview of privacy enhancing technologies in the era of big data analytics. Technical report 1.0, European Union Agency for Network and Information Security, December 2015
4. Itani, W., Kayssi, A., Chehab, A.: Privacy as a service: privacy-aware data storage and processing in cloud computing architectures. In: Eighth IEEE International Conference on Dependable, Autonomic and Secure Computing, pp. 711–716 (2009)
5. Zhang, K., Ni, J., Yang, K., Liang, X., Ren, J., Shen, X.: Security and privacy in smart city applications: challenges and solutions, enabling mobile and wireless technologies for smart cities: part 1. *IEEE Commun. Mag.* **55**, 122–129 (2017)
6. Martinez-Balleste, A., Perez-Martinez, P., Soalanas, A.: The pursuit of citizen' privacy: a privacy-aware smart city is possible. *IEEE Commun. Mag.* **51**(6), 136–141 (2013)
7. Roman, R., Zhou, J., Lopez, J.: On the features and challenges of security and privacy in distributed Internet of Things. *Comput. Netw.* **57**(10), 2266–2279 (2013)
8. Armbrust, M., et al.: Above the clouds: a Berkeley view of cloud computing. Technical report, University of California, Berkeley (2009)
9. Zhou, M., Zhang, R., Xie, W., Qian, W., Zhou, A.: Security and privacy in cloud computing: a survey. In: 2010 Sixth International Conference on Semantics, Knowledge and Grids, Beijing (2010)



Script Based Migration Toolkit for Cloud Computing Architecture in Building Scalable Investment Platforms

Rao Casturi^{1(✉)} and Rajshekhar Sunderraman^{2(✉)}

¹ V.P. Risk Management, Voya Investment Management,
Atlanta, GA 30327, USA

Rao.casturi@voya.com

² Department of Computer Science, Georgia State University,
Atlanta 30062, USA
raj@cs.gsu.edu

Abstract. The 2008 Financial Crisis which created a global financial market meltdown is mainly due to badly structured mortgage loans with poor or subpar credit quality and lack of proper tools to measure portfolio risks by the lenders. Even though several problems led to this crisis, we looked at this from a Big Data. Had the infrastructure and analytical analysis tools were present to the lenders, they would have found the various early warning signs on these mortgage loans and could have been better prepared for the crisis. Aftermath of the crisis, all the big financial institutions took a fresh look and embarked onto build various tools and frameworks to address this Big Data in their portfolios with data driven analysis. The 3Vs (Velocity, Volume and Variety) of the Big Data in our Mortgage Loan Analysis System challenges our traditional approach in collecting, processing and presenting the individual and aggregated loan level data in a meaningful format to facilitate our portfolio managers in decision making. The traditional methods are implemented on a standalone on-premises SQL server. Our Framework creates the foundation of migrating from traditional standalone database architecture (on-premises) to Cloud Computing environment using “Script Based Implementation”. The methods we present are simple but effective and saves resources in terms of Hardware, Software and on-going maintenance costs. Big Data “Capture, Transform, Calculate and Visualize” (CTCV) implementation takes a phased approach rather than a big bang model. Our implementation helps the Big Data Management to be part of organizational tool kit. This saves hard dollars and brings us in line with the overall firm strategic vision of moving to Cloud Computing for Investment Management Services.

Keywords: Big Data · Financial applications · Cloud Computing

1 Introduction

In any investment portfolio management, diversification of the portfolio holdings is critical to achieve client expected returns by minimizing the downside risk in the portfolio. Diversification can be within a specific sector or across sectors or different

types of fixed income bonds. One such investment strategy is to have exposure to the mortgage bonds in the portfolio along with other investment instruments [1]. The total Mortgage Debt Outstanding for 2017Q3 is estimated to be around \$14.7 trillion in U.S. The structure of these mortgage securities are called pools and they consists of mortgage loans taken by individuals. The raw data (factor data) is released on a monthly basis by various Mortgage Agencies like Ginnie Mae, Freddie Mac and Fannie Mae. The individual financial institutions either use third party vendor provided solutions or build their own data gathering applications to collate this information. The data set can be around 2 TB per month and grows from month over month as the loans and pools tend to increase as time progresses. For smaller institutions the cost in implementing third party vendor for Big Data solutions are expensive and can't justify the costs. The other problem is presenting this information in a useful and flexible manner for the portfolio managers. We were able to solve our Big Data problem by breaking it into smaller manageable phases and build a modular based frame work to save organizational costs and reduce maintenance and other infrastructure overheads. Our approach (CTCV Framework) gave our portfolio managers the ability to analysis huge amount of data in a very short period compared to the legacy EXCEL based application. The EXCEL based model was severely limited to the amount of information they can analyze in a given day due to the technical limitations of processing Big Data in EXCEL.

The current paper is organized into eight sections. Section 2 introduces the preliminary definitions and background information. Section 3 highlights the problem, presents related work done on migration of traditional database models to cloud based models in academia, and introduces our proposed solution. In Sect. 4, we present the solution and in Sect. 5 shows the implementation. Section 6 captures the results and Sect. 7 is conclusion with Sect. 8 laying the foundation for our future work.

2 Preliminaries and Background

The Investment Portfolio consists of Fixed Income Bonds [1] and or Equity securities. In the day and age of analytics, the key analytical indicators [2] of these investment assets drive lot of portfolio decisions. Calculating and picking up trends in these analytical measures is a challenge when we need to go through several millions of mortgage loans. Further the Bonds can be classified into corporate and mortgage or asset backed bonds. For our discussion, we will focus on the mortgage backed securities. The Mortgage Backed Securities (MBS) can be further classified into two broad categories by their defining attributes. The MBS security is backed by a pool of securities which can be residential-mortgage backed or commercial-mortgage backed. The Fig. 1 is a very high level of how an investor or an institution can invest into a Mortgage Backed Security (MBS) depending on their risk appetite.

We will discuss the structure of a mortgage pools which is relevant and in-scope for

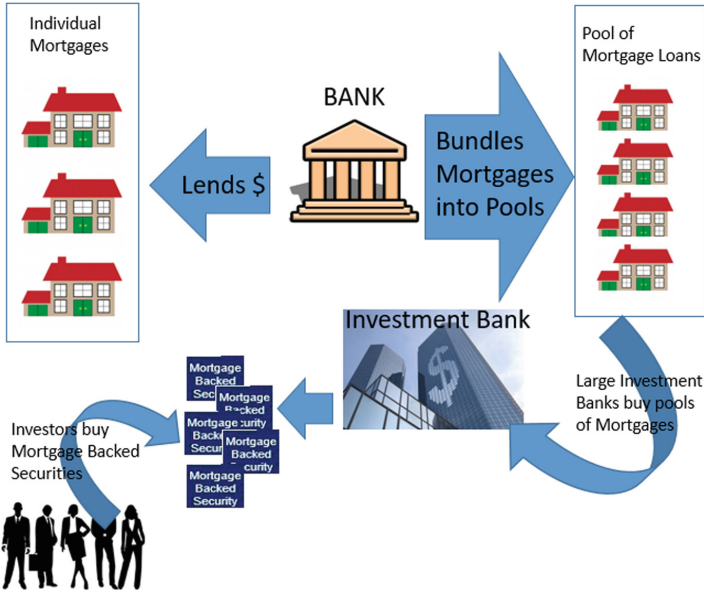


Fig. 1. Mortgage backed securities

this paper. The mortgage is a loan secured by the collateral of some specified real estate property which obliges the borrower to make pre agreed periodic payments. The lender usually banks, may require mortgage insurance to guarantee the fulfillment of the borrower’s obligation. Some of the borrowers can be qualified for mortgage insurance which is guaranteed by one of the three U.S. government agencies, Federal Housing Administration (FHA), the Veteran’s Administration (VA), and the Rural Housing Services (RHS). There are several types of mortgage designs used throughout the industry. A mortgage design is based on specification of interest rate, term of the loan and the manner in which the borrowed money is repaid to the lender. A pool of mortgage loans put together as a single asset is called a mortgage-pass through security. The cash flow of a mortgage pass through security depends on the cash flow of the underlying pool of mortgages. As each loan in a mortgage pass through security can have a different outstanding balance on the loan and different maturity it is customary to use a weighted average analytics for the overall pass through security. Figure 2 is a simple example of a pooled mortgage loan as a pass through security.

Furthermore, the MBS which are sold by Investment banks are divided into various tranches depending on the risk profile of the individual loans. These mortgage backed securities are very complex and the details are out of scope of this paper.

The Weighted Average Coupon (WAC) and Weighted Average Maturity (WAM) for the above mortgage pool is calculated as

Loan	Outstanding mortgage balance	Weight in pool	Mortgage rate	Months remaining
1	\$125,000	22.12%	7.50%	275
2	\$85,000	15.04%	7.20%	260
3	\$175,000	30.97%	7.00%	290
4	\$110,000	19.47%	7.80%	285
5	\$70,000	12.39%	6.90%	270
Total	\$565,000	100.00%	7.28%	279

Fig. 2. MBS pools, loans and their analytics

$$\text{WAC} = 0.2212 \times (7.5\%) + 0.1504 \times (7.2\%) + 0.3097 \times (7.0\%) + 0.1947 \times (7.8\%) + 0.1239 \times (6.90\%) = 7.28\%$$

$$\text{WAM} = 0.2212 \times (275) + 0.1504 \times (260) + 0.3097 \times (290) + 0.1947 \times (285) + 0.1239 \times (270) = 279 \text{ months (rounded)}$$

These are just 2 analytical measures but for a mortgage loan there are several of these measures which are important and which are calculated from the raw data set for investment management. All the different types (VA, FHA, RHS etc.) [1] are important and their % exposure in a loan is critical for portfolio manager's decision. This data is huge and some of the pools can be of thousands of loans. There are several calculations done on various dimensions (range bound). E.g. Showing the WAC over 0–2, 2–5, 5–7, 7–10 and above 10 may be one set of range and the other can be 0–3, 3–5, 5–7, 7–10 and above 10. Another example of a measure is Conditional Prepayment Rate (CPR). CPR is a sub calculation on Single Month Mortality Rate (SMMR) which is calculated for each month of the mortgage and use that SMMR to calculate the CPR for a specific month. The Eq. (2) shows the calculation for one month CPR. SMM is calculated with as shown in Eq. (1). Some of the mortgage calculations are recursive in nature.

$$\text{SMM}(t) = \text{Prepayment in month } (t) / (\text{beginning mortgage balance for month } (t) - \text{scheduled principal payment in month } (t)) \quad (1)$$

$$\text{CPR}(1) = 1 - (1 - \text{SMM}(1))^{12} \quad (2)$$

In U.S., there are three major types of passthrough securities are guaranteed by agencies created by Congress. This is done to increase the supply of capital to the residential mortgage market. Those agencies are the Government National Mortgage Association (Ginnie Mae- GN), the Federal Home Loan Mortgage Corporate (Freddie Mac- FH) and the Federal National Mortgage Association (Fannie Mae- FN).

The current paper is not focused on the solution which is provide to eliminate the EXCEL based solution but focus on scalability and flexibility by migrating the solution it to Cloud Computing environment. The migration of our existing SQL Solution

(replacing EXCEL based application) to Cloud Computing will give us savings in infrastructure, software costs. By solving the space constraints on data storage brings trend analysis in decision making [6].

3 Problem Statement, Related Work and Proposed Solution

The problem of having a flexible and scalable mortgage analytics from the huge data set by using EXCEL as calculation tool is very challenging and prone to three major issues. The EXCEL solution Extract Load and Transform (ETL) the raw agency mortgage files is not sustainable or practical and is very inefficient for decision making. Each file can contain millions of tuples or rows. The second issue of EXCEL as a calculation tool limits the ability of flexibility for any new metric calculations which are frequently needed on either ad-hoc basis or on a permanent basis. The modifications of EXCEL macros to accommodate any new calculations is difficult in our current version of EXCEL due to the complexity of code and lack of any version control or testing environment, opening a huge Operational Risk for the organization. The third constraint is the storage of historical data for trend analysis or data mining. The issues we have in our EXCEL version can be categorized into three main problem segments. (1) ETL is only possible for one deal at a time and is time consuming. (2) User defined calculations are not possible. (3) Trend and Data Mining capability is not available. Solving these three issues will increase the productivity of our investment teams giving more time for analysis rather than working on data collection and code manipulation.

To address the above mentioned EXCEL solution issues, our research focused on the existing academic and industry research on these issues of huge data stores and ETL tools to address our problem (1) and flexible calculation frameworks and data mining tools to address (2, 3) issues. There is a lot of academic research in terms of huge data store/ETL [16] and data mining tools.

The area we found is challenging is the flexible user defined calculations and running it on a distributed and on elastic data lakes and migration of SQL databases to cloud environment [10]. The mortgage pools are made up of individual loans. The data size of these pools can be viewed as Big Data. The main characteristics of Big Data is defined by the main three pillars by which we categorize the underlying data set. They are called 3Vs. [7].

The Fig. 3 shows 3Vs of Big Data in our study are Volume (Terabytes of data) in raw format with the Variety (Mixed data values) and Velocity (Changing daily) which makes it a best candidate for our study and implement our method of Capture, Transform, Calculate and Visualize (CTCV) to build a framework which can be leveraged for other investment data related decision systems.

The volume of the data is the number of tuples and attributes we have for each underlying pool. The distinct values or the domain values of each individual attribute can be a vast range of values which can be classified as variety of data set and the data changes from time to time and for our study we take a month over month change to

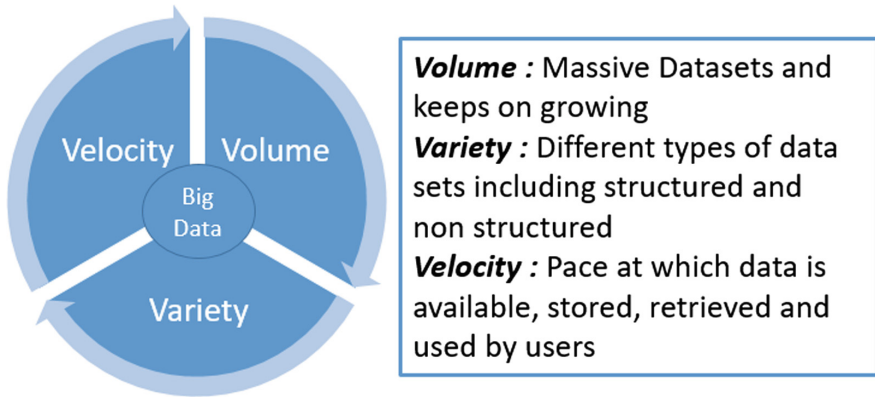


Fig. 3. Big Data categorization diagram

show the velocity is frequency and in our case it is monthly. There are several academic research papers on the Big Data and how we can leverage the existing technologies to source and store the data in a meaningful way. Big data analysis systems are very important to organizations because it enables them to collate, store, manage, and transform vast amounts data at the given instance, in a dynamically changing environment to gain the meaningful insights. The evolution of data management and knowledge systems [18] grow as necessity than nice to have. The initial sources are our raw data coming from various sub data base systems, flat files or any other source of information. In early 1970s the advent of relational data base systems (RDBMS) took the data store, retrieval to another level moving the needle from file based systems to relational set dependent systems. Codd's [14] paper on relational database design changed the way we process the data.

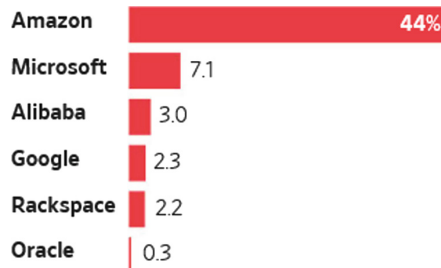
In academia as well as in industry there is a major research work carried out in terms of supporting large files coming from various source systems. During 2002 and 2003 Google came up with their proprietary file system Google File System (GFS) [4] which led the way to several other research groups to come up with their architecture to support large file systems. Google also published their MapReduce [5] programming model which can process terabytes of data on thousands of machines. MapReduce program was distributed over large clusters of commodity machines. During this time, Apache open-source developed Hadoop architecture and called it as Hadoop [17] Distributed File Systems (HDFS) and introduced their MapReduce programming model. The MapReduce architecture uses a map function specified by users that processes key-value pair to generate a set of intermediate key-value pairs, and a reduce function that merges all the intermediate values with the intermediate key [4]. Even though this is out of scope for our current paper, we are laying the architectural foundation for future work on processing our large data sets of mortgage pool information on a Cloud Computing environment using these distributed techniques. We keep our implementation open for our future needs.

As part of our research we evaluated several industry products in the Big Data and Cloud Computing space shown in the Fig. 4 including Oracle Cloud Platform,

Microsoft Cloud Technology (AZURE) and Amazon Web Services (AWS). All these vendors offer various cloud platform products. The best suited for our needs is Microsoft platform as our **Phase I** (Elimination of EXCEL based solution) *proposed and implemented solution is on a Standalone Microsoft SQL server*. This gives us a head start and saves resource time in learning other technology platforms which could be impact our business processes. Phase I used a lot of advanced normalization techniques [11] to improve the retrieval of information faster than traditional frontend driven controls in EXCEL. We utilized the RDBM concepts and techniques which helped us in solving several key issues in building BI dashboard [12] replacing EXCEL Solution. The other widely used Key-Value pair indexing [13] technique is used to build a distinct values list which serves the purpose of saving hard disk space and enables us to use lot of SQL In-Memory operations. Compressed Index Architecture of Microsoft did proved support in reducing our space constraints [3]. The proposed solution of migrating the standalone SQL Database solution to Azure Cloud Computing environment will provide the next stage for our “Financial Calculation Framework on Cloud Computing Environment”.

As part of our discovery phase (research) we evaluated several cloud technologies provided by vendors.

The Fig. 4 shows various vendors we evaluated in search of a suitable vendor for our solution. The Microsoft Azure Cloud platform is open and flexible cloud platform providing ability to build, deploy, manage applications across global network with various tools and programing language support to enhance an organizations ability to



Source: Gartner

Fig. 4. Industry players in cloud computing space

grow with technology outsource to minimize costs and maximize profits by focusing on the investment performance to grow in market place. With in-house expertise we decided to migrate our standalone on-premises phase I solution to Microsoft Azure Cloud Computing Environment.

The “pay as you use” model of Cloud Services are key for the flexibility and expandability of any organization’s growth. The main architectural components of any Cloud Service [8, 19] provider can be classified into three major categories. They are **SaaS** (Software as a Service) is centrally hosted and managed for the end customer and

usually based on a multitenant architecture. Example for SaaS can be Office 365 or Google Documents on Cloud. **PaaS** (Platform as a Service) is another service provided by the cloud service providers. In this architecture the customer will provide the application and the Cloud provider will provide the platform. **IaaS** (Infrastructure as a Service) is provided by vendor by running and managing server farms, running virtualization software, enabling the customer organization to create VMs (Virtual Machines) that run vendor's infrastructure.

4 Proposed Solution

After going through the pros and cons of Cloud Computing and the flexibility of the services provided by various Cloud vendors, we decided on using Microsoft Azure Cloud as our platform. We propose the solution in two phases. **Phase I** is building a standalone SQL DB solution to implement ETL and flexible calculation engine to calculate various analytical measures on our Big Data. This is not our focus of our current paper. The **Phase II** which is migration of the phase I solution is our focus.

The proposed Phase II or CTCV solution at high level, will take the existing installation of standalone in-house or on-premises solution and port it to Cloud Computing platform with minimal disruption to our business processes. The Fig. 5 gives a high level process flow of the events of our proposed implementation of CTCV.

Phase II – Migration of Standalone SQL solution to Cloud Computing

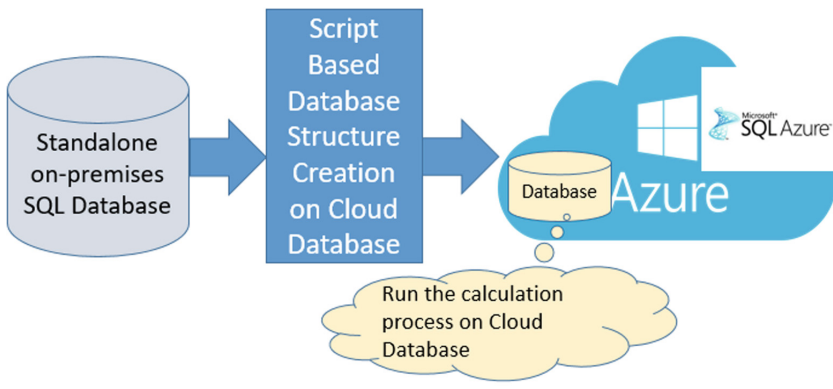


Fig. 5. Proposed solution architecture

Our preferred method is to take the SQL DB Script to build and implement the Cloud Instance rather than restoring the standalone database backup. Our approach of “Script Based” will have initial benefit of going through the objects where there will be changes needed to fit into the Cloud Computing framework. This shortens the implementation time and produces a cleaner and leaner version of the database schema.

With our script based framework it makes it more manageable by less sophisticated SQL users to migrate. The Other method of migrating the existing standalone databases with the backup sets which can create several issues (user access, roles, Active Directory groups (AD Groups) etc.) which a non-technical user can't be able to handle. In a "Backup Based Model" the testing of end-to-end may not be done till later phase. In a script based migration model we were able to test and modify the components needed as we move from one object group to another object group.

Our solution is easy to implement by any team or individual with some knowledge of SQL. With the simple steps which we propose will lead the organization to migrate to a more flexible and scalable data base solution in an organization. The flexibility and scalability comes from the fundamental design of a cloud architecture of "elastic" nature of the Cloud Computing services we discussed in our Cloud Architecture section (Related Work). The elastic nature of the Cloud Database Solutions enables our existing database to grown as the Bid Data grows as time passes. In the next section we will layout the actual implementation of our Mortgage Loan data with a sub set of information as part of our Phased implementation setup.

5 Implementation of Proposed Solution

Our proposed solution of migration of our Phase I solution (Standalone SQL Database) to Cloud Computing environment is done in two parts. Pre-migration and Post-migration. These two will set stage for a better and stable migration process.

Pre-migration: The first step is to verify the subscription of the Microsoft Azure Account with the institutional license management team and if not we need to obtain the needed subscription. Usually depending on the usage the organization purchases the subscription level. Once we have the proper license set up, check the access to the Azure portal (<http://portal.azure.com>). In the Fig. 6 we show the general layout of the Azure Portal which can be maintained by the subscription we have.

Our implementation is on a Windows Azure MSDN – Visual Studio Premium which gives us the ability to build elastic database servers with a 250 GB space. This is a called Standard S3: 100 Database Transaction Units (DTUs). We are using this space to prove the proposed architecture solution will work and can be implemented to bigger database needs. Our test data is for a sub set of Mortgage Agency Pools. The data set we have is for Agency FN and currently we have 12 raw files with 14 million total rows.

Proposed Implementation Steps: To implement our solution of Script Based, we first took the current implementation database (on-premises) schema script. These DDL scripts are saved as "Object Categories". E.g. For Tables, we called as AZURE – Table Script and for the Store Procedures are called as AZURE – Procedures. It is up to the individual team to decide the naming convention. This step gives the implementation team a change to modify any non-cloud scripts or rework the non-cloud scripts before implementing on the Cloud instance. One example of "script modification" is removing any cross-database-queries. Next step is to create a proper database server and database on the Azure Cloud. This can be done with simple steps of by going

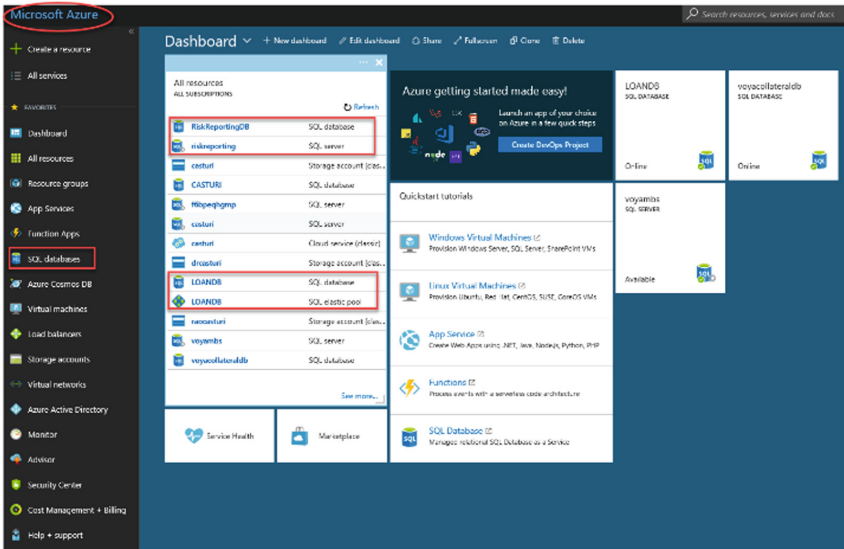


Fig. 6. AZURE set up of our MBS cloud database

through the Microsoft Documentation [9]. Once we have the Azure Cloud SQL Server and Database, the next step is to run the DDL scripts which we saved as “Object Category Scripts” our current implementation. If the SQL DDLs are standard, then we should not see any issues. We did encounter few issues which are rectified during the migration. The Azure SQL code won’t let us reference the <database>.<owner>.<object>. The <database or Schema name> is not permitted in the Azure syntax of referencing the database objects. This will pose an issue with migration if we have linked servers or queries going across multiple databases. The newer version of Azure SQL has a fix.

Create an “External Data Source” as shown in the Fig. 7 uses the architecture in the Azure SQL Database under the External Resources and use that in DML when writing the standard query referencing the remote SQL table of SQL Server instance in

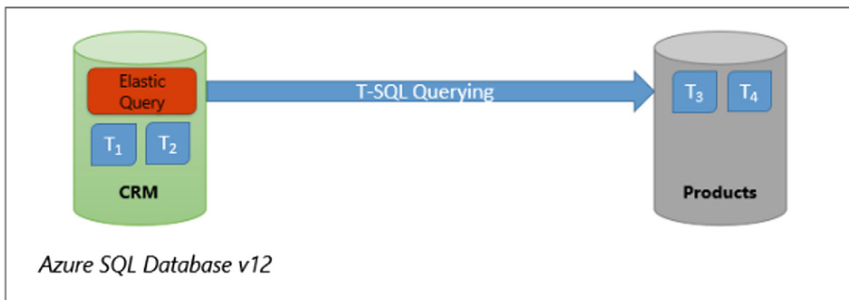


Fig. 7. Cross query set up schema.

Azure SQL installation. The Figs. 8 and 9 show the External Data Source Set up and how to reference and execute the cross database queries if needed.

Post Migration: Till this point of implementation, we used the Azure portal. Now it is time for connecting to the Azure Cloud via other client tools. Usually in the organi-

```
CREATE EXTERNAL DATA SOURCE RemoteReferenceData
WITH
(
    TYPE=RDBMS,
    LOCATION='myserver.database.windows.net',
    DATABASE_NAME='ReferenceData',
    CREDENTIAL= SqlUser
);
```

Fig. 8. Creating external data source in Azure [9]

```
CREATE EXTERNAL TABLE [dbo].[zipcode](
    [zc_id] int NOT NULL,
    [zc_cityname] nvarchar(256) NULL,
    [zc_zipcode] nvarchar(20) NOT NULL,
    [zc_country] nvarchar(5) NOT NULL
)
WITH
(
    DATA_SOURCE = RemoteReferenceData
);
```

Fig. 9. Data source concept in Azure SQL database [9]

zation, the data base professional (Database Administrators, Developers, Designers) use several client tools to connect to the SQL Database instances to do their regular tasks. The Azure SQL is no different and we would like to show the easy way to

connect to the already created Azure Cloud SQL database. For our project we used the Microsoft SQL Server Management Studio (MSSQLMS) as our client tool to connect to the Azure Database instance. While creation of the Azure SQL Server Azure will create a unique server name to reference the instance. The Fig. 10 shows the Azure SQL Server and also the Azure SQL Database with the names we provided for referencing them through our implementation. This is a big step as creating the database on Azure depending on the subscription level and we want to make sure we have enough service level contract in place to proceed.

Once the database schema is set up with the needed external sources, we then populate the mapping table data needed but importing the data from files or current

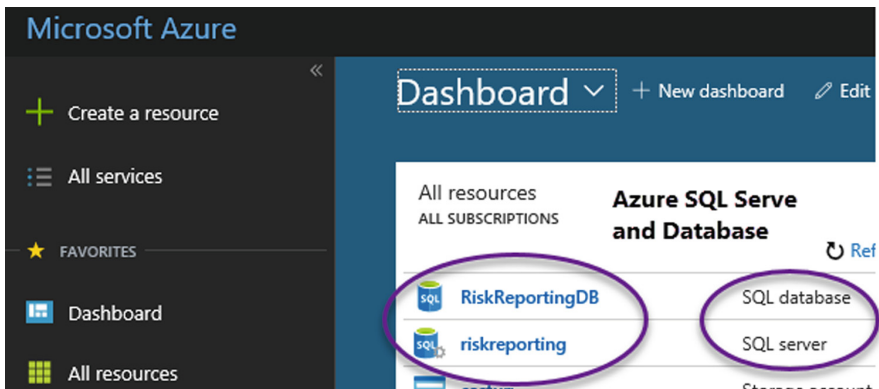


Fig. 10. Azure data dashboard

standalone SQL Database. For our instance we named our SQL Server as “riskreporting” and the instance can be referenced as “riskreporting.windows.net”.

Now to connect to the Server and to access the database there is one more step involved which is done through the Azure Portal. This is called setting up the client machine firewall to access the database server. The Fig. 11 shows the Firewall setting.

Usually if we are going to implement the Active Directory Group (AD Group) access they can be done through centralized IT security function but for our proposal we are maintaining the access control. The other reason is to have data secured and protected from other non-access individuals even though they are on our AD Group. Once the firewall IP address is added to the Azure SQL data server we can now connect to the database via the MSSQLMS. Figure 12 shows the screen where we give the server credentials, in our case “riskreporting.database.windows.net” and the log in credentials which were created during the process of setting up the Azure SQL Server and database.

We tested both (elastic, non-elastic databases) the installs with our script based approach to make sure we test them as part of our solution. We can now connect to the

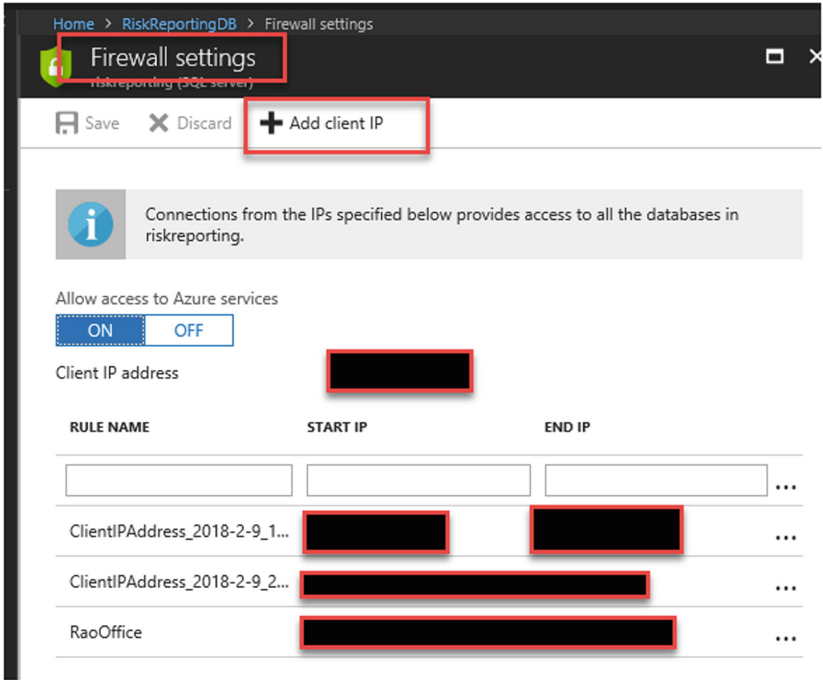


Fig. 11. The Azure firewall setting to allow the client access

Azure Cloud Database using the client tools by referencing the database name **riskreporting.database.windows.net**. This is shown in the Fig. 12 with the users credentials to be passed to have a the Microsoft SQL Query Editor to show the objects and the query panel to be available for the users.

Our Azure Database instance is RiskReportDB. The Fig. 12 shows the connection to the Azure SQL Server and the database on the server to which we have access. We set up 2 resource pools one with elastic (voyacollateraldb) and one without elastic database (RiskReportDB). This set up is to verify the elastic nature of the database which we need for our implementation to test the multiple month storage which solved our historical nature of the data store.

Database Transaction Unit (DTU) is a key measure for a single Azure SQL database at a specific performance level with a service tier. DTU is a blended measure of CPU, Memory, I/O (data and transaction log I/O). Once thing to note is the resources used by our workload do not impact the resources available to other SQL databases in the Azure Cloud and the resources used by other workloads do not impact the resources available to our SQL database. The DTU is a bounding box with CPU, IO and Memory as boundaries. The Fig. 13 shows the DTU box.

Once we have the database connection ready and the objects ready on Azure Database, we now can load the data via SSIS or any other standard ETL data load

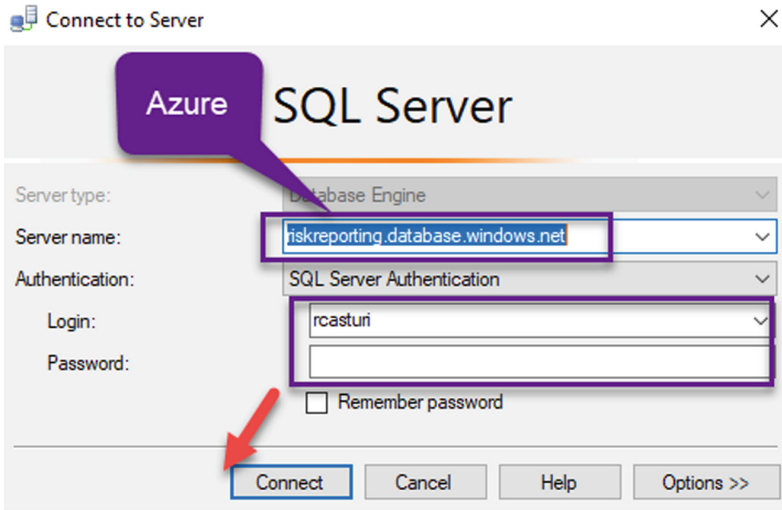


Fig. 12. Azure database connection via SQL client

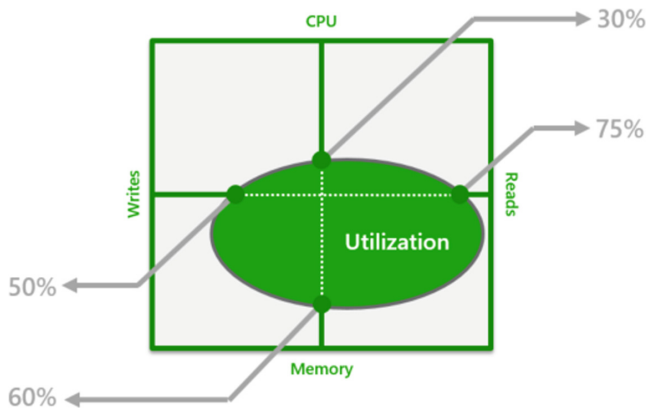


Fig. 13. Monitoring Azure database workload utilization within bounding box

methods. For our testing, we used the DTS Wizard to load the data for the users set up tables and for the factor data. With this step we are ready with all the needed base data set for the actual analytical calculations to be performed on our newly created Azure Database. We can run the calculation engine via client tool or via the Azure Portal. To run any SQL Queries on Azure portal we use the portal query panel provided by Azure SQL data server.

6 Results

We ran our user defined Analytical Aggregation Rules on the newly set up Azure SQL Cloud Computing environment. There were no errors. The final calculations were verified with the standalone SQL Database implementation which is our Phase I production. The verification is done by comparing the results from on-premises version to the Azure SQL Version. The approach is to take the calculations of one deal and compare it with the results produced by Azure implemented solution's calculation numbers. There are no deviations or errors in our newly implemented Azure SQL implementation (Phase II). The subscription we currently have is a very close match to the on-premises so we did not see the any significant improvement in performance. As we noted the higher subscription for more DTUs the better the performance. This is an advantage of utilizing Azure Cloud Computing services as we can increase our DTU subscription depending on our necessity. These are conservative estimates to give us ample room for migration and re-work if needed. As shown in the Fig. 14 we can see the spikes in the DTU usage which indicates the CPU on the Azure Server is active and processes the users requests.

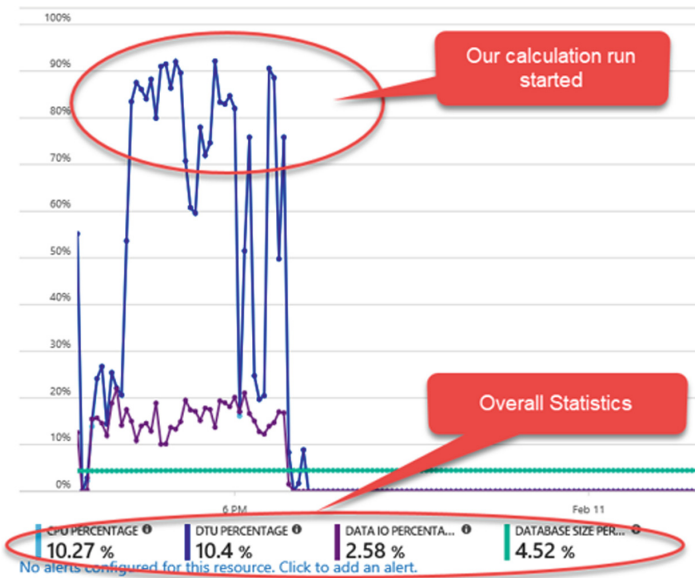


Fig. 14. Cloud computing calculation time chart

Another observation we did depending on the rating category by services without any business disruption. Here we applied ratings for several categories which are important for our organization. We recorded the platform (On-premises, Azure) suitable for our organization for future needs and plotted the rating of each category and assigned an appropriate rating between 0 and 5. The results of the rating based scale

will be discussed in our next section of conclusion as part of the comparison of the on-premises and Azure Cloud Computing Architecture. Figure 15 shows the plot by category and rating for each category.

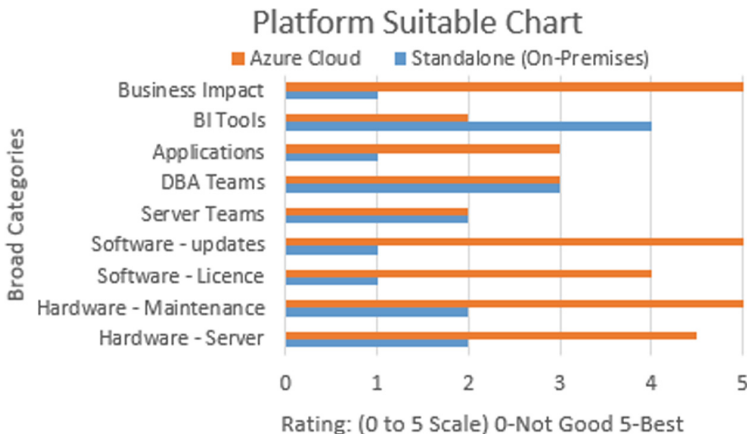


Fig. 15. Comparison (In-premises vs cloud architecture)

The higher the ranking the it is better suited product for our usage. As an example, if we take “Business Impact” category and compare the score of 5 for Azure vs. 1 for Standalone it shows that we can implement Azure solution won’t impact business as it scores 5 whereas the standalone solution can impact business if there is any migration requests are made to upgrade or add additional infrastructure resources. Whereas in BI tool category, the standalone instance has more flexibility and less expensive compared to Azure solution. This gap can be narrowed as more research and reporting frameworks migrate to Cloud Computing environment in the future.

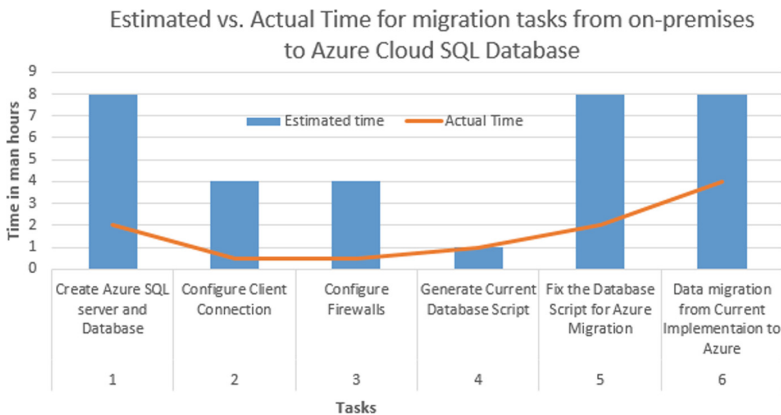


Fig. 16. Time chart (Expected vs actual implementation)

7 Conclusion

The results of migration from a standalone or on-premises SQL Database to Cloud was a successful and this is due to the preparation of the pre-implementation preparation we took before actually implementing the DDLs on Azure SQL platform. The initial estimates of our migration was very conservative giving us enough room to implement any work around for migration scripts to Azure SQL database. We plotted the initial estimated time for each task and the actual time taken for the task to implement. The graph in Fig. 16 shows the solid bars are estimated and the line showing actual time we took to implement. The shorter times for actual implementation can be largely contributed to the preparation and project planning going through various scenarios and studying several case studies which are available in industry and also on Microsoft's knowledge base.

This actually helped a lot in preparing the perfect "scripts" to upload and run on Azure SQL Database. Usually the rework is in making the scripts run without errors and that we were able to accomplish early in our implementation phase. We called this as pre-implementation task.

The Infrastructure as a Service (IaaS) [10] architecture promised by Microsoft is actually worked well for us. Due to the planning and execution the implementation by script based approach, we were able to achieve building the foundation of our migration tool kit to manage Big Data Management. The growth of asset under management AUM depends on the flexible and scalable enterprise architecture and we believe Microsoft Azure Cloud Computing platform solutions will enable to travel on a growth trajectory. The contribution of this paper is not just for any investment company but can apply to any data driven business in industry or in academia.

With all the above mentioned positive outcomes, we conclude the migration of standalone SQL database to Azure Cloud Computing environment will increase our investment portfolio manager's productivity, cut down the infrastructure costs of buying hardware (servers) and maintaining the software patches, building performance tuning tools in-house. The hard dollar saving can be put to work and can show profitability. The overall business goal of flexibility, scalability is accomplished by migration to cloud computational model. The growth of asset under management AUM depends on the flexible and scalable enterprise architecture and we believe Microsoft Azure Cloud Computing platform solutions will enable to travel on a growth trajectory.

During our migration process we noted there are few things we can share with other researchers who are in process of migrating from on-premises databases to cloud computing environment. If an organization decides to utilize the Cloud Computing Platform, there are few things should happen before they can start doing any production migrations. The first and foremost is to justify the need for cloud computing. This won't be suitable for a small company with fewer resources and there no requirement for computational needs to support the product or functions. If the organization's data can be managed and maintained on one or 2 SQL servers with one or two Database Administrators (DBA's) then cloud solution may not be an optimal solution. Building the infrastructure around the cloud will defeat the purpose of simplification and cost saving for the organization if the data is contained and there is no need for future

growth. The second main point we recommend is to evaluate the subscription services provided by the Cloud vendors. There are several options from which an organization can select the products depending on the need seen by the business. It is very important to be able to assess the need as building a solution in a lower DTU subscription and moving it to a higher DTU is a bit challenging. This can impact overall enterprise contract negotiations which can delay or impact the cost of project. The third point is on deciding elastic database architecture. Having the ability to grow and be able to utilize multiple databases in cloud is better than having multiple standalone database servers and link them through a distributed database architecture. The other options we would like to put it out for the researchers are Microsoft Azure Data Lakes and Data Warehouses. These can be used and enhance the company's productivity and reduce the dependency on standalone on-premises infrastructure saving cost (time and resources) in future for the organization.

The contribution of this paper is not just for any investment company but can apply to any data driven business in industry or in academia. The tool kit we proposed which includes the script based implementation framework along with the set-up of the client and development tools will make any migration seamless and very little downtime for an organization.

8 Future Work

As we mentioned earlier in the paper, this project implementation is a multiphase implementation designing each phase to be the launch pad for the next phase. From EXCEL version to the Standalone SQL Solution itself is a huge project which included flexible user calculations. During that design phase we did keep our next phase of implementing the standalone solution on Cloud Computing environment as that technology is evolving and we do have need to embrace that technology as the hardware, software cost is getting higher for a standalone SQL Solutions. With the implementation of the migration project, the future is to build the analytical trending models on the Cloud Computing platform to give our portfolio manager the cutting edge technology for the day to day investment decisions.

Our implementation added value to our organization and paved the path for other teams to migrate from standalone database instances to go in to next stage of Cloud Computing. Our paper and our successfully implementation demonstrated that script based implementation is worthwhile undertaking in migrating several of our on-premises databases to Azure Cloud Computing environment. The Microsoft tools [19] we mentioned and proposed will help mid-sized company to migrate their existing on-premises databases to cloud based architecture using the simple procedures mentioned in our paper. The methods are to be applied to any database driven applications in any organization. The solution is flexible and scalable and in future we are trying to make it a standard migration tool for our organization. The tool kit we proposed which includes the script based implementation framework along with the set-up of the client and development tools will make any migration seamless and very little downtime for an organization.

The other important area of our future research is on security and privacy [20] of the Cloud Computing Architecture. This currently is a concern to many financial organizations. The main areas of focus will be on the rules and controls around “Proprietary and Intellectual Property” and cyber intrusion of company financial data. We are currently undertaking this research and will bring out the current issues and possible solutions in building a framework for any financial institution to be able to utilize the cloud computing environment.

The other area we are focusing our research is in implement a full stack BI and Data Mining [15] framework on our Azure Cloud Computing Solution.

References

1. Fabozzi, F.: Fixed Income Analysis. CFA Institute Investment Series, 2nd edn. Wiley, Hoboken (2007)
2. Fabozzi, F.: The Hand book of Fixed Income Securities, 7th edn. McGraw-Hill, New York (2005)
3. Ozur, M., Tuna, H., Coffin, C., Sampaio, T.: Azure Virtual Datacenter. Microsoft, November 2017
4. Ghemawat, S., Gobioff, H., Leug, S.-T.: The Google file system. In: SOSP 2003, Bolton Landing, New York, USA, 19–22 October 2003
5. Gemayel, N.: Analyzing Google file system and hadoop distributed file system. Research Journal of Information Technology **8**(3), 66–74 (2016)
6. Weber, N.: Big Data: can we prevent the next financial crisis? King’s College London Economics & Finance Society (EFS)
7. Eaton, C., Deroos, D., Deutsch, T., Lapis, G., Zikopoulos, P.: Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data. McGraw-Hill, New York (2011). ISBN 978-0-07-179053-6
8. Microsoft Azure Documentation: Overview of Azure Data Lake Store, Microsoft
9. Microsoft Azure Documentation: Create an Azure SQL Database in Azure portal and Linked Databases Microsoft
10. Microsoft Azure Documentation: Load data from flat files into Azure SQL database. Microsoft
11. Elmasri, R., Navathe, S.: Fundamentals of Data Base Systems, 7th edn. Pearson, London (2015)
12. Esling, P., Agon, C.: Time series data mining. ACM Comput. Surv. **45**, Article no. 12 (2012)
13. Zaharia, M., Wendell, P., Konwinski, A., Karau, H.: Working with key/value Pairs. In: Learning Spark. O’Reilly Media, Inc., February 2015
14. Codd, E.F.: A relational model of data for large shared data banks. Commun. ACM **13**(6), 377–387 (1970)
15. Grossman, R., Gu, Y.: Data mining using high performance data clouds: experimental studies using sector and sphere. In: AMC KDD 2008, Las Vegas, Nevada, USA, 24–27 August 2008
16. Apache Hadoop. <http://hadoop.apache.org/>
17. White, T.: Hadoop The Definitive Guide, 3rd edn. O’Reilly Media Inc, Sebastopol (2012)
18. Han, J., Kamber, M., Pei, J.: Data Mining Concepts and Techniques, 3rd edn. Morgan Kaufmann, Burlington (2012)
19. Microsoft Press: Cloud Application Architecture Guide
20. Sobati Moghadam, S., Darmont, J., Gavin, G.: Enforcing privacy in cloud databases. In: Bellatreche, L., Chakravarthy, S. (eds.) DaWaK 2017. LNCS, vol. 10440, pp. 53–73. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-64283-3_5



Space-Adaptive and Workload-Aware Replication and Partitioning for Distributed RDF Triple Stores

Ahmed Al-Ghezi^(✉) and Lena Wiese

Institute of Computer Science, University of Göttingen, Göttingen, Germany
{ahmed.al-ghezi,wiese}@cs.uni-goettingen.de

Abstract. The efficient distributed processing of big RDF graphs requires typically decreasing the communication cost over the network. This requires on the storage level both a careful partitioning (in order to keep the queried data in the same machine), and a careful data replication strategy (in order to enhance the probability of a query finding the required data locally). Analyzing the collected workload trend can provide a base to highlight the more important parts of the data set that are expected to be targeted by future queries. However, the outcome of such analysis is highly affected by the type and diversity of the collected workload and its correlation with the used application. In addition, the replication type and size are limited by the amount of available storage space. Both of the two main factors, workload quality and storage space, are very dynamic on practical system. In this work we present our adaptable partitioning and replication approach for a distributed RDF triples store. The approach enables the storage layer to adapt with the available size of storage space and with the available quality of workload aiming to give the most optimized performance under these variables.

1 Introduction

The sources that produce web-data are rapidly growing everyday, by the increase in the digital life applications. The requirements of analysis and querying of these accumulated data are moving in the same trend. The Resource Description Framework (RDF) is widely used to represent the pieces of data that describe the things in the web and their relationships. RDF data can be seen as a set of triples in the form (Subject, Predicate, Object). A set of triples forms naturally one graph, such that each triple in the data set is modeled as a directed edge which points from a subject to an object and is labeled with a predicate. This featured structure of RDF makes it suitable for representing big web data, which could be big enough to hit the limits of central systems resources, and provide obvious motivation towards distributed storing and processing. Many recent work in this decade targeted the distributed processing of RDF data, focusing mainly to provide more distributed storage space. Unfortunately, partitioning of big RDF data sets is not a trivial task since it can dramatically affect the efficiency of

distributed query processing. SPARQL is the standard language used to query RDF data. Each SPARQL query could be represented as a graph on its own with some vertices and/or edges represented as variables. The query execution is then the process of finding all the sub-graphs in the RDF-graph that match the query graph. However, if a query can find only a part of the graph locally, it has to collect all missing parts from other hosts. The latter process is usually a costly operation in typical networked hosts. To deal with this problem two main directions are usually followed:

- better partitioning strategy, and
- replication of selected important triples across hosts.

The analysis of a given workload could identify recognizable trends, and highlight the parts of the data set that have higher probability to be targeted together by future queries [9]. However, in practical systems it might be difficult to have an initial workload at system start-up, besides that the quality of any available workload in term of its recognizable trends is not assured, as it behaves practically as a variable on its own.

On the other hand, supporting the partitioning by replications increases the chance of a query finds the missing local graph’s parts in the replicas [5]. However, the replication requires more storage space which is a very precious resource in a distributed system that is required to handle big data, and especially with RDF processing systems where the space is highly required to build more indices to boost the performance [7]. The replication approaches which are designed to save disk space can’t directly give the full performance in case the system has plenty of storage space. The amount of available storage space is considered a variable in practical systems since it is directly related to the data set size, and the size of other data container in the system like indices and statistics tables. In this context, the partitioning and replication strategies that are designed to perform well under certain assumptions about the storage space and workload might not show good performance in different practical environments.

In this work, we present our partitioning and replication approach that is superior to related approaches with its adaption to any given workload quality *and* to the availability of storage space. We differentiate between two types of replications: full and compressed. The two types have different sizes and benefits. Our system optimizes the decision of making full, compressed or no replication on the level of a single triple, by comparing derived cost and benefit values highlighted by the workload and the availability of storage space.

We extend our prior work [1] by providing a solid experimental evaluation with respect to other related systems, besides the details of compressed replication and the triples workload engagement. The rest of this paper is structured as follows: We first review the related work in Sect. 2. We then explain our system’s start-up behaviour in Sect. 3. Section 4 describes our approach to analyze the workload and derive numerical values for the triples engagement with the workload. Section 5 describes the basics of the system’s adaptability to storage space. We show the practical evaluation of our approach in Sect. 6, and finally we conclude in Sect. 7.

2 Related Work

The problem of graph partitioning has been a topic of many recent works including works which considered the problem of RDF graph partitioning. Huang [5], WARP [4], and TriAD [3] used METIS as a baseline tool to statically partition the RDF graph. TriAD assigns the METIS partitions to hosts based on a hash function. Other works focused on the semantic found in the RDF data set; a recent work by Xu [11] clusters the classes of RDF data set by applying a page-rank based algorithm prior to partitioning. EAGRE [12] recognizes specific entities and group all the entities that belong to the same RDF classes. The work in [10] considers path partitioning by finding closed paths within the RDF graph, and assigns paths sharing merged vertices to the same partition. Margo et al. [6] used another semantic-based concept which is edge partitioning; they generate an elementary tree from the RDF graph before partitioning it while trying to keep the general objective of reducing communication cost. Multiple works considered the workload when performing partitioning, however WARP [4] and Partout [2] had more distinguished approaches. Partout horizontally partitions the RDF data set into fragments inspired by the horizontal fragmentation of a classical relational table. It first normalizes and transfers the workload into a global queries graph preserving the connectivity and load impact information. The graph is used to generate fragments of matching triples from the data set. However, the result of Partout’s partitioning is highly depending on the quality of the available workload in terms of its queries’ frequency of appearance and on how it is fairly representing all parts of the data set. WARP [4] starts initially by statically partitioning the RDF graph by METIS, then uses a given workload to identify some important border triples to replicate. However, the approach treats all the workload queries whose frequencies are higher than a fixed threshold equally; this might give weak performance if the workload is varying and the storage space available for replication is limited.

A recent work of [8] identifies the properties which are queried together in a given workload, although that is embedded in the work of Partout [2]. Peng [9] scans the workload looking for frequent patterns, measured by the frequency of appearance. A fragmentation phase takes place by matching the frequent patterns with the data set. However, the concept used to generate the frequent patterns beside the workload normalization is still very similar to the min-terms used in Partout and WARP.

3 Initial Partitioning

At the time of system start-up, we typically have no assumption about the workload or the data set. In order for the distributed triple store to start working and collecting workload, it needs to initially partition the input data set without relying on the workload. The well tested METIS partitioner represents a very attractive choice at this stage, and it plays such role in systems like WARP [4]. METIS partitions the RDF graph into n partitions such that we have an approximately minimum number of edges in the cut. Each of the n partitions is assigned

to one host of the n hosts in the cluster. In order to execute a query, it should be sent to all working hosts. Each host tries to find all the sub-graphs that match the input query. However, we may pay communication cost whenever a query is long enough such that two or more partitions hold part of its complete results.

4 Workload Adaptability

After performing initial partitioning, the next step is to support it with replications of selected data. The objective is to increase the chance of local query execution and avoid paying any extra communication cost. The operation is limited by the availability of storage space. Thus, a wise decision about the specific data to replicate is very important. We describe in this section, our approach to analyze the workload, build a global queries graph, then measure the engagement level of the data-set triples with the workload. We will use this engagement level to systematically optimize the replication decision in Sect. 5.2.

Global Queries Graph

Instead of treating all the border triples equally, we make use of a collected workload to measure the importance of the border triples to the future queries, and group them upon their importance into fragments. Consider the sample workload in our running example Fig. 1. We first replace all of the variables, and non-frequent items in the workload (excluding properties) by a single variable Ω . This normalizes the workload and avoids the generation of irrelevant small fragments. The item is considered non-frequent if its frequency is less than a normalization threshold. The normalization prunes the first statistical quartile of the items in ascending order by their total frequencies; in the example, we prune “Kim” and “:person” and keep the other 5 item. We then convert the normalized query workload into a global queries graph G_p as shown in Fig. 2. Each vertex in the graph models a normalized triple pattern. There is an edge between any two normalized triple patterns when they fall in one or more queries in the workload, while the respective edge’s weight is the summation of those queries’ frequencies. The projection of this graph on the data set gives the necessary information about the engagement levels of the data set triples with the workload. Each vertex in the global queries graph represents a fragment of matching triples in the data set. The concept of the global queries graph is used in Partout [2]. However, since we are interested in replicating the triples that are located at partitions’ borders, we use the global query graph to build fragments of triples at those locations instead of building fragments for the whole data set.

4.1 Measuring the Engagement Level

For each border triple d at host h , we can measure two vital values, which we would then use to calculate the benefit of replicating it. The first value is its distance from the border *depth* while the second value is its engagement

level with the global queries graph G_p . Each vertex in G_p builds fragment of assigned triples. In order to measure the engagement level of a triple d , we first find how many vertices in G_p match d ; then for each vertex we look at its neighbour vertices and check how d is engaged with previous triples assigned to their fragments – then we consider the maximum possible engagement. In our running example, suppose that we are checking a border triple $d = (\text{b:karl} : \text{name} \text{ "Karl"})$; we can find that it is matching in G_p the vertices: $v_1 = (\Omega : \text{name} \text{ "Karl"})$ and $v_2 = (\Omega : \text{name} \Omega)$. We first take v_1 and check its neighbour vertices starting by $(\Omega : \text{born} \Omega)$ call it v_3 . We check if we have already a triple assigned to the fragment of v_3 and can be joined with d ; such triple should be in the form $(\text{b:karl} : \text{born} \Omega)$. If the check is true, we add 10 – which is the weight of edge (v_1, v_3) – to the current engagement level of d . We repeat for other vertices connected to v_1 . Then we repeat the calculation for v_2 and assign d to the fragment whose vertex gives the maximum engagement value. This method assigns to any triple a numerical engagement level, which we use in the next section to optimize the replication decision.

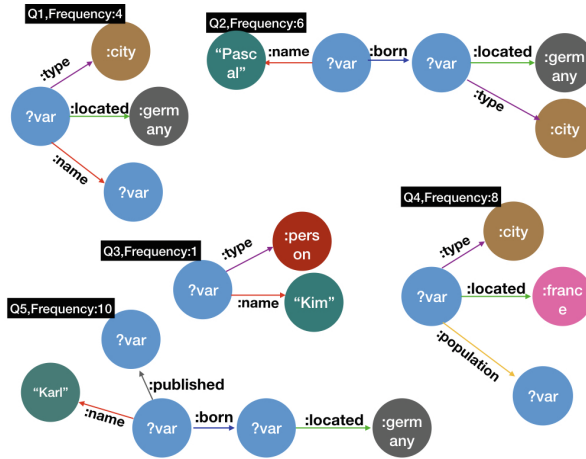


Fig. 1. Sample queries workload

5 Space Adaptability

Having more local replications increases the chance of local execution of query. However, the main limiting factor is the physical availability of storage space. Each host has a local main partition and needs to make a binary decision about each triple that is located in the neighbour hosts, whether to replicate this triple or not. We propose adding a third option which is to replicate the triple in compressed form. This highly increases the system flexibility to adapt to different levels of storage space.

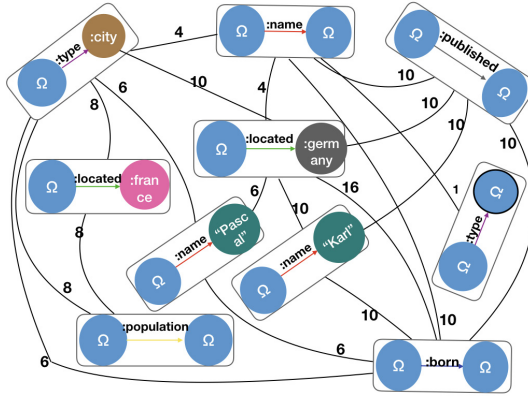


Fig. 2. Global query graph

5.1 Compressed Replications

In triple stores like RDF3X, a dictionary is used to map the textual URIs (found in the triples) to compressed numerical data; the mappings are then stored in different indices called dictionaries. We exploit these different data representations in the context of replication by defining two types: *full replication* where the host has full information about the graph data including the dictionary entries, and the *compressed replication* where the host has only the compressed numerical data. In order to show the effect of the two types on query execution we give the following example: Consider a query q which is composed of 4 triple patterns, and assume for simplicity that it was received for execution by 2 hosts (h_1 and h_2). Assume further that in the first round of execution, each host was able to find matches for one half of q , that is 2 out of 4 triple patterns. A second round of execution is necessary to produce the final query result, but it requires this time moving the first-round result of one host to the other. If h_1 has a full replication of the triples resulting from the first round in h_2 , it could directly perform the join locally, do the dictionary reverse mapping and deliver the textual query result. The only practical reason for h_1 not having the required replication is the insufficient storage space. If h_1 has a compressed replication (the numerical form) of the data produced at h_2 , it could perform the full second round locally, but requires sending the query result to the host where the dictionary entries are physically stored, in order to perform the dictionary reverse mapping and produce the textual final results. The size of the compressed data is mainly related to the number of triples in the data set, while the size of the textual data is related to the length of the text used. If G is an RDF graph and has a size of T triples, the minimum length of a numerical code value used in dictionary is given by: $\log_2 \frac{T}{\kappa}$ bits, where κ is the average number of edges per vertex in G . The total size in bytes of the T triples when stored in the numerical form is given by:

$$size(T) = \frac{3T \cdot \log_2(\frac{T}{\kappa})}{8} \quad (1)$$

Thus the size of the compressed data set is mainly relative to the number of the triples in the data set, with small logarithmic factor. We practically compared the sizes of both compressed and full data sets with different numbers of triple, and we find that the full data sizes is approximately 8 times greater than the corresponding compressed data. We use this value later in our cost functions in the next section.

5.2 Optimizing the Replication Decision

In Sect. 4.1 we obtained the engagement level of any triple with the global query graph which was built from the given workload. We use the engagement level of a triple d besides its *depth* (its distance from the partition border) to measure the importance of d for replication. The highly important triples are fully replicated, while the non-important triples are not replicated at all. The triples in between are having the possibility to have compressed replication. In order to have a systematic operation, we define cost and benefit functions that are smooth and flexible such that the system can directly make a decision about any triple. We start by finding the relative cost that a host i has to pay from its storage space in order to make a full replication to all the triples at depth δ in its neighbours partitions.

$$cost(i, \delta) = \frac{8D_i \cdot s_t}{\mathring{S}} \quad (2)$$

Where D_i is the total number of triples at depth δ in all hosts other than host i , s_t is the size of a single compressed triple given by Formula (1), \mathring{S} is the current remaining storage space available for replications at host i , and the factor 8 represents the size ratio of full to compressed data.

On the other hand, the benefit for host i of replicating the triples at depth δ is given by the following.

$$benefit(i, \delta) = \frac{1}{\delta} \cdot \frac{1}{R} \cdot \frac{engagementLevel}{maxRecordedEngagementLevel} \quad (3)$$

The factor R can be more than 1 when the network performance is relatively poor, or the length of the queries is expected to be small. Our replication system works from the perspective of each host, iteratively considers the triples by their depth, and performs full replication to any triple which has more benefit than cost; otherwise it performs compressed replication. The cost-benefit functions are designed such that the cost increases as the host is running out of space, while the benefit of replicating a triple is related to its distance and to the extent of its relative engagement to the workload-derived global queries graph. If the system happens to have abundance of storage space, the cost will resist the increase with deeper depth; if the system is very short in storage space, the cost would get rapidly high when the depth increases, such that the compressed replication

would be favoured in such situations. If the host runs out of space at certain depth, the replication operation would halt and the triples located at deeper depth are excluded from any replication. This replication approach is highly adaptable with any available storage space and available quality of collected workload, and provides optimized support for the initial data set partitioning.

6 Evaluation

In order to evaluate our partitioning and replication approach, we developed a distributed version of RDF3X¹ and adopted its basic operators to support the distributed processing of SPARQL queries. Our test system is composed of four shared-nothing hosts. Each host maintains an adopted version of RDF3X. RDF3X maps the textual RDF data into compressed numerical form, before storing them in six indices. The compressed data fit directly in our model of compressed replications. One out of the four hosts contains a separate partitioning layer that communicates with the external METIS tool to achieve the initial static partitioning with the initial absence of any workload.

Then the system starts receiving and executing queries; it uses the queries with their frequencies to build its workload. The partitioning layer in each host analyzes the workload and builds the global queries graph which is then used to measure the engagement level of the border triples. The replication is then performed based on the benefit and cost functions described earlier in Sect. 5.2. As a test set, we used the YAGO core² data set with more than 50M triples. We compare our system with three implementations based on WARP, Partout and Huang. Since Partout doesn't explicitly describe a replication method, and for the sake of fair comparison, we followed the same fragment assignment method used in the partitioning to build replication within the available space.

We focused in our evaluation on measuring how the system practically responds to different levels of workload and storage space. Regarding the storage space, we made three system-level runs with three gradual limits on the available storage space, and let the four systems perform replications according to their approaches as much as the storage space limit permits. The *first* level run has less than 3% of storage space available for replication, while the *second* and *third* have 10% and 50%, respectively.

The second factor to consider is the quality of the workload in terms of the heterogeneity of frequencies. We performed two runs to the workload-based systems (excluding Huang which does not consider workload, and excluding the initial static partitioning periods from our system and WARP). The first level of workload is designed to be poorer in terms of its queries frequencies distribution and its correlation to the future queries.

The last factor to consider is the type of queries used for evaluation. We generated random queries for testing purpose with different types and lengths,

¹ <https://code.google.com/archive/p/rdf3x/>.

² <https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/>.

but we focused on *chain queries* since it is better showing the effect of the partitioning and replication approach. The longer the query, the more its chance to go beyond the partition border. In this regard, we classified the queries also into three levels upon their length and complexity, starting from level 1 short and simple queries towards level 3 with the longest and most complex queries. On these queries complexity levels we aggregate the output execution times.

Figures 3 and 4 show the response times in milliseconds of our query execution over the four approaches: WARP, Partout and Huang, and our approach which we labeled APR. The Q.Type represents the query type explained above, while S.Level refers to the level of storage available in the system. Before executing the queries shown in Fig. 3, the system first sets up its partitioning and replication under the level 1 quality of workload, and with the three levels of storage space, giving a total of 3 full system setups. The queries in Fig. 3 are then subdivided into three levels based on their complexity. The poor workload affects directly the average response times of WARP and Partout especially on the low storage space level: here, WARP and Partout show performance similar to the static approach Huang. However, WARP enhances when there is enough storage space making use of its fixed-hop replications. Our system shows obvious better response time making use of its compressed replications which could still provide border replications in difficult storage space condition. Moreover, our system was less affected by the non-homogeneous workload, since its able to distinguish the highly referenced parts of the workload giving them higher priority to replicate. Figure 4 shows the systems behaviour in case of a more representative and homogeneous workload, where WARP could perform well when the queries were not too long. Our system is still showing average better performance and was able to locally execute most of the queries. Huang shows no difference in execution with respect to the workload but was able to perform better when there was more storage space employed for static replication.

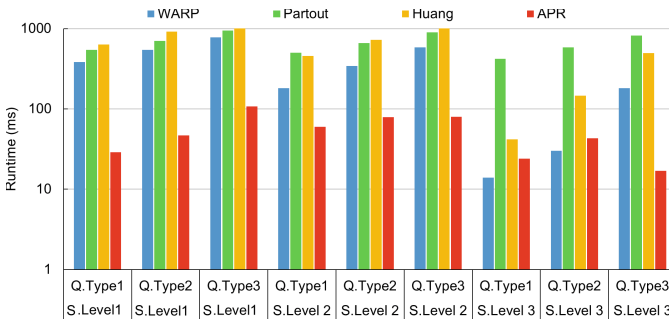


Fig. 3. Query performance under workload level 1

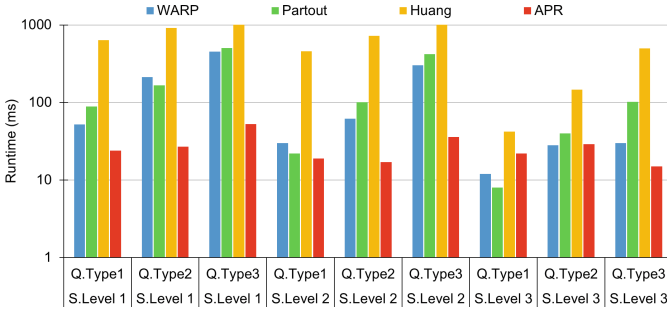


Fig. 4. Query performance under workload level 2

7 Conclusion and Future Work

In this work we have presented a partitioning and replication layer that can be efficiently integrated in a distributed RDF triple store that is built on shared-nothing working nodes. Our approach is optimized towards the real world physical factors that impact the query processing performance of partitioned data. Our replication approach directly adapts itself with the availability of storage space. The replication approach shows privileged adaption with differing quality of workload compared to other approaches. As a future work we first consider adding more flexibility to the partitioning and replication approach to tune itself more smoothly on running time and while the workload is being collected and built – instead of the staged steps that we have considered in our work so far. We also consider enhancing the initial partitioning stage that needs currently to be fully carried out by one host, and optimizing the memory consumption to increase the size of the data set that the system can handle.

Acknowledgements. The authors would like to thank Deutscher Akademischer Austauschdienst (DAAD) for providing funds for research on this project.

References

1. Al-Ghezi, A., Wiese, L.: Adaptive workload-based partitioning and replication for RDF graphs. In: Database and Expert Systems Applications, pp. 377–388. Springer International Publishing (2018)
2. Galárraga, L., Hose, K., Schenkel, R.: Partout: a distributed engine for efficient RDF processing. In: Proceedings of the 23rd International Conference on World Wide Web, pp. 267–268. ACM (2014)
3. Gurajada, S., Seufert, S., Miliaraki, I., Theobald, M.: TriAD: a distributed shared-nothing RDF engine based on asynchronous message passing. In: Proceedings of the ACM International Conference on Management of Data, pp. 289–300. ACM, New York (2014)
4. Hose, K., Schenkel, R.: WARP: workload-aware replication and partitioning for RDF. In: IEEE 29th International Conference on Data Engineering Workshops (ICDEW), pp. 1–6 (2013)

5. Huang, J., Abadi, D.J., Ren, K.: Scalable SPARQL querying of large RDF graphs. *Proc. VLDB Endow.* **4**(11), 1123–1134 (2011)
6. Margo, D., Seltzer, M.: A scalable distributed graph partitioner. *Proc. VLDB Endow.* **8**(12), 1478–1489 (2015)
7. Neumann, T., Weikum, G.: The RDF-3X engine for scalable management of RDF data. *VLDB J.* **19**, 91–113 (2010)
8. Padiya, T., Kanwar, J.J., Bhise, M.: Workload aware hybrid partitioning. In: *Proceedings of the 9th Annual ACM India Conference*, pp. 51–58. ACM (2016)
9. Peng, P., Chen, L., Zou, L., Zhao, D.: Query workload-based RDF graph fragmentation and allocation. In: *EDBT*, pp. 377–388 (2016)
10. Wu, B., Zhou, Y., Yuan, P., Liu, L., Jin, H.: Scalable SPARQL querying using path partitioning. In: *2015 IEEE 31st International Conference on Data Engineering (ICDE)*, pp. 795–806. IEEE (2015)
11. Xu, Q., Wang, X., Wang, J., Yang, Y., Feng, Z.: Semantic-aware partitioning on RDF graphs. In: Chen, L., Jensen, C.S., Shahabi, C., Yang, X., Lian, X. (eds.) *APWeb-WAIM 2017*. LNCS, vol. 10366, pp. 149–157. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-63579-8_12
12. Zhang, X., Chen, L., Tong, Y., Wang, M.: EAGRE: towards scalable I/O efficient SPARQL query evaluation on the cloud. In: Jensen, C.S., Jermaine, C.M., Zhou, X. (eds.) *ICDE*, pp. 565–576. IEEE Computer Society (2013)



Performance Comparison of Three Spark-Based Implementations of Parallel Entity Resolution

Xiao Chen^(✉), Kirity Rapuru, Gabriel Campero Durand, Eike Schallehn,
and Gunter Saake

Otto-von-Guericke-University of Magdeburg,
Universitaetsplatz 2, Magdeburg, Germany
{xiao.chen,kirity.rapuru,campero,eike,saake}@ovgu.de

Abstract. During the last decade, several big data processing frameworks have emerged enabling users to analyze large scale data with ease. With the help of those frameworks, people are easier to manage distributed programming, failures and data partitioning issues. Entity Resolution is a typical application that requires big data processing frameworks, since its time complexity increases quadratically with the input data. In recent years Apache Spark has become popular as a big data framework providing a flexible programming model that supports in-memory computation. Spark offers three APIs: RDDs, which gives users core low-level data access, and high-level APIs like DataFrame and Dataset, which are part of the Spark SQL library and undergo a process of query optimization. Stemming from their different features, the choice of API can be expected to have an influence on the resulting performance of applications. However, few studies offer experimental measures to characterize the effect of such distinctions. In this paper we evaluate the performance impact of such choices for the specific application of parallel entity resolution under two different scenarios, with the goal to offer practical guidelines for developers.

Keywords: Entity resolution · Apache Spark · Parallel computation
High/low-level APIs

1 Introduction

Undoubtedly, we are in the era of data. Along with computers being a pervasive companion in our daily lives, and Internet services connecting the world, data volumes and their variety have exploded in recent years. In this situation, big data processing frameworks are used to fulfill the needs of processing such large-scale data and supporting analysis tasks. With the help of such frameworks, developers can abstract the complexities of distributed programming, failure handling and data distribution, allowing them to focus on core domain tasks. Entity Resolution (ER) is a typical application that requires big data processing

frameworks. The reason is as follows: Its task is to identify digital records that refer to one real-world entity for one input dataset or to link and merge them when more than one input dataset exists. Its straightforward and common solution is pair-wise ER, which means it compares all possible combinations of input records and calculates their similarity scores. Based on their scores we can decide whether the pair of records refer to the same entity or not. As a result ER can be a time consuming process, which can be treated as a Cartesian Product on the input datasets. Therefore, we can see that ER is a typical case that requires big data processing frameworks, comparing to other applications that do not need quadratic time to perform.

Therefore, so far there has been a large majority of research that explores using big data frameworks to support parallel ER [3]. In recent years parallel ER using Apache Spark has received attention from the data management community. Spark is one of the most popular frameworks nowadays. Compared to earlier frameworks, such as MapReduce, its programming model is more flexible without the need of abstraction to “map” and “reduce” phases and it could provide good speeds by supporting in-memory computation without storing intermediate results to disk. In addition, MapReduce provides low-level programming, while Apache Spark supports both low-level and high-level programming because of the integration with the SQL library. However, almost all existing Spark-based parallel ER research uses Spark’s low-level API: the RDD API [10, 12, 14], and to date, there’s little research tackling parallel ER through the high-level APIs: DataFrame and Dataset (available for Scala and Java language. In this paper, APIs mean APIs for Java language) except for our research in [4], which introduces a detailed ER process using DataFrame API. In Apache Spark 2.0, Dataset and DataFrame APIs are unified and DataFrame is considered as Dataset<Row>. In this paper, we implement parallel ER with all three APIs and compare their performance under two different scenarios. These two scenarios mean two parallel ER processes: one has five steps including its last step: the evaluation step, the other one contains only the same first four steps without the evaluation step. The reasons why we have such two scenarios will be explained in Subsect. 3.2.

The rest of the paper is structured as follows: In Sect. 2, we provide some necessary knowledge of Apache Spark. In Sect. 3, we describe our employed ER process, and then explain our implementation considerations. Consequently, in Sect. 4, we introduce our experiments to compare and discuss three implementations. We conclude our paper in Sect. 5.

2 Apache Spark

Apache Spark is designed for fast and general processing of large-scale data. It supports acyclic data flow and in-memory computing, which make Spark run programs up to 100 times faster than Hadoop MapReduce in memory, or 10 times faster on disk [1]. Its main abstraction is resilient distributed data (RDD), which corresponds to the first kind of API in Spark: the RDD API, is a collection of immutable data partitioned across the nodes of the cluster that can

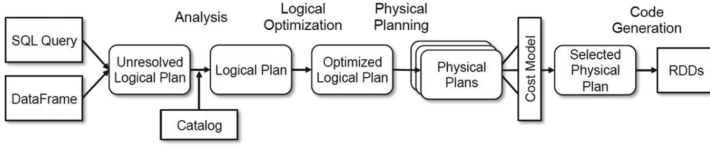


Fig. 1. Phases of query planning in Spark SQL [2]

be operated on in parallel, supports in-memory computing and provides fault tolerance. RDD supports two kinds of operations: transformations and actions. Transformations only create new RDDs from existing RDDs, while actions run a computation on RDDs and return values. In order to enable Spark to run more efficiently, transformations in Spark are all lazy, which means transformations for datasets are only scheduled but not computed right away. Persisting data in memory is one of the most important capabilities in Spark. There are several persistence levels available, such as `MEMORY_ONLY`, `MEMORY_ONLY_SER`, `MEMORY_AND_DISK`, which differentiate them from the location to persist data (memory or disk), whether to serialize data before persisting.

Spark also integrates four libraries: Spark SQL, MLlib for machine learning, GraphX for graphs and graph-parallel computation and Spark streaming for building scalable fault-tolerant streaming applications [1]. Spark SQL is the module of Apache Spark for structured data, which enables developers to query structured data inside Spark programs. Before Spark 1.6, there were only two APIs: RDD and DataFrame. The RDD API corresponds to the low-level API in Spark and the DataFrame API is the structured API in Spark SQL, which can benefit from a series of optimizations. Figure 1 shows the four phases using its cost-based optimizer Catalyst to optimize the application. It first analyzes a logical plan from references and optimize it, then it chooses the best physical based on costs, and last generates code to compile the query to Java code [2]. Besides, it also has a columnar storage called Tungsten, and is able to use Kryo serialization or Encoder to replace traditional Java serialization to minimize storage cost and improve efficiency [9]. However, a disadvantage of using the DataFrame API compared to the RDD API is that the DataFrame API has no type-safety for analysis errors, i.e., analysis errors cannot be caught during compile time. In order to overcome this shortcoming and at the same time keep the advantage of those optimizations, the Dataset API was introduced in Spark 1.6. Therefore, there are three kinds of APIs after Spark 1.6. In Spark 2.0, two structured APIs: the DataFrame API and the Dataset API are unified to a single Dataset API, and the DataFrame API is named `Dataset<Row>`, and the return type of a SQL query or SQL-based API is `Dataset<Row>` [1]. Spark does not force users to decide between a relational or a procedural API, since Spark SQL supports both of them [2].

3 Parallel Entity Resolution with Three Spark Java APIs

In this section, we first present the ER process used to compare the performance of different implementations in Subsect. 3.1. Next, we introduce several global design considerations in Subsect. 3.2.

3.1 The Entity Resolution Process Used for Comparison

To compare the performance of each implementation, we employ the following ER process [5]. The first step is data preprocessing, it is used to clean and standardize the input data. As a stand-in for more complex pre-processing we simply added null value replacement, for easing the similarity calculation.

After preprocessing, blocking is often adopted, which splits the whole input dataset into blocks and allows only to compare records within the same block, to reduce the search space for ER such that the corresponding computational complexity can be improved. Standard blocking techniques are used in our approach. In this step we first define a blocking key. Then, records with the same key are assigned to the same block. Standard blocking is straightforward, efficient and able to achieve reasonable reduction ratio and completeness. The selection of a suitable blocking key is essential for standard blocking, and it is a trade-off to choose it in a more general vs. specific way, based on requirements for efficiency and completeness [11]. In our experiments for datasets with personal information, we experimentally tested different blocking keys and the following blocking key is defined and determined as our final blocking key: first applying the double metaphone algorithm on attributes “surname” and “given_name” as the blocking key. The reason for using double metaphone is that, the double metaphone algorithm is designed to capture common transcription mistakes that people might make when recording information based on what they hear from speakers. In this algorithm, the letters that share a similar pronunciation are transformed to the same representation, this helps to recognize true matches. The decision of using this blocking key is because it reduces the search space for pair-wise comparison by a factor of 3000 and at the same time reaches a satisfactory completeness, which is a balanced solution for both goals.

The third step, pair-wise comparison, is the essential one of the ER process, which also turns ER to a compute-intensive and time-consuming task. Before we calculate the similarity between two records of a pair, we need to first get all candidate pairs, whose similarities are needed to calculate. This can be achieved by a join operation using the blocking key as the join attribute. During this operation, almost half of the pairs can be removed due to the transitive property, for example, the result of the join operation contains two pairs (a, b) and (b, a), one of them can be removed, because the similarity calculation is transitive. Then we have the final candidate pair set. The next step is to calculate the similarities of all pairs. Almost all datasets have more than one attribute, similarity functions need to be applied on each attribute to get a total comparison score afterwards. Similarity functions should be chosen based on the attribute properties. In our case, we applied Jaro-Winkler distance for such attributes that

are strings (including number strings), because Jaro-Winkler has been proved to be a good and efficient edit-distance metric for short strings, such as for name matching [7]; While for numbers, absolute difference functions are chosen because of their similarity and understandability.

After all intermediate results are obtained from pair-wise comparisons, the classification step is executed to decide whether each pair of records refers to the same real-world entity or not. In the classification phase, we classified each record pair to match or non-match based on the similarity scores of their attributes using a threshold-based method, which sums up all similarity scores to a total score and judges whether the score is higher than the threshold or not. If the score of a pair is higher than the threshold, this pair would be recognized as a match pair. In our experiments, 0.75 is determined because it serves the best trade-off between precision and recall. After the classification step, we save the results, which contain all candidate pairs, their similarity scores and the match or non-match decision.

The last step is evaluation. This step is optional based on the user's requirement or the availability of a ground truth. For our performance comparison, we take both scenarios into consideration, i.e., an ER process with evaluation or without evaluation. The reason will be explained in the next subsection. For the scenario with evaluation, we evaluate precision, recall and F-measure of our results. The ground truth in our synthetic data is provided by their record IDs. We count the number of true positive, false positive and false negative through comparing our obtained match or non-match result to the analyzed ground truth. Then precision, recall and F-measure are calculated based on true positive, false positive and false negative values, and we also save their values.

3.2 Global Design Considerations

Next, we will list several global considerations to design the comparison of Spark ER with three APIs.

Choice of Two ER Scenarios for Evaluating Impact of Data Reuse:

The reason, why we have two aforementioned scenarios in this paper: one with the evaluation step (Scenario 1) and one without the evaluation step (Scenario 2), is two fold. On one hand, two processes stand for two kinds of use cases. In some cases, people are concerned about only an approximate result of which records refer to the same entity without the need of knowing the exact precision and recall. Another case is without available ground truth, it is not possible to have the evaluation step. These two cases lead to an ER process without an evaluation step. For other cases, in which people need to know the result quality of ER and the ground truth is available, they lead to the ER process with an evaluation step. Talking a technical perspective, an ER process with an evaluation step involves the case of data reuse, while the process without an evaluation step does not involve data reuse. We want to know how data reuse affects the performance of different APIs.

Principle for DataFrame- and Dataset-Based Implementation: High-level APIs DataFrame and Dataset in Spark have been unified to Dataset API in version 2.0 for ease of learning, in which DataFrame was considered as a special Dataset with a row type and renamed to Dataset<Row> [1]. Nonetheless, for ease of expression in this paper we still use DataFrame API to stand for API of Dataset<Row>, while Dataset API means Dataset<T> API not including the special Dataset<Row> API. The difference between Dataset API (Dataset<T>) and DataFrame API (Dataset<Row>) is explained in the following. The return type of all SQL queries or SQL-like operation is Dataset<Row>, i.e., DataFrame, our DataFrame-based implementation is actually SQL-based implementation. And for our Dataset-based implementation, after we loaded the data as Dataset<T>, we don't use any API of Dataset<Row>, but other possible APIs for the general Dataset<T>.

Optimizations on Each Implementation: We optimize each implementation by tuning the following parameters: the level of parallelism and possible persistence options. Choosing a suitable level of parallelism for RDD is crucial to reach a good performance. The default level of parallelism for RDD is based on the input data size. For an ER task, for the case that the input data size is small, the data size for pair-wise comparison can be very large. Therefore, if the level of parallelism is determined by the input size, it cannot fulfill the parallelism requirement for steps after blocking. As a result, we have to tune the level of parallelism parameter to reach a good performance (After test experiments on different levels of parallelism are conducted, 320 is finally chosen for the RDD-based implementation). For Dataset-based APIs, the default level of parallelism is 200, which is proved to be a sufficient number in our experience and needs no specific tuning considerations.

Regarding persistence options, for Scenario 1 with the evaluation step, which involves data reuse, we designed different persistence options, which persist different relevant data. These persistence options are tested experimentally and for performance comparison, we take the best persistence option for each implementation and compare their runtime. To achieve the best performance in Spark, the persistence level taken is MEMORY_ONLY for all experiments. For Scenario 2 without the evaluation step, because it does not involve any data reuse case, persistence is not necessary.

4 Experiments

4.1 Experimental Setting

In this subsection, we introduce our experimental setting. It relates to two aspects. On the one hand, we describe the datasets that we used; on the other hand, we demonstrate our cluster based on Hortonworks Data Platform (HDP).

Table 1. Datasets used in experiments

Datasets	Name	Input size
1	$5 * 10^5 + 5 \%$	57.3M
2	$5 * 10^5 + 50 \%$	79.9M
3	$10^6 + 5 \%$	114M
4	$10^6 + 50 \%$	160M

Datasets. We use synthetic datasets for our experiments. By using synthetic datasets, we can easily know the ground truth and we can control the property of the datasets, such as sizes or duplicate percentages of datasets. In addition, our generated data is based on real-world data and can follow similar characteristics to the real data [6]. It is generated by a data generator called GECO [13]. GECO consists of GEnerator and COrruptor, which is specifically designed for generating ER datasets. For each record, an identifier is assigned, which can show the ground truth of the dataset. For the sake of data diversity and being able to cover different cases in reality, we generated the datasets stored in csv files using GECO, which all contain personal information with the following 14 attributes: rec-id, gender, given-name, surname, postcode, city, telephone-number, credit-card-number, income-normal, age-uniform, income, age, sex, and blood-pressure. The datasets used are shown in the upper part of Table 1. Their names can reflect their sizes and duplicate percentages: the first parts mean the number of original records that a dataset contains, such as 10^6 means there are 1 million original records. The second part stands in for the duplicate percentage based on the original records, such as 5% means the number of duplicates are 5% of the number of original records and are inserted into the dataset. Because of privacy reasons, it is very hard to find real datasets that contain personal information.

Cluster on Hortonworks Data Platform. Our cluster used for all experiments is deployed by HDP (Hortonworks Data Platform)-2.6, which is an open source Apache Hadoop distribution based on a centralized architecture (YARN) [8]. The cluster has ten hosts, which includes one node to manage the cluster, two nodes as HDFS NameNodes (one active and one standby) and seven executor nodes as HDFS DataNodes, which can also be used to run MapReduce, Hive, Tez and Spark applications. Each of them runs on virtual machines, whose hyper-visor is VMWare ESXi. Each host has four CPU cores, 16 GB RAM and 150 GB hard disk. However, our cluster is heterogeneous, since four nodes are with four Intel Xeon E5-2650 @2.00 GHz cores, two nodes are with four Intel Xeon E5-2650v2 @2.60 GHz cores, the last nodes are with four Intel Xeon E5520 @2.27 GHz cores. All hosts are connected by a 10 GBit Ethernet with a star topology. We submitted our Spark applications with jar files to our cluster and conducted a series of experiments to evaluate the runtime for different frameworks. The corresponding Spark version is 2.2.0 and Java version is 1.8.0. To

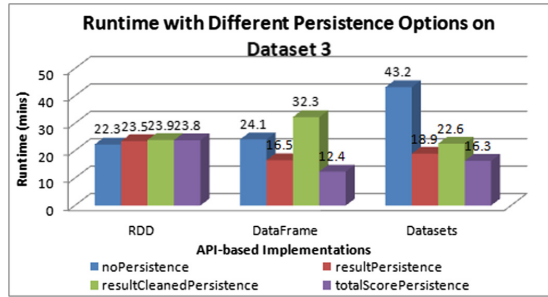


Fig. 2. Overview of runtime with four different persistence options on dataset 3

ensure a successful run of all submissions and that constant cluster resources are used for each submission, we have the following configurations: The driver has 1 GB memory for all experiments. Each executor is allocated with 6 GB memory, 2 GB space for executor memory overhead and 4 cores. We run each experiment five times and after dropping the highest and lowest result, our final result was obtained by averaging the remaining three results.

4.2 Scenario 1: With the Evaluation Step

Persistence Options for Three Implementations: For the first scenario, before we compare the runtime for each implementation, we need to test those persistence options we designed to find the best persistence option for all implementations. The persistence options are designed with the help of analyzing our ER process and observing the Spark web UI. Straightforwardly, data that is involved in multiple actions needs to be persisted to avoid repetitive computation. For our ER process, Result data which contains the pair-wise comparison results and ResultCleaned Data which contains the ground truth that can be used for the evaluation are data that need to be persisted with this straightforward thinking. If such data is not persisted, each time a related action is triggered, they have to be obtained through a series of transformations from the original input data. The TotalScore data is not involved by considering the above explained straightforward thought, but we consider to persist it because of the following consideration: The number of rows in totalScore and result data is quite large, the extra time needed due to persisting result data should be much more than that for totalScore data. Therefore, we have the assumption that persisting totalScore data may have a better effect than straightforwardly persisting result data. Based on the explanations above, we designed the following four persistence options: noPersistence, resultPersistence, resultCleanedPersistence and totalScorePersistence. We submitted all ER implementations with all four persistence options to our entire cluster (seven executors) to record all the runtime on datasets to find the best persistence option for them.

Figure 2 shows an overview of runtime on three implementations with four different persistence options on dataset 3 for the scenario with the evaluation step. The results on other datasets show a similar trend, therefore, here we only show the result on dataset 3 due to space limitation. Surprisingly, there are no obvious performance difference with those four persistence options for the RDD-based implementation. Even with persisting the relevant data, Spark runs the RDD-based implementation in the same way, which leads to similar runtime. And persisting data for the RDD-based implementation cannot avoid overhead, for the RDD-based implementation, we did not use any persistence for it. In contrast, DataFrame- and Dataset-based implementations can benefit from persistence. As we can see from Figure 2, by persisting result data, an improvement of a factor up to 1.5 and 2.3 for DataFrame-based and Dataset-based implementation can be achieved, respectively. By persisting totalScore data instead of the straightforward choice: result data, efficiency improvement can reach a factor up to 2x and 2.7x for them. We conclude that the straightforward choice to persist is not always the best, when the data that is used for multiple actions is quite large. Instead of directly persisting it, we can judge whether it is possible to persist its previous data when the computation between them is not time-consuming and their data sizes differ much, it is normally beneficial to persist the former one instead. Therefore, we take persisting totalScore data as the best persistence option for DataFrame- and Dataset-based implementations. Since the RDD-based implementation cannot benefit from persistence options, for runtime comparison we also took DataFrame- and Dataset-based implementations without any persistence into consideration.

Runtime Comparison. Based on the above discussion on persistence options, for Scenario 1, we have five different cases to compare their runtime: RDD-based without persistence, DataFrame-based without persistence, DataFrame-based with best persistence, Dataset-based without persistence, Dataset-based with best persistence. Each case is submitted to our Spark cluster with seven available nodes on different datasets and their corresponding runtime is recorded. Figure 3 shows the comparison result. As we can see from it, the DataFrame-based implementation with best persistence is the most efficient one, which is up to 2.5x and 1.6x faster than the RDD-based implementation and the Dataset-based implementation, respectively. However, if without persisting any data, the RDD-based implementation has a similar speed as the DataFrame-based implementation, and sometimes is even slightly faster than the DataFrame-based implementation, but both of them are much faster than the Dataset-based implementation. In conclusion, by using a proper persistence option, the DataFrame-based and the Dataset-based implementation are able to outperform the RDD-based implementation due to the obtained benefits from persistence. The RDD-based implementation did not change its running plan with the persistence options and cannot get benefits from persistence in Scenario 1.

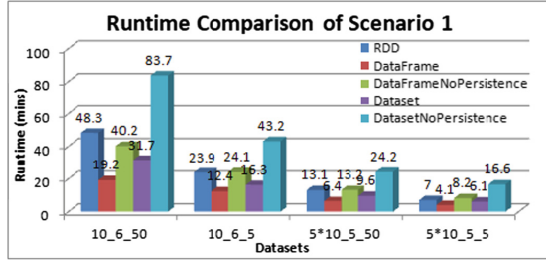


Fig. 3. Runtime comparison for Scenario 1

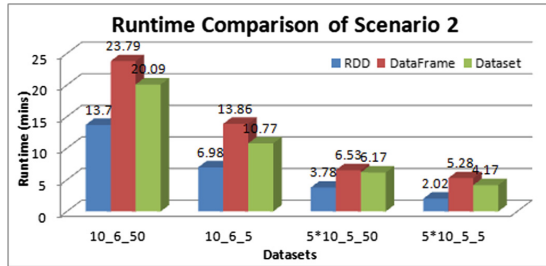


Fig. 4. Runtime comparison for Scenario 2

4.3 Scenario 2: Without the Evaluation Step

For the second scenario, our ER process does not include the evaluation step and we removed this step from Scenario 1 and kept the rest of steps as Scenario 2. Since there is no data reuse in Scenario 2, it is not necessary to have any persistence options. We submitted three implementations to our 7-nodes-cluster on the four datasets introduced in Table 1. Figure 4 shows the results. For all datasets, it shows a consistent result: among three implementations, the RDD-based implementation is the fastest, up to 2.6x and 2.1x faster than the DataFrame-based and the Dataset-based implementation, respectively. The DataFrame-based implementation is the slowest, even slower than the Dataset-based implementation, which is the opposite side to Scenario 1, when we consider the best persistence case for all of them. In conclusion, expected physical and logical optimizations do not help the DataFrame-based and the Dataset-based implementations much. The RDD-based implementation shows the best efficiency for running a general ER process without the evaluation step (without data reuse).

4.4 Threats to Validity

Our experimental results can be subject to several critical considerations.

As internal threats, our results might be affected by unstable network transmission speed between cluster nodes, or other factors in resource usage that

influence the causal behavior we observe between performance and choices for API/persistence configurations.

As external threats, first the synthetic datasets facilitate the ER process and the algorithms we use, specially they lead to a specific evaluation process; if we select another dataset, the results might be different. Our choices for calculation of similarity, blocking and threshold-based matching are selected to represent performance-wise a general use case, they also match well the datasets we use. Other configurations, using more complex similarity measures, different blocking techniques, and an alternative matching process (e.g., with classification) could lead to different performance observations for our experiments. Secondly, our cluster is heterogeneous and with the limited number of nodes available, which restricts the possible interpretation regarding speed-up and scalability for large scale applications. In our experiments, results indicate a promising trend that would have to be verified for large-scale settings. The cluster is deployed on HDP 2.6, the Spark version 2.2 and Java version 1.8.0 are installed on it. The results may change when using different platforms, Spark or Java versions. Third, finely tuned implementations might possibly lead to other observations, but for generality we adopted a straightforward solution.

5 Conclusions

In this paper we compare the performance of parallel ER using three APIs: RDD, DataFrame and Dataset in Spark, for two scenarios of a general ER process. For Scenario 1, the RDD-based implementation runs faster than the other two implementations, without consideration of any persistence option for them. However, the DataFrame and Dataset implementations benefit from persistence and the RDD implementation does not. Therefore, with the best persistence option for DataFrame and Dataset implementations, they are able to outperform the RDD implementation. For Scenario 2, the RDD implementation is the fastest, which conforms to the case in Scenario 1 without consideration of any persistence option.

High-level APIs are expected to be more convenient for developers. Furthermore they can be expected to outperform their low-level counterparts, given the physical and logical optimizations that they can introduce to the code, we observe that this is not the case for parallel ER, since their competitive edge only appears when persistence options are possible. Though it can be expected that future versions of Spark will overcome such limitations, we report a snapshot of the current state-of-the-art in using this framework for ER.

Acknowledgment. This work was supported by China Scholarship Council [No. 201408080093].

References

1. Apache: Apache spark. <http://spark.apache.org/>. Accessed 10 April 2018
2. Armbrust, M., et al.: Spark SQL: relational data processing in spark. In: Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, pp. 1383–1394. ACM (2015)
3. Chen, X., Schallehn, E., Saake, G.: Cloud-scale entity resolution: current state and open challenges. *Open J. Big Data (OJBD)* **4**(1), 30–51 (2018)
4. Chen, X., Zoun, R., Schallehn, E., Mantha, S., Rapuru, K., Saake, G.: Exploring spark-SQL-based entity resolution using the persistence capability. In: International Conference: Beyond Databases, Architectures and Structures (2018, Forthcoming)
5. Christen, P.: Data Matching: Concepts and Techniques for Record Linkage, Entity Resolution, and Duplicate Detection. DCSA. Springer Science & Business Media, Heidelberg (2012). <https://doi.org/10.1007/978-3-642-31164-2>
6. Christen, P., Vatsalan, D.: Flexible and extensible generation and corruption of personal data. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, CIKM 2013, pp. 1165–1168. ACM, New York (2013). <https://doi.org/10.1145/2505515.2507815>
7. Cohen, W., Ravikumar, P., Fienberg, S.: A comparison of string metrics for matching names and records. In: KDD Workshop on Data Cleaning and Object Consolidation, vol. 3, pp. 73–78 (2003)
8. Hortonworks: Hortonworks data platform. <https://hortonworks.com/products/data-platforms/>. Accessed 25 June 2018
9. Karau, H., Warren, R.: High Performance Spark. O’Reilly Media, Sebastopol (2017)
10. Mestre, D.G., Pires, C.E.S., Nascimento, D.C., de Queiroz, A.R.M., Santos, V.B., Araujo, T.B.: An efficient spark-based adaptive windowing for entity matching. *J. Syst. Softw.* **128**, 1–10 (2017)
11. Papadakis, G., Svirsky, J., Gal, A., Palpanas, T.: Comparative analysis of approximate blocking techniques for entity resolution. *Proc. VLDB Endow.* **9**(9), 684–695 (2016). <https://doi.org/10.14778/2947618.2947624>
12. Pita, R., Pinto, C., Melo, P., Silva, M., Barreto, M., Rasella, D.: A spark-based workflow for probabilistic record linkage of healthcare data. In: EDBT/ICDT Workshops, pp. 17–26 (2015)
13. Tran, K.N., Vatsalan, D., Christen, P.: GeCo: an online personal data generator and corruptor. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management, CIKM 2013, pp. 2473–2476. ACM, New York (2013). <https://doi.org/10.1145/2505515.2508207>
14. Wang, C., Karimi, S.: Parallel duplicate detection in adverse drug reaction databases with spark. In: EDBT, pp. 551–562 (2016)



Big Data Analytics: Exploring Graphs with Optimized SQL Queries

Sikder Tahsin Al-Amin¹(✉), Carlos Ordonez¹, and Ladjel Bellatreche²

¹ University of Houston, Houston, USA
sal-amin2@uh.edu

² LIAS/ISAE-ENSMA, Poitiers, France

Abstract. Nowadays there is an abundance of tools and systems to analyze large graphs. In general, the goal is to summarize the graph and discover interesting patterns hidden in the graph. On the other hand, there is a lot of data stored on DBMSs that can be potentially analyzed as graphs. External graph data sets can be quickly loaded. It is feasible to load data quickly and that SQL can help prepare graph data sets from raw data. In this paper, we show SQL queries on a graph stored in relational form as triples can reveal many interesting properties and patterns on the graph in a more flexible manner and efficient than existing systems. We explain many interesting statistics on the graph can be derived with queries combining joins and aggregations. On the other hand, linearly recursive queries can summarize interesting patterns including reachability, paths, and connected components. We experimentally show exploratory queries can be efficiently evaluated based on the input edges and it performs better than Spark. We also show that skewed degree vertices, cycles and cliques are the main reason exploratory queries become slow.

Keywords: Graph · Parallel DBMS · SQL

1 Introduction

Within big data analytics graph problems are particularly difficult given the size of data sets, the complex structure of the graph (density, shape) and the mathematical nature of computations (i.e. graph algorithms). In graphs analytics, the goal is to obtain insight and understanding of complex relationships that are present in the graph. Large-scale graphs have been widely applied in many emerging areas. Graphs can be represented in terms of database perspective. However, processing large graphs in large scale distributed system has not received much attention in DBMS using relational queries. Recent works on graphs offer vertex-centric query interface to express many graph queries [5], compares columnar, row and array DBMSs for recursive queries [13]. There are some libraries [9, 17] available for large graph processing. Also, it has been established that columnar DBMS perform better than array DBMS or GraphX for certain kind of graph queries [4].

Relational database systems remain the most common technology to store transactions and analytical data, due to optimized I/O, robustness and security control. Even though, the common understanding is that RDBMSs cannot handle demanding graph problems, because relational queries are not sufficient to express important graphs algorithms and a poor performance of database engines in the context of graph analytics. Consequently, several graph databases and graphs analytics systems have emerged, targeting large data sets, especially under the Hadoop/MapReduce platform. First, we analyze graphs stored on a DBMS. Using SQL query, we can reveal many interesting properties from the graph like *indegree* and *outdegree*, highly connected components, potential paths, isolated vertices, triangles and so on. Then we focus on the evaluation of queries with linear recursion, which solve a broad class of difficult problems. Using recursion, we can answer questions like reachability, detecting cycles, adjacency matrix multiplication, paths and so on. We perform our experiments on columnar parallel DBMS with shared nothing architecture. While our queries can work in any DBMS, it is experimentally that proven that columnar and array DBMSs present performance substantially better than row DBMSs for graphs analysis [13]. Also, parallel database systems have a significant performance advantage over Hadoop MR in executing a variety of data-intensive analysis benchmarks [14].

2 Definitions

This is a reference section which introduces definition of a graph from mathematical and database perspective, recursive queries and our problem definition. Each subsection can be skipped by a reader familiar with the material.

2.1 Graph

Let $G = (V, E)$ be a directed graph with $n = |V|$ vertices and $m = |E|$ edges. An edge in E links two vertices in V and has a direction. Our definition allows the presence of cycles and cliques in graphs. A cycle is a path which starts and ends at the same vertex. A clique is a complete subgraph of G . The adjacency matrix of G is a $n \times n$ matrix such that the cell i, j holds a 1 when exists an edge from vertex i to vertex j .

From database perspective, graph G is stored in relation (table) E as a list of edges (adjacency list). Let, relation E be defined as $E(i, j, v)$ with primary key (i, j) representing the source and destination vertices and v representing a numeric value e.g. cost/distance. A row from relation E represents the existence of an edge. In summary, from a mathematical point of view E is a sparse matrix and from a database perspective, E is a long and narrow table having one edge per row. For example, if we have city names as graphs, then i and j will be city names and v can be the distance or travel cost between them. However, if we have phone calls as a graph, then E will have edges per person pair, not one edge per phone call or message.

2.2 Recursive Queries

Recursive queries are used here to analyze graphs. Let, relation E is the input for a recursive query using columns i and j to join E with itself multiple times. Let R be the resulting relation returned by a recursive query, defined as $R(d, i, j, v)$ with primary key (d, i, j) , where d represents recursion depth, i and j identify an edge at some recursion depth and v represents the numeric value. We include v in both E and R to have consistent definitions and queries. We study queries of the form: $R = R \cup (R \bowtie E)$, where the result of $R \bowtie E$ gets added to R itself.

2.3 Problem Definition

In this paper, we study how to express the computation of several graph algorithms with queries and how we can optimize those queries. We solve from simpler to harder graph problems with relational queries.

3 Exploring Graphs

In this section, we discuss the main contribution of our work. First we discuss how we can optimize relational queries and then use the optimized queries to solve from simpler to harder graph problems.

3.1 Optimizing Recursive Queries

Pushing GROUP-BY and Duplicate Elimination: First, we review our previous optimization technique to optimize recursive queries [13]. This optimization corresponds to the classical graph problem (reachability). $E \bowtie E$ may produce duplicate vertex pairs. We can compute a GROUP BY aggregation on E , grouping rows by edge with the grouping key i, j . A byproduct of a GROUP BY aggregation is that duplicates get eliminated in each intermediate table. Therefore, a SELECT DISTINCT query represents a simpler case of a GROUP BY query.

In recursive queries, the fundamental question is to know if it is convenient to wait until the end of recursion to evaluate the GROUP BY or it is better to evaluate the GROUP BY during recursion. Considering that a GROUP BY aggregation is a generalization of the π operator extended with aggregation functions, this optimization is equivalent to pushing π through the query tree, like a traditional SPJ query. In relational algebra terms, the unoptimized query is $S = \pi_{d,i,j,sum(v)}(R)$. And the optimized query is given below. Therefore, it is unnecessary to perform a GROUP BY or DISTINCT in the final S . Hence, the GROUP BY we used in queries discussed later, they all follow the optimized version.

$$S = \pi_{1,i,j,sum(v)}(R_1) \cup \pi_{2,i,j,sum(v)}(R_2) \cup \dots \quad (1)$$

Partitioning and Maintaining Duplicate Table E: While performing self joins ($E \bowtie E$), it is more feasible to maintain a duplicate copy of E (E_d). As we are self joining on $E.j = E.i$, we can maintain a duplicate version of E and partition one E by i column and other E by j column to optimize the join computation. Partitioning capability of a DBMS divides one large table into smaller pieces based on values in one or more columns. Partitions can improve parallelism during query execution and enable some other optimizations. The graph should be partitioned in such a way that uneven data distribution and costly data movement across the network is avoided. The latter is possible when the parallel join occurs locally on each worker node. Partitioning provides opportunities for parallelism during query processing. There is a distinction between partitioning at the table level and segmenting a projection. Table partitioning segregates data on each node for fast data purges and query performance. And segmenting distributes projection data across multiple nodes in a cluster. Hence, the join query will have better performance.

Here, we perform the partitioning by vertex. All the neighbors of a vertex are stored on the same machine. Our assumption is that there is not one high degree vertex but there are a few high degree vertices. This number is greater than the number of machines in the cluster. If high degree vertices are less than the number of machines, queries will be slow.

Encoding: This optimization method is limited to columnar DBMS only. There are many encoding types in columnar databases. The default encoding is ideal for sorted, many-valued columns such as primary keys. It is also suitable for general purpose applications for which no other encoding or compression scheme is applicable. Encoding options in DBMS include run length encoding (RLE), which replaces sequences (runs) of identical values in a column with a set of pairs, where each pair represents the number of contiguous occurrences for a given value: (occurrences, value). RLE is generally applicable to a column with low-cardinality, and where identical values are contiguous. However, the storage for RLE and AUTO encoding of CHAR/VARCHAR and BINARY/VARBINARY is always the same.

3.2 Building Graph Summaries with Queries

Finding Indegree and Outdegree: The *outdegree* of a vertex v is the number of outgoing edges of v and the *indegree* of v is the number of incoming edges of v . We can get the *indegree* and *outdegree* of each vertices from a graph. This can help to answer queries like which is the most visited page on the web or who is the most popular person in a social network. The following two SQL queries find the *indegree* and *outdegree* respectively. These queries are $\mathbf{1} * E$ and $E * \mathbf{1}$ meaning matrix-vector multiplication.

```
SELECT j AS nodes, COUNT(j) AS indegree
FROM E
GROUP BY j;
```



```
SELECT i AS nodes, COUNT(i) as outdegree
FROM E
GROUP BY i;
```

Using these, we can also detect the source-only or destination-only vertices. That is, the vertices with 0 as *indegree* are source-only and vertices with 0 as *outdegree* are destination-only vertices. However, the above queries only select the vertices who have *indegree* or *outdegree*. To get source-only or destination-only vertices we have to find *indegree* and *outdegree* for all vertices.

Counting and Enumerating Triangles: As triangles are subsets of cliques, it is fundamental to understand connectivity. For instance, triangle count in a network is used to compute transitivity, an important property for understanding graph evolution over time. Triangles are also used for various other tasks completed for real-life networks, including community discovery, link prediction, and spam filtering. It is possible to detect and count how many triangles are presented in the graph using relational queries. We can detect the number of triangles by performing join operations on E ($E \bowtie E \bowtie E$) where each join uses $E.j = E.i$. Here, we are performing a self-join twice. It is a very costly operation and makes the queries slower. As discussed above, we can optimize the query performance by maintaining a duplicate version of E and partition them based on vertices. Then, if E_D is the duplicate version of E , the join operation will be $E \bowtie E_D \bowtie E$.

Exploring Paths:

(a) *Potential Number of Paths:* We can get the potential number of paths from a graph. We can use the information from *indegree* and *outdegree* discussed above. The number of paths (P) passing through a certain vertex is the multiplication of its *indegree* and *outdegree*. The Eq. 2 gives all the paths that can be generated from a graph including cycles. However, if we use GROUP BY or DISTINCT to eliminate duplicate vertex pairs, the number of paths generated will be much less than the one generated by this equation. And paths may also represent connections, for instance, chemical compounds, or subparts of a part, that is, not necessarily distances/cost.

$$potential\ P = \sum (indegree(\forall i) \times outdegree(\forall i)) \quad (2)$$

where $indegree_i = outdegree_i$

(b) *Top K Connectivity Vertices:* Highly connected vertices mean the vertices to which most edges are connected. In other words, these vertices are responsible to generate more paths. Therefore, we can get them from *indegree* and *outdegree*. The highest connected vertex will be $max(indegree(i) \times outdegree(i))$. And the vertices whose both *indegree* and *outdegree* are 0 are isolated vertices in the graph. We can also get top K highest or lowest connected vertices in such way.

(c) *Exploring Paths:* We can find all the vertices starting from vertex u or check if there exists a path between vertex u and v . To find all the reachable vertices starting from u , we can first filter E on $E.i = u$ and store the vertices on another table/relation P . Then to find the reachable vertices, we can perform join $P \bowtie E$ on $P.j = E.i$. This will give reachable vertices of path length 3. To get all the reachable vertices with different path lengths we can repeatedly set P to $P \cup (P \bowtie E)$ on the same Join condition as mentioned before. The union operation will keep only the identical vertex pairs. We can also check if there is a connection between two vertices u and v with this method until a row with $P.j = v$ arrives. As we are filtering rows at first, P is much smaller than E , this bound to have benefits in query time. Also, we can apply the pushing GROUP BY optimization as discussed earlier and eliminate duplicates at each round.

Using this method, we can also find the vertices where distance is below/above a certain value. We only populate P where the value of v column is below/above a threshold value.

Finding Connected Components: A weakly connected component of a directed graph G is a subgraph G' such that for any vertices $u, v \in G$, exists an un-directed path between them. We can compute with the SPJA query for matrix-vector multiplication (join between two tables and aggregation). This is an improvement of HCC, an iterative algorithm proposed in [7]. In contrast with this, we avoid the second join, necessary to find the minimum value for each entry of the new and the previous vector. We propose inserting an artificial self loop in every vertex; by setting $E(i, i) = 1$, for every i . At each iteration, the resulting vector will compute $S_d \leftarrow \pi_{j:\min(E.v * S.v)}(E \bowtie_{j=i} S_{d-1})$. The algorithm stops when the current vector is equal to the second. The relational query is presented below.

```
INSERT INTO S1
SELECT E.i, min (S0.v*1)v
FROM E JOIN S0 on S0.i=E.j
GROUP BY E.i;
```

Adjacency Matrix Multiplication: In our work, we multiply adjacency matrix to get the transitive closure of a graph using recursive queries. The standard and most widely used algorithm to evaluate a recursive query comes from deductive databases and it is called Seminaive [2,3]. Let R_k represent a partial output relation (table) obtained from $k - 1$ self-joins with E (input relation) as operand k times, up to a given maximum recursion depth k : $R_k = E \bowtie E \dots \bowtie E$. Here, each join uses $E.j = E.i$ and is a matrix-matrix multiplication. The general form of recursive join is $R_{d+1} = R_d \bowtie_{R_d.j=E.i} E$, where the join condition $R_d.j = E.i$ links a source vertex with a destination vertex if there are two edges connected by an intermediate vertex. Assuming graph E mentioned in Sect. 2 as input, the π computes $d = d + 1$, $i = R_d.i$, $j = E.j$ and $v = R_d.v + E.v$ at each iteration.

$$R_{d+1} = \pi_{d,i,j,v}(R_d \bowtie_{R_d.j=E.i} E) \quad (3)$$

The final result is the union of all the partial results: $R = R_1 \cup R_2 \cup \dots \cup R_k$ for recursive depth k . This query evaluation stops when R_d becomes empty at some iteration because there are no rows to satisfy the condition then. Applying pushing GROUP BY optimization as discussed earlier helps this query to perform faster as we are eliminating duplicates at each round.

4 Experimental Evaluation

In this section, we provide an overview of how we conducted our experiments and our findings. First, we discuss how we set up the experimental parameters and then we discuss our experimental outcome. We perform comparisons of our queries with Spark in parallel machines and present the results.

4.1 Experimental Setup

DBMS Software and Hardware: We conducted experiments on an eight node cluster each with Intel Pentium(R) Quadcore CPU running at 1.60 GHz, 8 GB RAM, 1 TB disk, 224 KB L1 cache, 2 MB L2 cache and Linux Ubuntu 14.04 operating system. So for parallel computation, total RAM size is 64 GB and total disk space is 8 TB and 32 cores. We used Vertica [8], a columnar DBMS supporting ANSI SQL to execute the queries. However, our queries are standard SPJ queries and will work on other RDBMSs too. We used Python as the host language to generate SQL queries and submit the queries to the databases/systems as it is faster than JDBC. We compared our results with Spark-GraphX which is used for graph-parallel computation in Spark.

Data Sets: We used both synthetic and real graph data sets summarized in Table 1. For synthetic graph data sets, we generated graphs with varying complexity. We generated graphs with varying clique sizes using uniform distribution where clique sizes increase either linearly (cliqueLinear) or geometrically (cliqueGeometric). For cliqueLinear data set, we generate the cliques with sizes 2, 3, 4... and for cliqueGeometric data set we generated the graphs with sizes 2, 4, 8.. and so on. For real data sets we used from the Stanford SNAP repository. Both real data sets have a significant number of cliques and medium diameter. All the time measurements are taken as the average of running each query five times and excluding the maximum and minimum value.

4.2 Parallel Graph Summarization

In DBMS, we calculate the *indegree* for each vertex using the relational query mentioned above. Spark-GraphX includes in its library an implementation of *indegree* as graph operators. It also includes an implementation to find the max *indegree*. Table 2 shows that Spark takes a lot of time to calculate the *indegree* for all data sets. As the relational query in DBMS is very simple and does not require any join or costly operations, it is very fast and executes within seconds.

Table 1. Data sets.

Name	Type	n	m	Density
tree10m	Synthetic	10M	10M	Sparse
wiki-vote	Real	8k	103.6k	Sparse
cliqueLinear	Synthetic	48.5k	10M	Dense
cliqueGeometric	Synthetic	2047	1.5M	Dense
web-Google	Real	875k	5.1M	Dense

For counting triangles, we perform experiments both with and without optimization. As we are performing self-join while counting triangles using relational queries, we can accelerate the performance with the optimizations discussed above. Hence we used maintaining duplicate E while partitioning both tables based on columns and encode using RLE. Spark-GraphX includes in its library an implementation of counting triangles. Table 3 shows the results of counting triangles using relational queries in DBMS with and without optimization and in Spark. Spark loses to DBMS whether we perform the optimization or not. However, with our optimized method, it takes almost half the time in DBMS than the normal method. Partitioning and maintaining duplicate E speeds up the performance of join operation and the queries execute faster.

In Table 4, we see the results of what happens when we eliminate duplicate results from each step of recursive queries. We try to find all the reachable vertices from a particular vertex up to path length 6. Using GROUP BY at each recursive step eliminates the duplicates and thus accelerates the performance. However, for tree10m data set, it performs better when we do not eliminate the duplicates. Here, the total number of paths generated in both methods are same for this data set, so doing an extra GROUP BY operation in each recursive step is taking time. As for other methods, if we perform the optimization with GROUP BY, it works better. The number of paths in each recursive step grows when there are many duplicate paths. We stop the program each time after 10 min and put “Stop” on the table. We see the other methods could not finish calculating all the paths when there is no GROUP BY optimization. Also, storage requirements can grow exponentially for dense graphs. However, Spark performs better in this cases. Path reachability requires a union operation at each iteration. As Spark computes the union operation in main memory, it has a significant advantage over DBMS to perform this kind of operations.

We also perform finding the connected components in both DBMS and Spark. For DBMS, we used recursive relational queries as mentioned in the previous section. For Spark-Graphx, it includes in its library an implementation of Connected Components similar to HCC, propagating minimum vertex-ids through the graph. From Table 5, we see the time it takes for both systems. Dataset tree10m takes most time for both systems as it does not have any connected component. So the program iteratively checks for connected components until there are no vertices left. However, for all data sets, DBMS performs better than Spark.

Table 2. Time to compute indegree (in seconds).

Data set	DBMS	Spark
tree10m	1.4	8.4
wiki-vote	0.1	6.1
cliqueLinear	0.1	7.5
cliqueGeometric	0.2	8.5
webGoogle	0.6	20.4

Table 3. Time to enumerate triangles (in seconds).

Data set	DBMS (no optimization)	DBMS (with optimization)	Spark
tree10m	6.5	3.1	59.2
wiki-vote	0.9	0.4	16.4
cliqueLinear	5.1	2.3	27.9
cliqueGeometric	41.8	22.9	51.7
webGoogle	7.0	2.1	142.7

Table 4. Path reachability for path length 6: pushing GROUP BY (in seconds).

Data set	Not pushing GROUP BY	Total paths	pushing GROUP BY	Total paths	Spark
tree10m	11.7	126	48.9	126	2.8
wiki-vote	Stop	-	2.3	2316	2.0
cliqueLinear	Stop	-	3.4	145	1.4
cliqueGeometric	Stop	-	4.0	1025	1.7
webGoogle	39.4	2898192	12.3	7083	2.3

Table 5. Time to compute connected components (in seconds).

Data set	DBMS	Spark
tree10m	45.1	649.7
wiki-vote	4.3	12.9
cliqueLinear	2.2	21.7
cliqueGeometric	2.5	22.4
webGoogle	30.2	60.3

For adjacency matrix multiplication, we performed all the optimization techniques. As it is a matrix-matrix multiplication, the result will also be a matrix. Table 6 shows the results of performing the multiplication using relational queries in DBMS and Spark-GraphX. We can accelerate the performance using optimizations discussed above. When we multiply consecutively, the time difference between two methods become more significant. Using GROUP BY eliminates duplicates from next multiplication and matrix size becomes smaller. Also, partitioning based on columns improves the join performance. For Spark-GraphX, we slightly modify the implementation of transitive closure in its library to perform the multiplication as many times as we need. DBMS performs better than Spark-GraphX in every cases. We stop the program each time after 10 min.

Table 7 shows output for some queries discussed above. For each data set, it shows the maximum *indegree* and expected number of paths to be generated. Fig. 1 shows the maximum number of *indegree* and *outdegree* generated per recursion depth of recursive queries while calculating the adjacency matrix multiplication.

Table 6. Adjacency matrix multiplication time (in seconds).

Data set	Multiply once			Multiply 2 times		
	Without optimization	With optimization	Spark	Without optimization	With optimization	Spark
tree10m	6.7	7.2	181.1	12.2	18.4	408.3
wiki-vote	2.3	1.9	25.9	68.5	8.9	68.3
cliqueLinear	31.7	3.9	73.8	Stop	7.6	138.2
cliqueGeometric	358.1	18.2	486.4	Stop	36.3	Stop
webGoogle	26.1	23.2	535.2	286.9	120.5	Stop

Table 7. Exploring graphs.

Data set	Edges	Max <i>Indegree</i>	Expected paths
tree10m	10M	1	10M
wiki-vote	103.6k	457	4.54M
cliqueLinear	10M	311	2333.79M
cliqueGeometric	1.5M	1024	1224.34M
webGoogle	5.1M	6326	60.68M

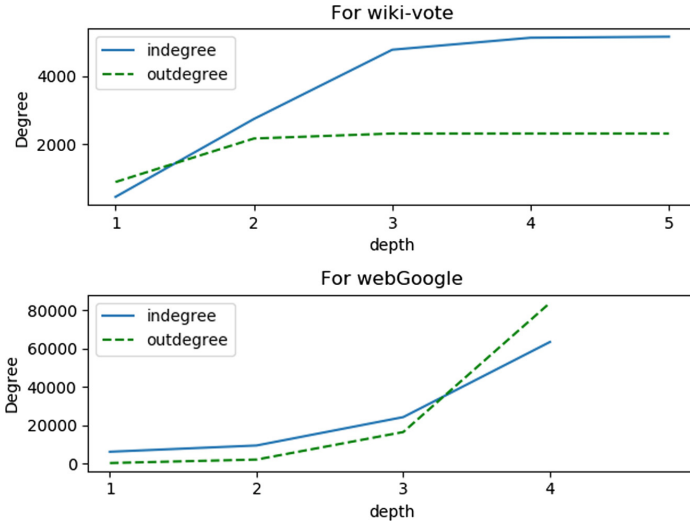


Fig. 1. Maximum number of *indegree* and *outdegree* per adjacency matrix multiplication for webGoogle and wiki-vote data sets

5 Related Work

Research on recursive queries and parallel computation is extensive, especially in the context of deductive databases [1, 3, 18, 21, 23] or adapting deductive database techniques to relational databases [11, 12, 22]. There is significant theoretical work on recursive query computation in the Datalog language [10, 16]. More recently, [18] proposed a hybrid approach to query graphs with a language similar to SPARQL, maintaining data structures with graph topology in main memory and storing graph edges as relational tables like we do. This work considers reachability queries (i.e. transitive closure), but not their optimization on modern DBMSs. There is research [13] on comparing row, columnar and array DBMSs for higher data volume that shows columnar DBMS has advantage over row and array DBMSs in many cases.

Although numerous applications are related to graphs, querying from large graphs using relational queries stored on a DBMS has not received much attention. In [24], the authors revisited the issue of how RDBMS can support graph processing at the SQL level. To support graph processing, they proposed new relational algebra operations. Another recent work have focused on retrieving paths using a path query language from a network graph where they present the Nepal query language [6]. In [20], they presented a formalization of graph pattern matching for Gremlin queries. Gremlin [15] is a graph traversal language and machine, provides a common platform for supporting any graph computing system. The Boost Graph Library (BGL) [17] provides some general purpose graph classes. The graph algorithms in BGL [17] includes most popular graph searching and shortest path algorithms. Stanford Network Analysis Platform

(SNAP) [9] is a general purpose, high-performance system, and graph mining library that easily scales to massive networks with billions of nodes and edges. Other recent works on graphs offer vertex-centric query interface to express many graph queries [5] and study compression techniques for indexing regular path queries in graph languages [19].

6 Conclusions

We can perform many queries in a very short time on a graph stored on a DBMS. Large-scale graphs can be quickly loaded and analyzed in parallel systems within a very short time. We represented the graph in terms of database perspective and stored them as a form of triplets. SQL queries are good enough for common graph problems like reachability, vertex degree, triangles, isolated vertices, expected paths from the graph, adjacency matrix multiplication, and so on. The queries reveal many interesting properties and patterns from the graph. We proposed several optimization methods on the queries. Optimizing the query performance helps to execute the queries faster than usual. Our experimental results show that our queries perform better than Spark-GraphX in most cases. However, there are some graph problems that cannot be solved with SQL queries like detecting planarity, traveling salesman problem, finding all cliques and hierarchical graph summarization.

For future work we have the following: detecting shortest cycles from a graph, counting maximal cliques meaning the highest number of vertices connected each other, showing actual paths when possible (for low number of paths), parallel speed up meaning how the queries perform when we vary the number of machines and discovering more complex patterns beyond paths. Moreover, we also plan to optimize the algorithms in Spark.

References

1. Abiteboul, S., Hull, R., Vianu, V.: Foundations of Databases : The Logical Level, Facsimile edn. Pearson Education POD, Boston (1994)
2. Agrawal, R., Dar, S., Jagadish, H.: Direct and transitive closure algorithms: design and performance evaluation. *ACM TODS* **15**(3), 427–458 (1990)
3. Bancilhon, F., Ramakrishnan, R.: An amateur’s introduction to recursive query processing strategies. In: *Proceedings of ACM SIGMOD Conference*, pp. 16–52 (1986)
4. Cabrera, W., Ordonez, C.: Scalable parallel graph algorithms with matrix–vector multiplication evaluated with queries. *Distrib. Parallel Databases* **35**(3–4), 335–362 (2017)
5. Jindal, A., Rawlani, P., Wu, E., Madden, S., Deshpande, A., Stonebraker, M.: VERTEXICA: your relational friend for graph analytics!. *Proc. VLDB Endow.* **7**(13), 1669–1672 (2014)
6. Johnson, T., Kanza, Y., Lakshmanan, L.V.S., Shkapenyuk, V.: Nepal: a path query language for communication networks. In: *Proceedings of the 1st ACM SIGMOD Workshop on Network Data Analytics, NDA 2016*, pp. 6:1–6:8 (2016)

7. Kang, U., Tsourakakis, C.E., Faloutsos, C.: PEGASUS: a peta-scale graph mining system implementation and observations. In: Proceedings of the 2009 Ninth IEEE International Conference on Data Mining, ICDM 2009, pp. 229–238 (2009)
8. Lamb, A., et al.: The Vertica analytic database: C-store 7 years later. *Proc. VLDB Endow.* **5**, 1790–1801 (2012)
9. Leskovec, J., Krevl, A.: SNAP datasets: stanford large network dataset collection, June 2014. <http://snap.stanford.edu/data>
10. Libkin, L., Wong, L.: Incremental recomputation of recursive queries with nested sets and aggregate functions. In: Cluet, S., Hull, R. (eds.) DBPL 1997. LNCS, vol. 1369, pp. 222–238. Springer, Heidelberg (1998). https://doi.org/10.1007/3-540-64823-2_13
11. Mumick, I., Finkelstein, S., Pirahesh, H., Ramakrishnan, R.: Magic Conditions. *ACM TODS* **21**(1), 107–155 (1996)
12. Mumick, I., Pirahesh, H.: Implementation of magic-sets in a relational database system. In: ACM SIGMOD, pp. 103–114 (1994)
13. Ordonez, C., Cabrera, W., Gurram, A.: Comparing columnar, row and array DBMSs to process recursive queries on graphs. *Inf. Syst.* **63**, 66–79 (2016)
14. Pavlo, A., et al.: A comparison of approaches to large-scale data analysis. In: Proceedings of ACM SIGMOD Conference, pp. 165–178 (2009)
15. Rodriguez, M.A.: The Gremlin graph traversal machine and language (invited talk). In: Proceedings of the 15th Symposium on Database Programming Languages, DBPL 2015, pp. 1–10 (2015)
16. Seshadri, S., Naughton, J.: On the expected size of recursive Datalog queries. In: Proceedings of ACM PODS Conference, pp. 268–279 (1991)
17. Siek, J., Lee, L.Q., Lumsdaine, A.: Boost c++ libraries. <https://www.boost.org/>
18. Sakr, S., Elnikety, S., He, Y.: Hybrid query execution engine for large attributed graphs. *Inf. Syst.* **41**, 45–73 (2014)
19. Tetzl, F., Voigt, H., Paradies, M., Lehner, W.: An analysis of the feasibility of graph compression techniques for indexing regular path queries. In: Proceedings of the Fifth International Workshop on Graph Data-management Experiences & Systems, GRADES 2017, pp. 11:1–11:6 (2017)
20. Thakkar, H., Punjani, D., Auer, S., Vidal, M.-E.: Towards an integrated graph algebra for graph pattern matching with Gremlin. In: Benslimane, D., Damiani, E., Grosky, W.I., Hameurlain, A., Sheth, A., Wagner, R.R. (eds.) DEXA 2017 Part I. LNCS, vol. 10438, pp. 81–91. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-64468-4_6
21. Ullman, J.: Implementation of logical query languages for databases. *ACM Trans. Database Syst.* **10**(3), 289–321 (1985)
22. Valduriez, P., Boral, H.: Evaluation of recursive queries using join indices. In: Expert Database Systems, pp. 271–293 (1986)
23. Youn, C., Kim, H., Henschen, L., Han, J.: Classification and compilation of linear recursive queries in deductive databases. *IEEE TKDE* **4**(1), 52–67 (1992)
24. Zhao, K., Yu, J.X.: All-in-one: graph processing in RDBMSs revisited. In: Proceedings of the 2017 ACM International Conference on Management of Data, SIGMOD, pp. 1165–1180 (2017)

Biological Knowledge Discovery from Big Data (BIOKDD)

9th International Workshop on Biological Knowledge Discovery from Big Data, BIOKDD 2018

In recent years, there has been a rapid development of biological technologies producing more and more biological data, i.e., data related to biological macromolecules (DNA, RNA, and proteins). The rise of Next Generation Sequencing (NGS) technologies, also known as high-throughput sequencing technologies, has contributed actively to the deluge of these data. In general, these data are big, heterogeneous, complex, and distributed in databases all over the world. Analyzing biological big data is a challenging task, not only because of its complexity and its multiple and numerous correlated factors, but also because of the continuous evolution of our understanding of the biological mechanisms. Classical approaches of biological data analysis are no longer efficient and produce only a very limited amount of information, compared to the numerous and complex biological mechanisms under study. This leads to the necessity to adopt new computer tools and develop new *in silico* high performance approaches to support us in the analysis of biological big data and, hence, to help us in our understanding of the correlations that exist between, on the one hand, structures and functional patterns in biological macromolecules and, on the other hand, genetic and biochemical mechanisms. Biological Knowledge Discovery from Big Data (BIOKDD) was a response to these new trends.

This workshop contains fascinating papers that deal with BIOKDD.

Mourad Elloumi




Organization

Program Committee

Mourad Elloumi (Chair)	University of Tunis, Tunisia
Irena Rusu	University of Nantes, France
Davide Verzotto	University of Pisa, Italy
Nadia Pisanti	University of Pisa, Italy
Dominique Lavenier	IRISA, CNRS, Rennes, France
Emanuel Weitschek	UniNettuno University, Italy
Daisuke Kihara	Purdue University, USA
Bhaskar DasGupta	University of Illinois at Chicago, USA
Vladimir Makarenkov	University of Québec, Canada
Ronnie Alves	Instituto Tecnológico Vale D.S., Brasil
Matteo Comin	University of Padova, Italy
Abdelouahid Lyhyaoui	Abdelmalek Essaâdi University, Morocco
Adrien Goëffon	University of Angers, France
Béatrice Duval	University of Angers, France
Tolga Can	Middle East Technical University, Turkey
Maad Shatnawi	Higher Colleges of Technology, UAE
Mirto Musci	University of Pavia, Italy
Malik Yousef	Zefat Academic College, Israel
Evangelos Theodoridis	Dynamo, London, UK
Giosuè Lo Bosco	University of Palermo, Italy
Tomas Flouri	University College London, UK
Solon Pissis	King's College, London, UK



New Modeling Ideas for the Exact Solution of the Closest String Problem

Marcello Dalpasso¹  and Giuseppe Lancia²  

¹ DEI, University of Padova, Via Gradenigo 6, 35131 Padova (PD), Italy
marcello.dalpasso@unipd.it

² DMIF, University of Udine, Via delle Scienze 206, 33100 Udine (UD), Italy
giuseppe.lancia@uniud.it

Abstract. In this paper we consider the exact solution of the closest string problem (CSP). In general, exact algorithms for an NP-hard problem are either branch and bound procedures or dynamic programs. With respect to branch and bound, we give a new Integer Linear Programming formulation, improving over the standard one, and also suggest some combinatorial lower bounds for a possible non-ILP branch and bound approach. Furthermore, we describe the first dynamic programming procedure for the CSP.

Keywords: Closest String Problem · Dynamic programming
Parametrized complexity · Integer Linear Programming

1 Introduction

The Closest String is a very well known computational biology problem. In this problem we are given a set S of n related genomic sequences, which for instance, might correspond to homologous genes in n different species. The goal is to determine a new sequence s which represents them all (a *consensus*) and that can then be used, e.g., as a query over genomic data bases. When the input set contains an overwhelming majority of sequences from a particular species (most notably human), a consensus which minimizes the average difference from S will be biased towards that species as well. In order to obtain an unbiased consensus, the objective can be set to minimizing the *maximum* difference from s to any string in S . This is the Closest String Problem (CSP).

The CSP is NP-hard, even for a binary alphabet [10]. The first theoretical results and algorithms were proposed in [1] which gave an ILP model and a randomized approximation algorithm based of LP-rounding. The ILP model was then utilized in a string of papers on CSP as well as other, related, problems [5–7]. The main results obtained on these problems are PTAS based on the ILP formulation and Randomized Rounding, coupled with random sampling [6].

Heuristic approaches, but with no performance guarantee, were also studied for the CSP (see, e.g., [8]).

As far as exact approaches are concerned, many of them were based on ILP formulations [9], sometimes parametrized by some of the problem parameters (such as $n, m, |\Sigma|$, see [2, 3]). Moreover, an exact method based on constraint programming [4] was also investigated.

Our Contribution. In this paper we consider the exact solution of the CSP and start from its most effective ILP formulation in the literature. We show how to improve over this formulation by giving a new, equivalent, formulation with a much smaller number of variables. Our formulation is particularly well suited for the case in which the input sequences are *aligned* sequences, i.e., rows of a multiple sequence alignment. This is a standard condition for bioinformatics inputs but it was not exploited by the previous ILP models.

Our second contribution is a dynamic programming procedure which, besides being interesting on its own, can also be used to prove in a new way an already known fact, i.e., that CSP is polynomial for constant n .

As this is a work in progress, the computational experiments are missing from this first version and are deferred to a full journal version of the paper.

2 Problem and Notation

Let $\Sigma = \{\sigma_1, \dots, \sigma_{|\Sigma|}\}$ be an alphabet. The Hamming distance of two strings $s, t \in \Sigma^m$ is defined as

$$d(s, t) = |\{i : s[i] \neq t[i]\}|$$

In the CSP, we are given n strings $S = \{s_1, \dots, s_n\}$, each of length m . We want to find a string $s \in \Sigma^m$ such that

$$\max_{i=1, \dots, n} d(s, s_i) \leq \max_{i=1, \dots, n} d(t, s_i) \quad \forall t \in \Sigma^m$$

If we define the ball of center s and radius r as

$$B(s, r) := \{x \in \Sigma^m : d(x, s) \leq r\}$$

then the CSP calls for determining a center such that there is a ball of center s which contains S and has the smallest possible radius.

For each string i and character $\sigma \in \Sigma$, let D_σ^i be the set of positions where $s_i[j]$ is not σ , i.e., $D_\sigma^i = \{j : s_i[j] \neq \sigma, j = 1, \dots, m\}$.

Data Preprocessing and Canonical Form. We might assume that the input is in the form of a matrix of n rows and m columns. Each row represents a string. Then we could preprocess the input according to the following easily proved observations

1. We can delete identical rows.
2. We can permute the columns however we want.

3. We can delete columns in which only one symbol of the alphabet appears (as we know that the optimal solution will certainly have that symbol in that position).
4. Let N be the maximum, over all columns, of the number of symbols that appear in one column. Then we can replace Σ with the subset of its first N characters. Namely, assume that in a column the different symbols π_1, \dots, π_K appear (from top to bottom). Then, we replace each occurrence of π_i with σ_i , for $i = 1, \dots, K$.

By applying the above rules, each input matrix can be put in a *canonical form*, namely:

1. All rows are different.
2. Each column i contains the symbols $\sigma_1, \dots, \sigma_{k(i)}$, with $k(i) \geq 2$.
3. The matrix is made of consecutive *blocks*, where each block is a sequence of identical columns. If b is the number of blocks, then by $l(i)$, for $i = 1, \dots, b$, we denote the number of columns in block i , and by $d(i)$ we denote the number of distinct rows in block i (notice that these are substrings of the complete rows of the matrix).
4. The j -th distinct row of a block i is a string of length $l(i)$ made entirely of the character σ_j .

For example, consider the following input matrix:

```

a b a c a a b e a d d
b c a c e d d d e e a
a b a c a a b e a d d
a b a c a a b e a d d
b b a c a c c d e d d
b c a c e a d d e e a
    
```

When put in canonical form the matrix becomes

```

a a a | a a a | a a
b b b | b b b | b b
a a a | a a a | a a
b b b | a a a | c c
    
```

In this matrix there are 3 blocks, with $l(1) = 3$, $l(2) = 4$, $l(3) = 2$. Furthermore, $d(1) = d(2) = 2$ and $d(3) = 3$. The original alphabet $\{a, b, c, d, e\}$ has become $\{a, b, c\}$.

This type of data preprocessing is particularly important for genomic inputs, which, rather than raw DNA sequences s_1, \dots, s_n corresponding to homologous genes (which normally have different lengths, due to indels and mutations), are *aligned* sequences s'_1, \dots, s'_n , all of the same length and possibly containing gaps (i.e., they are defined over the extended alphabet $\Sigma \cup \{-\}$). Assume for the sake of example that the sequences `cattgccat`, `cattaggat`, `attaccg`, `gattccc`, `ccat` are related. The first step to highlight this relation would be to put them in a multiple alignment, so they become the rows of a matrix such as

```

c a t t - g c c - a t
c a t t a g - - g a t
- a t t a - c c g - -
g a t t - c c c - - -
- - - - - c c - a t

```

The goal of an alignment is to retrieve common, conserved, motifs, such as **att** above. In good alignments there are a few, but quite long, motifs and we are going to exploit this fact in the ILP formulation. If we preprocess the multiple alignment, by permuting the columns as 1, 6, 5, 9, 7, 8, 10, 11, 2, 3, 4, we obtain an input of 5 blocks over an alphabet of 3 symbols. If we use $\{1, 2, 3\}$ as the alphabet for the preprocessed matrix, the matrix has become

```

1 1 | 1 1 | 1 1 | 1 1 | 1 1 1
1 1 | 2 2 | 2 2 | 1 1 | 1 1 1
2 2 | 2 2 | 1 1 | 2 2 | 1 1 1
3 3 | 1 1 | 1 1 | 2 2 | 1 1 1
2 2 | 1 1 | 1 1 | 1 1 | 2 2 2

```

3 Strengthening the Best ILP Formulation

Let us start by recalling the basic, but most effective in the literature, ILP model (see [9] for a comparison of ILPs for CSP). For each $1 \leq j \leq m$ and $\sigma \in \Sigma$, denote by $x_{j,\sigma}$ a 0–1 variable which is 1 if character j of solution is σ . Then we have the following formulation

$$\text{Minimize} \quad r \tag{1}$$

$$s.t. \quad \sum_{\sigma} x_{j,\sigma} = 1 \quad \forall j = 1, \dots, m \tag{2}$$

$$\sum_{\sigma} \sum_{j \in D_{\sigma}^i} x_{j,\sigma} \leq r \quad \forall i = 1, \dots, n \tag{3}$$

$$x_{i,\sigma} \in \{0, 1\}^{m \times |\Sigma|} \tag{4}$$

We are going to strengthen this model by reducing the number of its variables (henceforth making it applicable to instances where this number would have otherwise been too high) and by proposing some valid inequalities to cut-off fractional solutions of the LP relaxation.

We will exploit our preprocessing of the input matrix into blocks. In order to describe the idea, let us first start with the simplest possible case, i.e., an input matrix consisting of only one block i.e., of m identical columns (clearly, in this case the solution could be found immediately by simple combinatorial reasoning with no need for an ILP model, but the case is preparatory for the more general case, coming next). Notice that, in this case, the number of rows is $n = |\Sigma|$, since (for the matrix in canonical form) each column contains the symbols $\{\sigma_1, \dots, \sigma_n\}$. In particular, row i is the string $\sigma_i \sigma_i \dots \sigma_i$. Then, instead of having $m|\Sigma|$ variables (i.e., mn variables in this case), we introduce only $|\Sigma|$

integer variables y_1, \dots, y_n where y_i represents the Hamming distance of the solution to row i (i.e., the number of letters $\neq \sigma_i$ in the solution).

The new model is

$$\text{Minimize} \quad r \tag{5}$$

$$\text{s.t.} \quad y_i \leq r \quad \forall i = 1, \dots, |\Sigma| \tag{6}$$

$$\sum_{i=1}^n (m - y_i) = m \tag{7}$$

$$y_i \geq 0, \text{ integer} \quad \forall i = 1, \dots, |\Sigma| \tag{8}$$

Notice that in constraints (7), the number $(m - y_i)$ is equal to the number of symbols σ_i in the solution. Hence, the values y_i identify the solution string, since for this particular type of input only the number of occurrences of each symbols matters, and not the order of the symbols.

Let us now turn to the general case, in which the input consists of b blocks. For each block j we have integer variables y_{ij} , with $i = 1, \dots, d(j)$. The meaning of y_{ij} is to count the Hamming distance of the substring of the solution string *consisting only of the columns in block j* from the i -th distinct row of block j . Notice that $l(j) - y_{ij}$ is the number of symbols σ_i which appear in the solution string in the positions corresponding to the columns of block j .

For each row $r = 1, \dots, n$, and block $j = 1, \dots, b$, let us denote by $\pi(r, j)$ the index of the symbol appearing in row r of block j (i.e., row r is made entirely of $\sigma_{\pi(r, j)}$).

Then we get to the new general model for CSP, which is

$$\text{Minimize} \quad r \tag{9}$$

$$\text{s.t.} \quad \sum_{j=1}^b y_{\pi(i, j)} \leq r \quad \forall i = 1, \dots, n \tag{10}$$

$$y_{ij} \leq l(j) \quad \forall i = 1, \dots, d(j) \tag{11}$$

$$\sum_{i=1}^{d(j)} (l(j) - y_{ij}) = l(j) \quad \forall j = 1, \dots, b \tag{12}$$

$$y_i \geq 0, \text{ integer} \quad \forall i = 1, \dots, n \tag{13}$$

If we consider the simple example given in the previous section, the basic ILP model (1) applied to the rows of the multiple alignment has $m|\Sigma|$ variables $x_{j, \sigma}$, i.e., 55 variables. The new model, applied to the preprocessed alignment, has $3 + 2 + 2 + 2 + 2 = 11$ variables y_{ij} . Alady on this toy example the reduction in size of the ILP can be appreciated, but it is clearly much bigger for real-life sequence alignments.

3.1 Valid Inequalities

Still in the spirit of improving over the basic ILP (1)–(4), we describe some valid inequalities that could be added to it to strengthen its LP-relaxation.

Let $D = d(s_a, s_b) = \max_{i,j} d(s_i, s_j)$. Then the optimal solution value can never be smaller than $D/2$. In fact, if s^* is the optimal solution, by triangle inequality it must be

$$D = d(s_a, s_b) \leq d(s_a, s^*) + d(s_b, s^*) \leq 2 \max_i d(s_i, s^*)$$

Hence, since D must be integer, we can add the following constraint to the model:

$$r \geq \left\lceil \frac{\max_{i,j} d(s_i, s_j)}{2} \right\rceil \tag{14}$$

This constraint can cut-off optimal LP-relaxation solutions of (1)–(4). Consider the following example: $s_1 = 000$, $s_2 = 111$, the optimal solution value is 2, and $D = \max_{i,j} d(s_i, s_j) = 3$, so that $\lceil \frac{D}{2} \rceil = 2$. An optimal solution of the LP-relaxation of (1)–(4) is x^* with $x_{j,\sigma}^* = 0.5$ for each $j = 1, 2, 3$ and $\sigma = 0, 1$. The optimal LP-relaxation value is $r^* = 1.5$ but this value violates the constraint $r \geq 2$.

Similarly to the above, single, inequality, we can derive valid inequalities as follows. Since the Hamming distance obeys triangle inequality, if s is any feasible solution, it must be

$$d(s, s_i) + d(s, s_k) \geq d(s_i, s_k) \tag{15}$$

for all $1 \leq i < k \leq n$. The constraints (15) can be expressed by the following linear inequalities:

$$\sum_{\sigma} \left(\sum_{j \in D_{\sigma}^i} x_{j,\sigma} + \sum_{j \in D_{\sigma}^k} x_{j,\sigma} \right) \geq d(s_i, s_k) \quad \forall 1 \leq i < k \leq n$$

which can also be written as

$$\sum_{\sigma} \sum_{j \in D_{\sigma}^i \cup D_{\sigma}^k} l_{j,\sigma} x_{j,\sigma} \geq d(s_i, s_k) \quad \forall 1 \leq i < k \leq n$$

where $l_{j,\sigma} = 1$ if $j \in D_{\sigma}^i \Delta D_{\sigma}^k$ and $l_{j,\sigma} = 2$ if $j \in D_{\sigma}^i \cap D_{\sigma}^k$.

We now describe an exponential-size class of inequalities which could, however, be separated in polynomial time.

Assume to have a feasible solution, of value D (the smaller, the better). Take any input sequence s_i and look at any subset A consisting of D of its characters. The optimal solution cannot be different from all of them, or else it would have cost $\geq D$. But we already have a solution of value D , so we look for a solution of value $\leq D - 1$. Hence, we get that

$$\sum_{j \in A} x_{j, s_i[j]} \geq 1 \tag{16}$$

is a valid inequality to add for searching a solution of value better than D . Notice that there are $n \times \binom{m}{D}$ such possible inequalities (exponentially many, since D can be proportional to m).

In general, the *separation problem* for a class \mathcal{C} of inequalities is the following. Given a point $x^* \in \mathbb{R}^n$ find an inequality $ax \geq b$ in \mathcal{C} such that $ax^* < b$, or conclude that no such violated inequality exists. If \mathcal{C} has exponential size, it is fundamental that the separation problem can be solved in polynomial time for an ILP model based on \mathcal{C} to be effectively used. We now show how the inequalities (16) can be in fact separated in polynomial time.

Consider each sequence s_i one at a time. For $j = 1, \dots, m$, define $a_j = x_{j, s_i}^{* [j]}$. Now, sort the a_j in increasing order, $a_{p(1)} \leq \dots \leq a_{p(m)}$, and let A be the set of the first D of them (i.e., $A = \{p(1), \dots, p(D)\}$). Then A achieves the minimum possible sum of x^* , and if this sum is < 1 , we have found a violated inequality (16), otherwise, there are no violated inequalities for s_i and we move to s_{i+1} .

We thus proved:

Theorem 1. *The inequalities (16) can be separated in polynomial time (namely $O(nm \log m)$).*

4 Dynamic Programming and Parametrized Complexity

Since for any string s its maximum distance from set S is $\leq m$, we are really interested in strings whose maximum distance is $\leq m - 1$. We can then use Dynamic Programming. For $0 \leq d_i \leq m - 1$ ($i = 1, \dots, n$) and $1 \leq l \leq m$, define

$$W(d_1, \dots, d_n, l) \in \{\text{TRUE}, \text{FALSE}\}$$

to be TRUE if there exists a string $s^* \in \Sigma^m$ such that $d(s^*[1, \dots, l], s_i[1, \dots, l]) \leq d_i$ for each $i = 1, \dots, n$. With this notation, we are eventually interested in finding the minimum D such that $W(D, D, \dots, D, m) = \text{TRUE}$.

For $\sigma, \tau \in \Sigma$, define $\Delta(\sigma, \tau) \in \{0, 1\}$ to be 0 if $\sigma = \tau$ and 1 otherwise. Then the DP recursion is

$$W(d_1, \dots, d_n, l) = \bigvee_{\sigma \in \Sigma} W(d_1 - \Delta(\sigma, s_1[l]), \dots, d_n - \Delta(\sigma, s_n[l]), l - 1) \tag{17}$$

To see this, consider the subproblem relative to all prefixes of length l and let $x \in \Sigma^l$, be its solution. If $x[l] = \sigma$, for x to have distance d_i from an l -prefix of a string, x must have distance at most d_i in the first $l - 1$ positions for all strings i that end in σ , while for strings that end in something $\neq \sigma$, x must have distance at most $d_i - 1$ in the first $l - 1$ positions.

As for the base-case, there are two of them. The first base-case is when $\min_i d_i = 0$. Let k be such that $d_k = 0$. Then $W(d_1, \dots, d_n, l) = \text{TRUE}$ if and only if for all $i = 1, \dots, n$ it is $d(s_i[1, \dots, l], s_k[1, \dots, l]) \leq d_i$. The second base-case is when $l \leq \min_i d_i$, in which case $W(d_1, \dots, d_n, l) = \text{TRUE}$. Note that since in the DP recursion, l always decreases, while sometimes at least one of the d_i 's decreases, either the first or the second base-case are eventually bound to happen.

The above Dynamic Program requires to fill a table of m^{n+1} entries. At each entry we must consider $|\Sigma|$ symbols, so that the total complexity is

$$O(|\Sigma|m^{n+1})$$

To reconstruct the solution, we can set $s^*[l]$ to be any σ which makes true the \bigvee in (17).

A few simple considerations can be made to speed-up the solution of the DP. First, notice that the $\bigvee_{\sigma \in \Sigma}$ in (17) can be replaced by $\bigvee_{\sigma \in \Sigma(l)}$, with $\Sigma(l) = \{s_i[l], i = 1, \dots, n\}$. Then, notice that if at any step we compute an entry $W(d_1, \dots, d_n, l) = \text{TRUE}$, then we can set to TRUE all entries $W(d'_1, \dots, d'_n, l)$ with $d'_i \geq d_i$ for each i . (In effect, entries for which $l \leq \min_i d_i$ could be omitted since their value is always true. So the memory required is $2^n + 3^n + \dots + m^n < m^{n+1}$).

Finally, note that to compute the optimal value, i.e.,

$$v = \min\{D : W(D, \dots, D, m) = \text{TRUE}\}$$

it is sensible to proceed bottom-up, and not top-down or with binary search. I.e., we start with $v = 0$, and

$$\text{while } W(v, \dots, v, m) = \text{FALSE}$$

we increase v . This way we avoid the computation of useless entries in the last column.

As far as the parametrized complexity of CSP is concerned, the above Dynamic Program, of complexity $O(|\Sigma|m^{n+1})$ gives a proof of the following result:

Theorem 2. *If n is a constant the CSP can be solved in polynomial time.*

5 Two Combinatorial Bounds

We conclude by describing two simple bounds which could be used in any non-ILP based branch-and-bound approach to the problem.

A first combinatorial bound, call it L_k , follows from the dynamic programming procedure. For any fixed k , the optimal value over k input strings is a lower bound to the global optimal. Hence, if we can compute L_k quickly, we can have bounds (the maximum of which is our best lower bound). When $k = 2$ the problem is trivial, since the optimal solution is just half of the Hamming distance of 2 strings. For $k = 3$, the D.P., in *linear time* for constant alphabet size, can solve the problem. Hence, we could, e.g., compute L_3 for all triples of input string and take the maximum. This, however can be expensive (there are $O(n^3)$ triples). We suggest a heuristic that works this way. Fix k , e.g., $k = 3$ or even 4 or 5. Then find a k -clique of large weight in the complete graph where each sequence is a node and the edges are weighted by Hamming distance. This

can be done via some fast and good clique heuristics from the literature. Then, compute the bound (i.e., solve the DP) for this clique. Repeat for other cliques.

We close with a final, very simple and quick to compute, combinatorial lower bound. At each position j , denote by S_σ^j , the subset of strings that have σ in position j . Then, call

$$d_{j,\sigma} := n - |S_\sigma^j|$$

i.e., $d_{j,\sigma}$ is the number of symbols different from σ in column j of the input. Now, define for each j

$$d_j := \min_{\sigma \in \Sigma} d_{j,\sigma}$$

and

$$L := \left\lceil \frac{\sum_{j=1}^m d_j}{n} \right\rceil$$

Then we get

Lemma 3. *L is a valid lower bound for CSP.*

The proof is the following: No matter what the solution puts in position j , it will be in conflict with at least d_j strings in that position. Added over all positions, this is the total number of conflicts (i.e., the sum of Hamming distances) of the solution to the input strings, and, dividing by n , we get that the average Hamming distance of any solution must be at least $\frac{1}{n} \sum_{j=1}^m d_j$. But since the maximum is at least the average (and we can also round to integer), we get the Lemma.

References

1. Ben-Dor, A., Lancia, G., Ravi, R., Perone, J.: Banishing bias from consensus sequences. In: Apostolico, A., Hein, J. (eds.) CPM 1997. LNCS, vol. 1264, pp. 247–261. Springer, Heidelberg (1997). https://doi.org/10.1007/3-540-63220-4_63
2. Chen, Z., Wang, L.: Fast exact algorithms for the closest string and substring problems with application to the planted (L, d) -Motif model. IEEE/ACM Trans. Comput. Biol. Bioinform. **8**(5), 1400–1410 (2011)
3. Gramm, J., Niedermeier, R., Rossmannith, P.: Exact solutions for closest string and related problems. In: Eades, P., Takaoka, T. (eds.) ISAAC 2001. LNCS, vol. 2223, pp. 441–453. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-45678-3_38
4. Kelsey, T., Kotthoff, L.: Exact closest string as a constraint satisfaction problem. Procedia Comput. Sci. **4**, 1062–1071 (2011)
5. Lanctot, J., Li, M., Ma, B., Wang, S., Zhang, L.: Distinguishing string selection problems. Inf. Comput. **185**, 41–55 (2003)
6. Li, M., Ma, B., Wang, L.: On the closest string and substring problems. J. ACM **49**(2), 157–171 (2002)
7. Ma, B.: A polynomial time approximation scheme for the closest substring problem. In: Giancarlo, R., Sankoff, D. (eds.) CPM 2000. LNCS, vol. 1848, pp. 99–107. Springer, Heidelberg (2000). https://doi.org/10.1007/3-540-45123-4_10

8. Mauch, H., Melzer, M.J., Hu, J.S.: Genetic algorithm approach for the closest string problem. In: Proceedings of the 2003 IEEE Bioinformatics Conference on Computational Systems Bioinformatics. CSB2003, pp. 560–561 (2003)
9. Meneses, C., Lu, Z., Oliveira, C., Pardalos, P.: Optimal solutions for the closest-string problem via integer programming. *INFORMS J. Comput.* **16**(4), 419–429 (2004)
10. Sim, J.S., Park, K.: The consensus string problem for a metric is NP-complete. In: Proceedings of the Annual Australasian Workshop on Combinatorial Algorithms (AWOCA), pp. 107–113 (1999)



Ensemble Clustering Based Dimensional Reduction

Loai Abdallah¹(✉) and Malik Yousef²

¹ Department of Information Systems,
The Max Stern Yezreel Valley Academic College, Jezreel, Israel
Loail984@gmail.com

² Department of Community Information Systems,
Zefat Academic College, 13206 Zefat, Israel
malik.yousef@gmail.com

Abstract. Distance metric over a given space of data should reflect the precise comparison among objects. The Euclidean distance of data points represented by a large number of features is not capturing the actual relationship between those points. However, objects of similar cluster both often have some common attributes despite the fact that their geometrical distance could be somewhat large. In this study, we proposed a new method that replaced the given data space to categorical space based on ensemble clustering (EC). The EC space is defined by tracking the membership of the points over multiple runs of clustering algorithms. To assess our suggested method, it was integrated within the framework of the Decision Trees, K Nearest Neighbors, and the Random Forest classifiers. The results obtained by applying EC on 10 datasets confirmed that our hypotheses embedding the EC space as a distance metric, would improve the performance and reduce the feature space dramatically.

Keywords: Decision trees · Ensemble clustering · Classification

1 Introduction

This research presents a new mapping procedure that replaces a given data space into categorical space based on ensemble clustering (EC).

The EC procedure will replace the traditional geometric distance between two given points and will be embedded in the machine learning classification algorithms Decision Trees (DT), k-nearest neighbors (KNN) and Random Forest (RF). The updated algorithm will be named DT-EC, KNN-EC and RF-EC respectively.

In the real world, many datasets are represented in a geometric space. This may not reveal the real similarity among the data points. Therefore, in this research, we propose a transformation function that transforms the original data space into other feature space, which is called EC feature space.

Unsupervised clustering algorithms use different metrics to depict the distance or similarity between two points such as Euclidean distance, correlation similarity, cosine similarity and other metrics. This similar function is used to assign points to clusters.

The EC space for a given dataset is based on a clustering process used frequently with various parameter values. The new space is depicted as cluster labels for each iteration.

This procedure of combining several clustering of a set of data points without being able to access the initial attributes is called cluster ensemble. Combination clustering is a much harder procedure than the combination of supervised categories. Topchy et al. [1] and Strehl et al. [2] tackled this problem by creating a general opinion that avoids a direct alternative to the correspondence issue. Modern studies have shown that consensus clustering can be discovered with the use of graph-based, information-theoretic or statistical procedures without clearly solving the label correspondence issue as stated in [3]. Other scientific general opinions were also considered in [4–6].

A clustering based learning method was recommended in [7]. In this study, many clustering algorithms are run to create various (unsupervised) models. The novice then makes use of the labeled data to predict labels for the whole clusters (within the forecasts that all points in a similar cluster have a similar label). By doing this, the algorithm forms several theories. The one that reduces the PAC-Bayesian bound is selected and utilized as the classifier. The authors believe that one or more of the clustering runs will generate a good classifier and that their algorithm will discover it. The technique we used is completely different from Derbeko et al.'s [7] in many ways, pertaining to the assumptions made. Our assumption was that only the equivalence classes are quite pure after running the clustering algorithm many times. In addition, our assumption was not based on the fact that one or more of the clustering runs produces will generate a good classifier, instead of that, the true classifier can be estimated rather effectively by some equivalence classes. Basically, the points, that is often part of the similar clusters in the various clustering iterations, will depict an equivalence class rather than single points and the distance metric outlined between these equivalence classes. Abdallah et al. [8, 9] developed a distance function based on ensemble clustering and used it in the framework of the k-nearest neighbor classifier and then they improve it by selecting the sampling for unsupervised data to be labeled by an expert.

Modern study by Yousef et al. [10] has applied EC classification by comparing it to two-class SVM and one-class classifiers used on sequence plant microRNA data. The results show that K- Nearest Neighbors-EC (KNN-EC) exceeds other techniques. The results emphasize that the EC procedure contributes to building a stronger model for its classification.

This paper is arranged as follows: in the next section, we will describe the ensemble clustering technique. Section 3 describes the decision trees classifier integrated with the ensemble clustering and the random forest classifier integrated with ensemble clustering. Experimental results of running the suggested classifiers on bioinformatics sequences data is shown in Sect. 4. Lastly, Sect. 5 presents the conclusion.

2 Ensemble Clustering Technique

In this section, we described the technique for constructing the new categorical space using the ensemble clustering technique.

Take a look at the following model. Let D be a set of data points in some space χ . Instances are assumed as i.i.d. which are spread based on some unknown fixed distribution ρ .

The given space does not always indicate the real comparison among the given data points. Nevertheless, it is recognized that points that belong to similar cluster have some common attributes, even though they differ according to the given space.

The major problem encountered using such a technique is that there is no recognized method for selecting the best clustering. Many attempts have been made to choose the ideal parameter values of the clustering algorithms in supervised and unsupervised settings, but up till now, no general solution to this problem has been found [12, 13].

Hence, we chose to run various clustering algorithms many times with various parameter values. The result of all these runs produces a cluster ensemble [14].

The clustering results are kept in a matrix known as the clusters matrix $C \in Mat_{N \times K}$, where K is the number of times the clustering algorithms were run. The i th row entails the cluster identities of the i th point in the different runs. This results in a new instance space $\chi_{cl} = \mathbb{Z}^K$ which comprises of the rows of the clusters matrix.

The following example illustrates this situation (Fig. 1):

Given the dataset in Fig. 1(a) the number of clusters is unknown, the clusters are not elliptic and the number of points within each cluster is unstable. Thus, running k-means with fixed k is not enough to capture the real structure of the data. Hence, the easiest way to capture the real structure of the data is to run various values of the parameter k. In the illustration for example, we run $k = 2, 3, 4, 5$ as described in Fig. 1(b–e). Figure 1(f) describes the equivalence classes that we get from the k-means of different iterations.

2.1 Decision Trees Classifier

The Decision Trees (DT) model was updated to include the new ensemble distance space and we evaluate the contribution of the EC space in improving the performance.

Decision trees classifier is one of the most well-known approaches for classification. They are widely used due to their good learning capabilities and simple interpretation. One can describe the DT model as a set of logical rule.

The decision-tree algorithms are many in number. The notable ones comprise:

1. ID3 (Iterative Dichotomiser 3)
2. C4.5 (successor of ID3)
3. CART (Classification And Regression Tree)
4. MARS: extends decision trees to handle numerical data better.

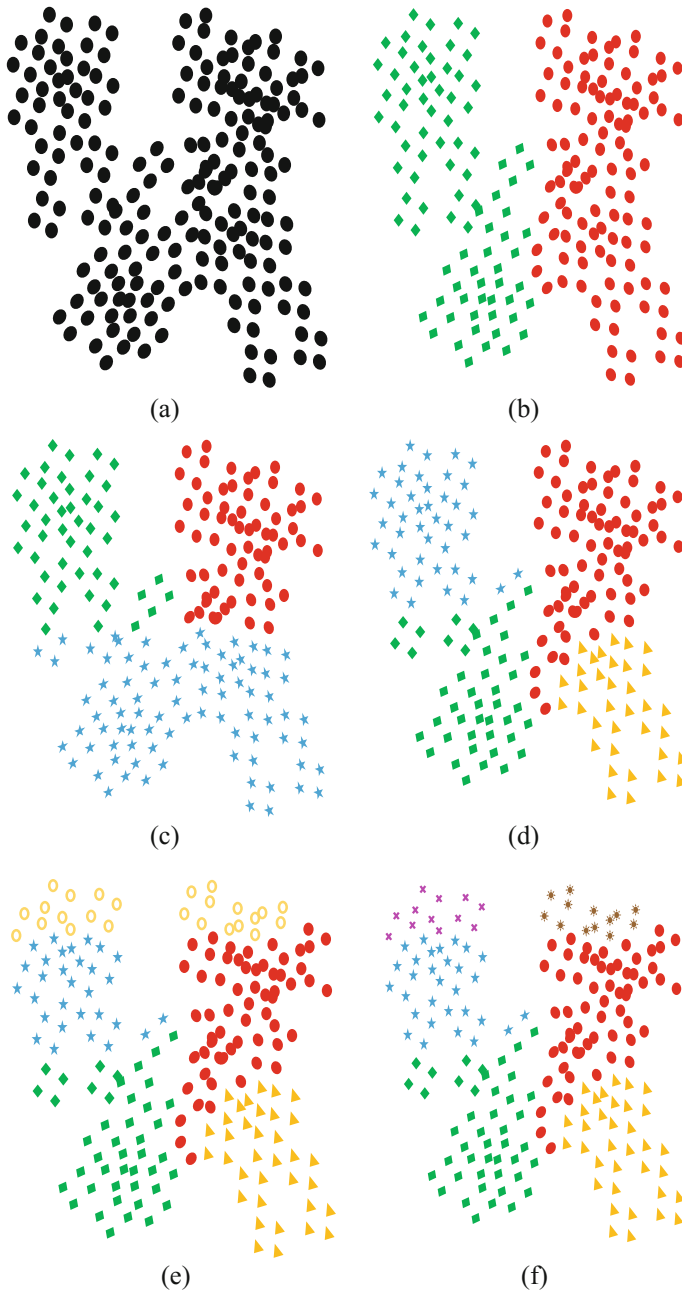


Fig. 1. Example of Euclidean space and Clustering based space methods to distinguish the points into clusters. (a) The original data, (b) k -means with $k = 2$, (c) k -means with $k = 3$, (d) k -means with $k = 4$, (e) k -means with $k = 5$. (f) the equivalence classes.

The DT model mainly consist of two main recursive stages for creating the decision tree.

- (1) Evaluation of splits for every single attribute and choosing of the best split.
- (2) Creation of partitions using the best split.

After the overall best split is determined, partitions can be made by simply applying the splitting criterion to the data. The difficulty lies in determining the best split for each and every attribute. In order to identify the best splitting attribute, there are three common splitting scheme: *Entropy*, *Gini*, *Classification error*.

Let D be a set of labeled points. Let that the data contain n classes, then:

$$Entropy(D) = - \sum_{j=1}^m p_j \log_2 p_j$$

$$Gini(D) = 1 - \sum_{j=1}^m p_j^2$$

$$Classificationerror(D) = 1 - \max_j [p_j].$$

Where p_j is the relative frequency of class j in D .

According to these measurements, we can estimate the information gained for each and every attribute, and then identify the best splitting attribute. This procedure will be repeated until some stopping criteria are satisfied.

This algorithm has several important advantages further than its classification performance. It is inexpensive to construct, the runtime to classify new unclassified objects is very efficient, and it does not need to store the training data point in order to make the classification procedure.

2.2 Decision Trees Using the Ensemble Clustering

Our approach integrates the ensemble clustering technique with the paradigm of the decision trees classifier. Firstly, in order to achieve this mission, we run different clustering algorithms several times. Secondly, we construct the clustering matrix by collecting all the clustering results. Finally, we run the standard decision trees classifier on a categorical data (i.e., the data within the clustering matrix).

Using this approach, we prove that the similarity between the points in the new categorical space is better than the original given space.

2.3 Random Forest Classifier

The Random Forest (RF) or the random decision forest classifier was also used to estimate the performance of this algorithm based on the new representation space on the ensemble clustering. To this end, we will now give a short overview of the random forest classifier.

Random Forests constructs many classification trees. Each tree was built according to a subset of the features of the data and subset of the data. At each iteration, the algorithm randomly selects a subset of features, and construct the tree based on these features. Repeats this procedure several times and bring a decision forest to us. To categorize a new object from an input vector, the algorithm put down the input vector in each of the trees in the forest. Each tree presents a classification result, and this gives us the tree votes for that class. The forest selects the classification that has the most votes (total number of trees in the forest). Breiman [11] explains that the forest error rate is based on two things:

- The relationship between any two trees in the forest.
- The strength of each individual tree in the forest. A tree having a low error rate is a strong classifier.

3 Experiments on Numerical Datasets

To review the merit of the mapping procedure developed in the previous sections, we compare its results to the original Decision tree (DT) on the Euclidean space. Then we run the Decision tree based on the Ensemble Clustering space (DT-EC) on the same data after mapping the data using the ensemble clustering results. Moreover, we compare the DT-EC results to the results of the k-Nearest Neighbors classifier described in [8]. Additionally, we apply the random forest applied to the original set of features and on the EC features.

3.1 Datasets

The data consist of microRNA precursor sequences where each sequence consists of 4 nucleotide letters {A, U, C, G}. The source of such data is miRbase [12]. We have used part of the data that was used in different studies [13].

One simple way of representing sequences that consist of 4 nucleotide letters is by employing k-mers frequency. The k -mer counted in a given sequence were regulated by the length of the sequence.

Our features includes k-mer frequencies, other distance features that was just recently suggested by [16] (still not published), secondary features suggested by [14] and additionally many features describing pre-miRNAs have been proposed [15] are included in the features set. The number of features is 1038 features (Table 1).

The main data consists of information from 15 clades. The sequences of *Homo sapiens* were taken out of the data from its clade Hominidae. The data sets were passed from the process of removing homology sequences (keeping just one representative). One can generate about 256 datasets by considering a pair of tow clades including itself. We have randomly considered 10 datasets from those set of datasets shown in Table 2.

Table 1. The list of clades used in the study, the first column presents the name of the clade, the second column is the number of precursors available on miRBase, and the third column is the number of precursors after preprocessing the data.

Dataset	Number of precursors	Number of unique precursors
Hominidae	3629	1326
Brassicaceae	726	535
Hexapoda	3119	2050
Monocotyledons (Liliopsida)	1598	1402
Nematoda	1789	1632
Fabaceae	1313	1011
Pisces (Chondrichthyes)	1530	682
Virus	306	295
Aves	948	790
Laurasiatheria	1205	675
Rodentia	1778	993
<i>Homo sapiens</i>	1828	1223
Cercopithecidae	631	503
Embryophyta	287	278
Malvaceae	458	419
Platyhelminthes	424	381

Table 2. 10 datasets. The first column is the name of the first clade positive data, second column is the second clade negative data

Positive data	Negative data
Aves	Embryophyta
Cercopithecidae	Malvaceae
Embryophyta	Laurasiatheria
Fabaceae	Nematoda
Hexapoda	Aves
Laurasiatheria	Brassicaceae
Malvaceae	Fabaceae
Brassicaceae	Hexapoda
Hominidae	Cercopithecidae
Monocotyledons	<i>Homo sapiens</i>

3.2 Model Performance Evaluation Introduction

We have tested different numbers of EC clusters ranging from 10 to 100 iterated 10 times. For each level, we have run 100 iterations with equal sample size and calculated the mean of each performance measurements described below.

For every setup model, we determined several performance measures for the analysis of the classifier such as sensitivity, specificity and accuracy based on the following formulations (with TP: true positive, FP: false positive, TN: true negative, and FN referring to false negative classifications):

$$Sensitivity = \frac{TP}{TP + FN} (SE, \text{ recall})$$

$$Specificity = \frac{TN}{TN + FP} (SP)$$

$$Sensitivity = \frac{TP + TN}{TP + FN + TN + FP} (ACC)$$

4 Results

We have conducted a comparison study for using EC procedure that combined with Decision Trees (DT), K- Nearest Neighbors (KNN) classifiers and Random Forest (RF). The results are presented in Fig. 2. The results are clearly shown that the performance for each EC combined method is improving as the number of EC cluster is increasing. Interestingly, the accuracy becomes stable with a small variance after the number of cluster reaches 50. The accuracy line of the traditional method using all the features is represented by a solid line, which is a fixed value all over the different levels of clusters. The results show that the EC combined methods are able to get similar results with about 50 features of EC as using 1038 original features. In general, Random forest and Random forest-EC perform better than other methods by 1%. However, the aim of the study is not to achieve high performance rather than suggest a reduced feature space.

DT-EC is reaching better results on the data ringNormBasic with difference of 12% and 11% on the data MiceProtein.

The DT-EC and KNN-EC have comparable results in most cases.

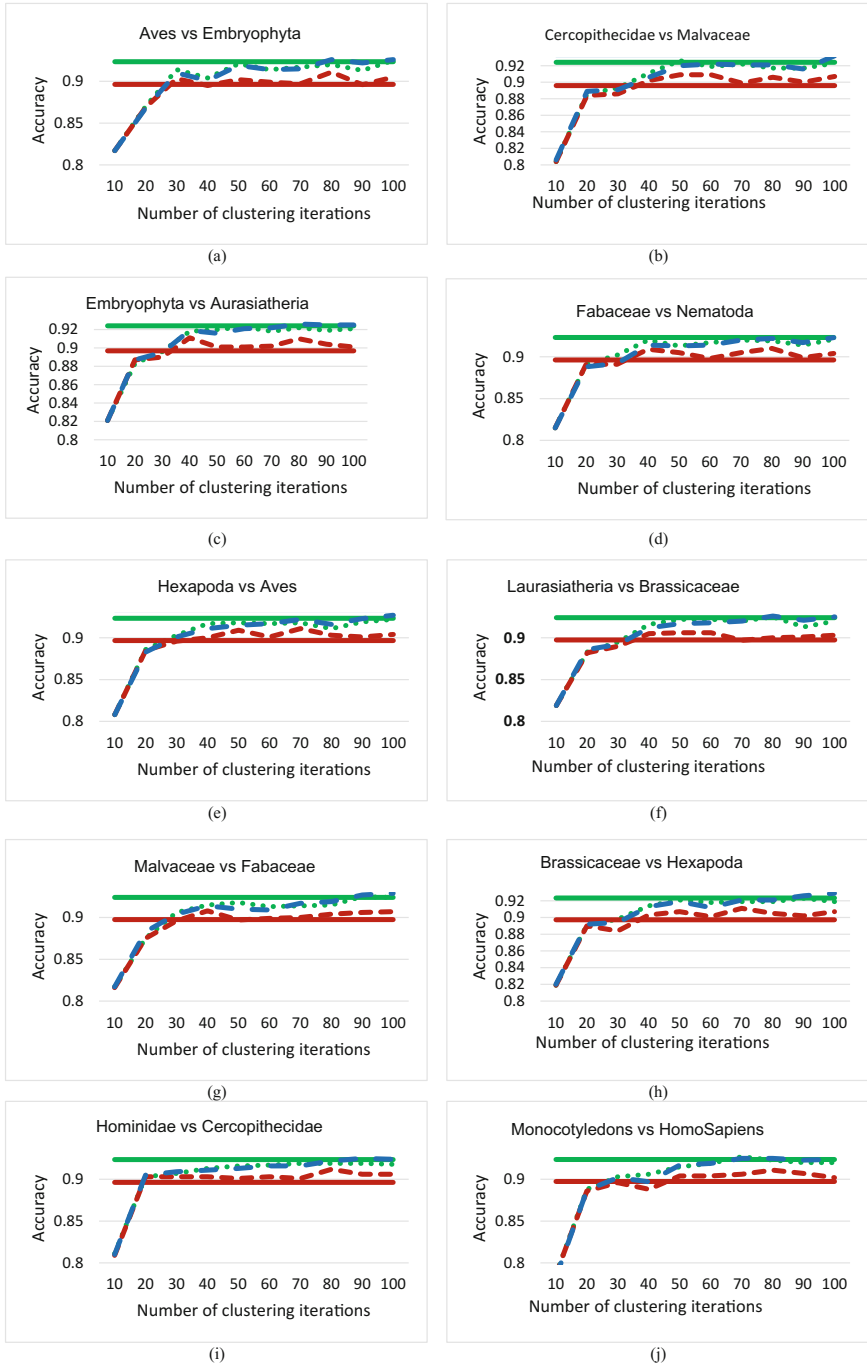


Fig. 2. The 10 results for the 3 classifiers applied to the original feature sets and to the EC feature space.

5 Conclusion

In this work, we presented a new mapping technique according to the ensemble clustering. Our method run different clustering algorithms several times in order to construct a new categorical space according to the clustering results.

Our main assumption was that the points within the same cluster share common traits more than the points within different clusters. Thus, this represent the objects based on the clustering space which may be better than the geometric space. Then we integrate this technique with the decision trees, k-NN and random forest classifier to review the performance of those algorithms based on the new space.

Our algorithm, however, is general and can be integrated with many algorithms. In this research, we use only the k-means clustering algorithm with different k values. In the future work, there are several directions: (1) checking the effect of the clustering algorithm to build an ensemble clustering space. (2) how to detect poor clustering results based on the training data. (3) how to reduce the volume of the data by combining similar point based on the EC.

Acknowledgment. This research was supported by the Max Stern Yezreel Valley College for LA and by Zefat Academic College for MY.

References

1. Topchy, A., Jain, A.K., Punch, W.: Combining multiple weak clusterings. In: Third IEEE International Conference on Data Mining, pp. 0–7 (2003)
2. Strehl, A., Ghosh, J.: Cluster ensembles – a knowledge reuse framework for combining multiple partitions. *J. Mach. Learn. Res.* **3**, 583–617 (2002)
3. Topchy, A., Jain, A.K., Punch, W.: Clustering ensembles: models of consensus and weak partitions. *IEEE Trans. Pattern Anal. Mach. Intell.* **27**(12), 1866–1881 (2005)
4. Dudoit, S., Fridlyand, J.: Bagging to improve the accuracy of a clustering procedure. *Bioinformatics* **19**(9), 1090–1099 (2003)
5. Fern, X.Z., Brodley, C.E.: Random projection for high dimensional data clustering: a cluster ensemble approach. In: Proceedings of the Twentieth International Conference on Machine Learning, vol. 20, pp. 186–193 (2003)
6. Fischer, B., Buhmann, J.M.: Bagging for path-based clustering. *IEEE Trans. Pattern Anal. Mach. Intell.* **25**(11), 1411–1415 (2003)
7. Derbeko, P., El-Yaniv, R., Meir, R.: Explicit learning curves for transduction and application to clustering and compression algorithms. *J. Artif. Intell. Res.* **22**, 117–142 (2004)
8. AbedAllah, L., Shimshoni, I.: k nearest neighbor using ensemble clustering. In: Cuzzocrea, A., Dayal, U. (eds.) DaWaK 2012. LNCS, vol. 7448, pp. 265–278. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-32584-7_22
9. AbedAllah, L., Shimshoni, I.: An ensemble-clustering-based distance metric and its applications. *Int. J. Bus. Intell. Data Min.* **8**(3), 264–287 (2013)
10. Yousef, M., Khalifa, W., AbedAllah, L.: Ensemble clustering classification compete SVM and one-class classifiers applied on plant microRNAs data. *J. Integr. Bioinform.* **13**(5), 304 (2016)
11. Breiman, L.: Random forests. *Mach. Learn.* **45**(1), 5–32 (2001)

12. Griffiths-Jones, S.: miRBase: microRNA sequences and annotation. *Curr. Protoc. Bioinform.* Chapter 12, Unit 12.9.1–10 (2010)
13. Yousef, M., Nigatu, D., Levy, D., Allmer, J., Henkel, W.: Categorization of species based on their microRNAs employing sequence motifs, information-theoretic sequence feature extraction, and k-mers. *EURASIP J. Adv. Signal Process.* **2017**(1), 70 (2017)
14. Yousef, M., Nebozhyn, M., Shatkay, H., Kanterakis, S., Showe, L.C., Showe, M.K.: Combining multi-species genomic data for microRNA identification using a Naive Bayes classifier. *Bioinformatics* **22**(11), 1325–1334 (2006)
15. Sacar, M.D., Allmer, J.: Data mining for microRNA gene prediction: on the impact of class imbalance and feature number for microRNA gene prediction. In: 2013 8th International Symposium on Health Informatics and Bioinformatics, pp. 1–6 (2013)
16. Yousef, M., Yousef, A., Allmer, J.: K-mer Distance a New Set of Features for Delineating among Pre-Cursor microRNAs from Different Species (2018)



Detecting Low Back Pain from Clinical Narratives Using Machine Learning Approaches

Michael Judd, Farhana Zulkernine^(✉), Brent Wolfrom, David Barber,
and Akshay Rajaram

Queen's University, Kingston, ON K7L 2N8, Canada
{m. judd, farhana}@queensu. ca,
{brent. wolfrom, david. barber}@dfm. queensu. ca,
arajarem@qmed. ca

Abstract. Free-text clinical notes recorded during the patients' visits in the Electronic Medical Record (EMR) system narrates clinical encounters, often using 'SOAP' notes (an acronym for subject, objective, assessment, and plan). The free-text notes represent a wealth of information for discovering insights, particularly in medical conditions such as pain and mental illness, where regular health metrics provide very little knowledge about the patients' medical situations and reactions to treatments. In this paper, we develop a generic text-mining and decision support framework to diagnose chronic low back pain. The framework utilizes open-source algorithms for anonymization, natural language processing, and machine learning to classify low back pain patterns from unstructured free-text notes in the Electronic Medical Record (EMR) system as noted by the primary care physicians during patients' visits. The initial results show a high accuracy for the limited thirty-four patient labelled data set that we used in this pilot study. We are currently processing a larger data set to test our approach.

Keywords: Text-mining · Natural language processing · Machine learning
Clinical decision support system · Back pain

1 Introduction

Physicians around the world use a variety of EMR systems to log data about patients' visits. These systems include patients' demography, physical and health-related information, laboratory tests, medications, billing information and unstructured text notes that the physicians enter in the EMR system based on their conversations with the patients. The patient-specific medical data, and more specifically, the free-text reports, provide an invaluable source of information for medical diagnosis. However, analyzing the unstructured text notes pose a difficult challenge due to the pervasive acronyms, spelling and typing errors, and dense author and domain-specific idiosyncrasies [1]. At the same time, correct extraction and appropriate presentation of the knowledge in the text notes and linking it with other structured health data can help physicians immensely in deciding about the right treatment plan. There is a strong movement

towards analyzing free-text medical data to revolutionize the healthcare research not only for the sake of biomedical research such as diseases, comorbidities and drug interactions, but also for data-driven and evidence-based decision-making in healthcare [2]. With hospital readmission penalties reaching an all-time high of over five billion dollars in the United States, the need for medical decision-support systems has never been greater [3].

With this global movement towards implementing EMRs, significant interest has been directed towards automating the processing and analyses of medical data [4]. The information within these EMRs enables us to build machine learning models for disease prediction which is also known as case-detection and suggest alternative treatments. Because the unstructured text notes provide a wealth of information, researchers are exploring a variety of Natural Language Processing (NLP) [5], Text Mining (TM) [4], and semantic knowledge management techniques [6] to accurately process this data. A recent review of text-mining algorithms for case-detection shows that they can be very effective when combined with the structured information within the EMR for case-detection; providing above 90% sensitivity and specificity in some studies [7]. Medical Decision Support Systems (DSS) such as IBM Watson Health [8] are applying cognitive computing techniques to ingest information from many sources along with statistical data mining and machine learning techniques to predict possible disease diagnosis and suggest alternative treatment plans. Medical data in the EMRs also contain hybrid data such as lab reports associated with image data. Researchers are working on extracting knowledge from these hybrid-distributed data sources and linking them to create efficient DSS [9].

Despite the potential, researchers are still facing many challenges and obstacles when applying NLP and TM on medical data. One major difficulty arises from the strict privacy and security requirements concerning medical data, which sets ethical and legal obstacles in accessing the data [10]. The free text data contains doctors' subjective opinions. Therefore, developing TM and machine learning algorithms to perform reliable case-detection based on doctors' chart notes poses yet another critical challenge. Furthermore, the text data are often incomplete or erroneous, contain domain specific terminology, and express negative, positive or neutral sentiments, which need to be interpreted correctly to develop a reliable DSS.

In this pilot study, we focus on applying unstructured text data analytics to the Open Source Clinical Application Resource (OSCAR) EMR [11] to diagnose chronic low back pain patterns. Low back pain is a common and costly health condition in Canada. A survey of 2400 Canadian's revealed an 83% lifetime prevalence of low-back pain, with 61% reporting low-back pain in the past year [12]. Canada's cost of medical expenditures for low back pain alone are estimated between \$6 and \$12 billion per year [13]. As low back pain is often first reported to primary care practitioners, these health professionals act as the gate keepers to referrals for further costly investigations [14], including imaging and specialist appointments. In alignment with guidelines from the College of Family Physicians of Canada, imaging for low back pain should not be recommended unless red flags are present [15]. Red flags can include: suspected epidural abscess or hematoma, suspected cancer, suspected infection, severe or progressive neurologic deficit, and a suspected compression fracture. However, literature shows that primary care physicians are in fact over imaging for non-specific low back

pain [13]. Low back pain symptoms are not detectable from the various health metrics such as blood pressure, height, weight, body mass index, and lab values. The overall goal of this pilot project was to study the effectiveness of NLP in analyzing the free text notes and identifying patients with low back pain by validating the results against the existing manual auditing process.

We explored existing open source de-identification tools to anonymize the text data while retaining necessary information in a format so that we can subsequently apply NLP techniques. We studied existing NLP tools and applied Apache cTAKES [16] to extract medical terms. Finally, we developed an information processing workflow and a DSS for the diagnosis and classification of low back pain. Therefore, the main contributions of the paper are: (i) a study of free and open-source software (FOSS) for de-identification and NLP of unstructured medical text notes, (ii) definition of an information processing workflow, and (iii) implementation and validation of a DSS for the classification of low back pain for a very small set of labelled data.

The remaining sections of the paper are organized as follows. Section 2 provides the background and lists some of the related work. Our framework is presented in Sect. 3. A discussion of the results is presented in Sect. 4. Section 5 concludes the paper stating our ongoing and future work.

2 Background and Related Work

2.1 Anonymization of Clinical Text

Anonymization of Protected Health Information (PHI) in clinical text is a crucial step within text-mining EMRs to preserve patient confidentiality. Anonymization involves the removal of patient identifiers defined by regulatory acts such as HIPPA. In most cases, only researchers with local authorization can analyze the data due to anonymization methods not being 100% accurate and as such, creates a bottleneck for research in the area [6].

Automating the process of de-identifying clinical text is a key to the success of finding a generalized way to text-mine EMRs. Currently, there are two main algorithmic methods used to anonymize PHI: rule-based and machine learning [17]. Both methods have their own unique sets of benefits and disadvantages to consider. Rule-based systems are a proven method that can guarantee that specific elements are always anonymized but requires time-consuming rule specification and customization for each data-set. Researchers must add entity specific information such as patient names or nearby locations to large dictionaries for the algorithm to work [18].

Conversely, most of the machine learning methods use supervised learning methods which attempt to identify words as PHI or not PHI. These methods do not require extensive customization; however, they require large datasets annotated by domain experts which are difficult to obtain. In a review of automatic de-identification methods for medical records, Meystre et al. [19] stated that methods based on dictionaries performed better when PHI is rarely mentioned in text, but machine learning methods performed better over all. The review also concluded that hybrid methods using both

machine learning and rule-based pattern matching performed best in terms of accurately removing PHI.

The major downside of all anonymization methods is the reduction in data quality for NLP processing. Medical text data is already highly irregular and difficult for NLP tasks, and the anonymization of entities makes medical concepts recognition increasingly more difficult. Any anonymization algorithm would have to be carefully tuned not to over-scrub the data to withhold the accuracy of NLP algorithms.

2.2 Clinical NLP Systems

Numerous clinical NLP systems have been developed to solve many of the biomedical text-mining problems [1, 4, 16]. They work by using standard NLP tasks such as tagging, chunking, parsing, entity recognition, and summarization combined with the information from machine readable medical domain knowledge-base systems such as the Unified Medical Language System (UMLS) [22] to extract clinical concepts from free-text. Some of the most accurate systems being developed are unfortunately proprietary, however, there are successful FOSS systems including cTAKES, CLAMP Toolkit, MedEx, MedKAT/p, and more recently QuickUMLS [1]. These systems have all been used in separate studies with varied levels of success, however, little work has been done to compare the effectiveness of these existing clinical NLP tools.

Currently, cTAKES, the system implemented in this framework, seems to have the most comprehensive solution for clinical NLP. With a large user community and the Apache Software Foundation behind the project, the system should continue to grow and improve. Originally developed by the Mayo Clinic, cTAKES is based upon the Apache Unstructured Information Management Architecture (UIMA) framework. It utilizes the OpenNLP toolkit for NLP and the UMLS for medical concept annotation. The design of the system is pipeline-based, consisting of different modules that include a sentence boundary detector, tokenizer, normalizer, part-of-speech tagger, shallow parser, and named entity recognizer [20].

An alternative system that showed promise for building a generic framework for case-detection was QuickUMLS [1]. QuickUMLS is a tool for fast, unsupervised biomedical concept extraction from medical text. To find medical concepts, QuickUMLS takes advantage of Simstring [21], a fast algorithm for approximate string matching. The tool achieves precision and recall comparable to cTAKES for medical concept extraction at speeds up to 135 times faster than the comparable systems. This makes QuickUMLS a strong contender as a generalized text-mining tool for clinical decision support when scalability becomes increasingly important. The downside of exclusively using QuickUMLS is that it lacks the features of a full NLP suite such as negation and part of speech tagging for the possibility of deeper text analytics.

cTAKES and QuickUMLS, along with many other NLP systems rely heavily on the UMLS for clinical information extraction. The UMLS is a resource containing many health and biomedical vocabularies developed by the US National Library of Medicine to enable interoperability between computer systems. It integrates over two million names mapping to about 900,000 concepts from more than sixty families of biomedical vocabularies, and about twelve million relationships exist between these concepts [22]. The main UMLS tool that NLP systems use is called the Metathesaurus

[22], which is a large biomedical thesaurus organized by concepts or meanings, and links to similar concepts from 200 vocabularies over various languages. It also identifies relationships between concepts with a tree structure, along with definitions and types for these concepts. For building a generic multilingual NLP framework for medical data, UMLS is extremely useful as it provides many synonyms for medical terms and supports multiple languages.

2.3 Clinical NLP Systems

Based on our review of the literature on disease case-detection, previous research efforts were mostly geared towards diagnosing specific disease cases rather than building a general framework such as the framework proposed in this paper. Many case-detection frameworks applied custom-built NLP solutions or rule-based algorithms to diagnose cancer, heart disease, and mental health with reasonable success. However, they all lack the generalization to apply to other disease conditions [15, 23].

In recent years, researchers have proposed some generalized machine learning-based case-detection frameworks [17, 24]. D’Avolio et al. [24] developed a generalized framework which consisted of a modified version of cTAKES for the NLP component and the Machine Learning for Language Toolkit for case-detection. Like the method used in this paper, the algorithm tested a number of different machine learning algorithms, attempting to maximize the classification accuracy. The system chose various feature sets of cTAKES output to classify three types of cancer with 0.90 accuracy in two of three cases.

Szlosek and Ferret [17] created a clinical DSS using SpaCy, a general NLP library with linear time processing algorithms, and the Scikit-learn library of machine learning algorithms. SpaCy was used to clean, tokenize, and vectorize the text data.

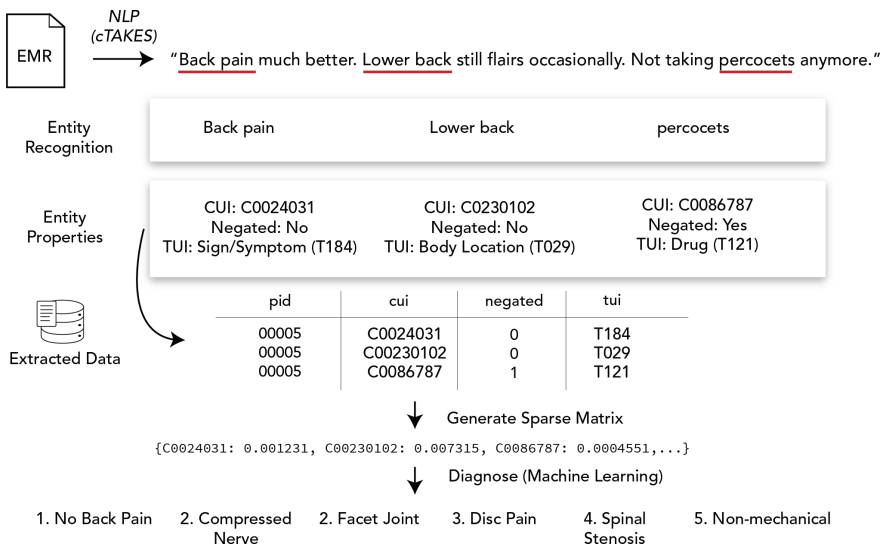


Fig. 1. Brief framework overview, focusing on Apache cTAKES output.

Case-detection was done using Scikit, which tested C-Support Vector Classification, Decision Tree Classifiers, and k-nearest neighbors classifiers. The DSS quickly managed to classify brain injuries with 0.98 sensitivity and 0.10 specificity. Although this system has an extremely high sensitivity rate, a DSS must have reasonable specificity since many people without brain injuries could be screened positive.

Combining clinical narratives and structured data from EMRs, including hospital billing codes and international classification of disease codes, also proved to be similarly effective to that of strictly using free-text data. Xu et al. [23] used the private medical NLP system MedLEE to first identify clinical concepts, then used those concepts combined with the hospital codes to encode medical conditions. These conditions were used as features to implement both a heuristic rule-based approach and machine learning approach to diagnose colorectal cancer cases. The Ripper machine learning algorithm with coded data and text data proved to be most effective, with 0.95 sensitivity and 0.96 specificity.

3 Our Framework

3.1 Data

We used data from the OSCAR EMR which included lab results, medical procedures, medications, and diagnoses including free-text data for social history, medical history, and clinical encounters. For this study, we only analyzed the encounter notes which contained seven-years of unstructured narrative text data recorded by the primary care provider during scheduled or walk-in appointments. For validation purposes, thirty-four patients of ages between 18–65 with no history of cancer were hand-selected with a complaint of back pain for greater than twelve months prior to data extraction. We attempted to classify each patient’s pattern of low back pain as disc pain, facet joint pain, compressed nerve pain, symptomatic spinal stenosis, or without back pain. The output from the framework was then compared to the gold-standard results obtained from a manual review process conducted by a medical domain expert. The Centre for Effective Practice (CEP) Clinically Organized Relevant Exam (CORE) Back Tool was used as a guide throughout the manual diagnosis process [25].

3.2 Pipeline

The focus of this study was to build a generic clinical notes text mining system to provide physicians with clinical decision support. With that in mind, the success of the project has largely been made possible through the utilization of numerous FOSS projects which are actively being developed in the field of clinical free-text processing. The framework was designed to consist of four main components, which can be extended or replaced with different algorithms. The pipeline consists of:

1. *Anonymization*: The open-source de-identification tool Deid to anonymize PHI [26].
2. *NLP*: cTAKES to add rich semantic and medical concept annotations.

3. *Information extraction*: A custom-built cTAKES output parser to extract clinical concepts and metadata into a standardized SQL table
4. *Case-detection algorithm*: Multiple supervised classifiers from Scikit-learn to classify the pattern of low back pain.

Anonymization. Anonymization of PHI is integrated in the data processing pipeline using an automated Perl-based de-identification software package for free-text medical records, Deid, that was designed by Neamatullah et al. [26]. The software uses lexical look-up tables, regular expressions, and simple heuristics to locate both HIPAA PHI, and additional PHI including common names and date variations. In Neamatullah’s study, the performance of the system was evaluated using a gold-standard manually annotated corpus which achieved an overall recall of 0.967 and precision of 0.749 [26].

Deid applies pattern matching to find PHI using numerous configuration parameters that can be set to meet the requirements of a country’s regulations. This makes Deid, or any pattern matching for that matter, a strong contender for countries which have strict regulations for PHI. Before Deid is used, the user must manually fill in dictionary files for patient names, doctor’s names, nearby hospitals, and places to use for de-identification. This step is required for guaranteed de-identification, however Deid does come with a basic American dictionary to find names and places, and can work without the manually created dictionary files. Once the dictionary files are prepared, a local text file has to be created with each patient’s clinical chart notes extracted from the patient database.

NLP. After anonymization, we applied cTAKES in the NLP step to extract relevant medical terminology and relevant negation from the free-text. Figure 1 depicts a simple example of cTAKES called the default clinical pipeline. The pipeline uses the UMLS to annotate text with anatomical sites, signs/symptoms, procedures, diseases/disorders and medications. For every medical entity cTAKES recognizes, it provides a normalized UMLS Unique Concept Identifier (CUI), Type Unique Identifier (TUI), plus values for negation, and a subject. Negation is detected in cTAKES using part of speech tagging, where the program can identify cases such as “not taking Percocets” which will negate the Percocets entity as shown in Fig. 1.

It should be noted that the current case-detection algorithm uses a modified cTAKES pipeline consisting of medical entity recognition and negation, and not the rest of the NLP components. cTAKES provides the ability to not only define a custom pipeline with any of the features within cTAKES but also permits the creation of custom dictionaries to add into cTAKES to create additional domain knowledge.

Information Extraction. The annotations which cTAKES provides contain rich clinical context for each patient. The output from the NLP was parsed using a custom-built entity parser, which iterates through the XML output from cTAKES extracting and inserting all CUI occurrences and their associated negations into a structured data format to be used by the case-detection algorithm. CUIs with negation are recognized as a separate term with a “-” prepended to the CUI. CUIs are unique codes from the UMLS which reference to medical concepts. Each CUI in the UMLS is mapped to numerous medical dictionary concepts in a tree-like data structure and has relationships to other CUIs.

Case Detection Algorithm. The structured data extracted from cTAKES provides a wealth of information to mine and develop a case-detection algorithm. The data extracted presents itself as a bag-of-words model, which contains the multiset of all CUIs within a patient’s record. This bag-of-words model in comparison to most text-classification approaches greatly reduces the feature-set size that a machine learning algorithm must account for. The medical term frequency $f_{t,p}$ is computed for each patient’s record such that t is the number of times a medical term occurs in patient record p . The term frequency is then normalized by dividing by the sum S of CUIs contained within each patient record as shown in the equation below.

$$f_{t,p} = \frac{\sum_{t \in p} f_{t,p}}{S} \quad (1)$$

From the calculated medical term frequencies, a sparse matrix was generated for machine learning using two different approaches. The first sparse-matrix created a superset of all CUIs, including every CUI for every patient. This method works well for small datasets but could cause scalability problems for large datasets. Additionally, this method accounts for medical terms that should not normally be considered in the case of a back pain diagnosis (Fig. 2).

```

1 all_patients = {001:[C0000241,C0058291,...],...}
2 backpain_terms = [lumbar pain, sciatica, ...]
3 back_cuis = []
4 sparse_matrix = {}
5
6 # get back pain terms from umls api
7 for term in backpain_terms:
8     back_cuis += get_umls_cui(term)
9
10 # generate sparse matrix
11 for id in all_patients:
12     sparse_mat[id] = {}
13     for cui in back_cuis:
14         sparse_matrix[id][cui] =
15         all_patients[id].count(cui) / len(all_patients[id])

```

Fig. 2. An overview of the second approach, creating a sparse matrix for each patient’s CUIs associated with back pain

Using the manually diagnosed and labelled training data of the patients, four supervised machine learning models were trained using the Scikit-learn library to classify the five patterns of back pain. These four machine learning models included Bernoulli Naïve-Bayes (BernoulliNB), Multinomial Naïve-Bayes (MultinomialNB), Linear Support Vector Classifier (LinearSVC), and Perceptron neural network with stochastic gradient descent classification. To obtain optimal classification accuracy, each model was trained through a fit and score exhaustive search to tune the hyper-parameters of each classifier.

4 Results

From the manual diagnosis, the patient distribution of back pain patterns included twelve patients with disc pain, five with compressed nerve pains, one with facet joint pain, one with spinal stenosis, one with non-mechanical back pain, and fourteen others without any back pain. This variation presents a problem for any supervised machine learning approach. Using the single case patterns as training data would worsen the detection accuracy and using the single cases as test data would always fail. With that in mind, we chose to omit the three single back pain cases from the patient dataset and split the remaining data into 65% training and 35% testing dataset. With this dataset, both sparse matrix generation approaches led to an optimal precision and recall values of 100% for the Linear Support Vector, and slightly lower values for both Naïve-Bayes algorithms. The linear perceptron performed rather poorly in all test cases possibly because of the very small size of training data. The results are shown in Fig. 3 where for each model the first column denotes the sensitivity and the second column denotes specificity.

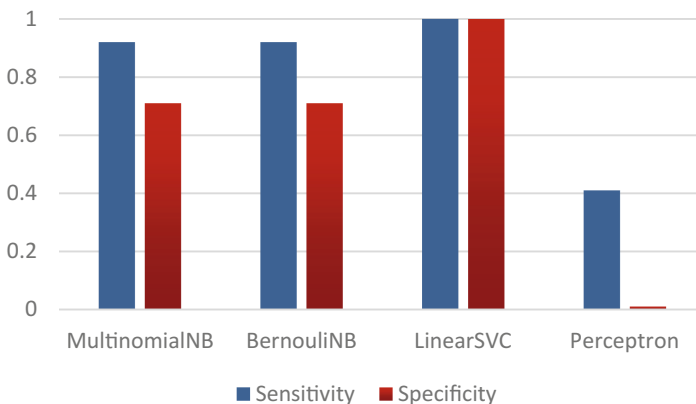


Fig. 3. Sensitivity and specificity for the four compared supervised learning algorithms.

Table 1 shows the performance measured and recorded for each component in the data processing pipeline. The processes were executed on an Intel Core i5-6200U 2.3 GHz CPU with 8 GB of RAM. For 3.27 Megabytes of plain-text clinical notes, the anonymization and cTAKES processing totalled five hours and ten minutes. For the small dataset used in this study, both approaches to generating sparse matrices had negligible impact on the performance. For both approaches, the case-detection models were by far the most time-efficient parts of the pipeline, with an average total training and testing time of less than 1 s.

Table 1. Running times for the core components of the framework

Framework component	Computational time
Anonymization (Read database, preprocess, anonymize)	13.2 min (~54%)
NLP processing (cTAKES)	3.4 min (~14%)
Information extraction	7.6 min (~31%)
Case-detection algorithms (testing + training)	<1 s (<1%)

4.1 Discussion

For this pilot NLP study on low back pain we had a limited set of labelled data. Therefore, these results are not a clear indicator of success. With the dataset, machine learning algorithms can be over or under trained, and generally modelling becomes inherently difficult due to the small sample size. The study provided valuable insights about the various processing steps in the pipeline, their relative performance with respect to time, existing FOSS tools and their performances. We are currently working on a larger, 1200 patient dataset for further evaluation.

As shown in Table 1, the computational performance of some of the system components such as Deid and cTAKES were underwhelming. However, the machine used for testing was not designed for powerful computation. We are currently exploring Apache Tika which supports integration with Apache cTAKES, and QuickUMLS as alternative tools to improve the performance of the NLP step. None of the system components applied parallel processing or even multi-core processing. The utilization of these methods would also greatly decrease the execution time of the complete analytic pipeline.

5 Conclusion and Future Work

Chronic low back pain deteriorates the quality of life. Medical imaging is an expensive procedure which can be effectively prescribed by physicians if the type of pain can be better diagnosed with the help of a medical DSS. Diseases such as low back pains and depression are difficult to diagnose from regular health metrics. Recent studies are exploring NLP and machine learning techniques to extract relevant diagnostic information from doctors' free text encounter notes data in the EMRs. In this paper we present our study on existing FOSS tools and algorithms, and the prototype of a simple medical DSS. The DSS can classify patients with low back pain based on unstructured text encounter notes data from an EMR system. We implemented a data processing pipeline utilizing FOSS anonymization algorithms, clinical NLP tools, and machine learning models to diagnose disc pain or compressed nerve pain using a limited number of patient data and achieved a 100% accuracy. The results also indicate that a generalized framework can be used to diagnose other medical conditions.

Since this is a preliminary study, there are numerous areas for improvement. Our ongoing work focuses on processing a larger dataset containing a greater number of patients' data. The current framework has been designed with loosely integrated

components such that each component can be replaced with a better option as one becomes available, or extended by contributing to the open-source projects.

The anonymization algorithm worked well for a project of this size, but the run times were too long to be used with larger datasets. Rudolf's [27] Clinical Records Anonymization and Text Extraction System showed massive improvements in anonymization performance in comparison to the current algorithm, which would add significant value to the current framework. It uses regular expressions as well, but runs at 14 Mb/s.

Additionally, the data extracted from cTAKES for case-detection is also currently underutilized. cTAKES and information extraction together takes almost half of the total processing time as shown in Table 1 but only a fraction of the extracted information is used in the later part of the pipeline for diagnosis. The long processing time will cause scalability issues for larger datasets. Other state of the art NLP tools such as IBM's Watson Health, Apache Tika with cTAKES, spaCy, and QuickUMLS can be explored to address the above issues. Big data processing frameworks and parallelization of the data processing pipeline can also improve the processing time.

Finally, the current case-detection algorithm is relatively simple. Using the NLP data which cTAKES provides in combination with the structured data within the EMR such as age, weight and lab reports, and developing a more complex model such as the long short-term memory Recurrent neural network model to take into account the patient encounter times will allow creating a more robust DSS. Diagnosis of diseases is often not very straight forward. Having a confidence interval for each classification and using an ensemble learning model will increase users' trust on the system and help build a reliable DSS that can be trusted by medical practitioners.






References

1. Soldaini, L., Goharian, N.: QuickUMLS: a fast, unsupervised approach for medical concept extraction (2016)
2. Jensen, P.B., Jensen, L.J., Brunak, S.: Mining electronic health records: towards better research applications and clinical care. *Nat. Rev. Genet.* **13**, 395–405 (2012)
3. Rau, J.: Medicare's Readmission Penalties Hit New High. <https://khn.org/news/more-than-half-of-hospitals-to-be-penalized-for-excess-readmissions/view/republish/>. Accessed 15 Jan 2018
4. Hassanzadeh, H., Nguyen, A., Koopman, B.: Evaluation of medical concept annotation systems. In: Proceedings of Australasian Language Technology Association Workshop, pp. 15–24 (2016)
5. Buckley, J., et al.: The feasibility of using natural language processing to extract clinical information from breast pathology reports. *J. Pathol. Inform.* **3**, 23 (2012)
6. Spasić, I., Livsey, J., Keane, J., Nenadić, G.: Text mining of cancer-related information: review of current status and future directions. *Int. J. Med. Inform.* **83**, 605–623 (2014)
7. Ford, E., et al.: Extracting information from the text of electronic medical records to improve case detection: a systematic review. *J. Am. Med. Inform. Assoc.* **23**(5), 1007–1015 (2016)
8. IBM: IBM Watson Health. <https://www.ibm.com/watson/health>. Accessed 3 Jan 2018
9. Sevenster, M., van Ommering, R., Qjan, Y.: Bridging the text-image gap: a decision support tool for real-time PACS browsing. *J. Digit. Imaging* **25**, 227–239 (2012)

10. Jensen, K. et al.: Analysis of free text in electronic health records for identification of cancer patient trajectories. *Sci. Rep.* **7** (2017)
11. OSCAR Canada: About OSCAR. <http://oscarcanada.org/about-oscar/brief-overview>. Accessed 28 Oct 2017
12. Bone and Joint Canada: Low Back Pain. <http://boneandjointcanada.com/low-back-pain/>. Accessed 10 Jan 2018
13. Canadian College of Family Physicians of Canada: Evidence-informed primary care management of low back pain. http://www.cfpc.ca/uploadedFiles/Directories/Committees_List/Low_Back_Pain_Guidelines_Oct19.pdf. Accessed 11 Jan 2018
14. Webster, B., Courtney, T., Huang, Y.H., Christiani, D.: Physicians' initial management of acute low back pain versus evidence-based guidelines. *J. Gen. Intern. Med.* **20**, 1132–1135 (2005)
15. Devereaux, M.: Low back pain. *Prim. Care: Clin. Off. Pract.* **31**, 33–51 (2004)
16. Savova, G.K., et al.: Mayo clinical text analysis and knowledge extraction system (cTAKES): architecture, component evaluation and applications. *J. Am. Med. Inform. Assoc.* **17**, 507–513 (2010)
17. Szlosek, D.A., Ferrett, J.: Using machine learning and natural language processing algorithms to automate the evaluation of clinical decision support in electronic medical record systems. *Gener. Evid. Methods Improv. Patient Outcomes* **4**(3), 1222 (2016)
18. Ferrández, O., South, B.R., Shen, S., et al.: Evaluating current automatic de-identification methods with Veteran's health administration clinical documents. *BMC Med. Res. Methodol.* **12**, 109 (2012)
19. Meystre, S., Savova, G., Kipper-Schuler, K., Hurdle, J.F.: Extracting information from textual documents in the electronic health record: a review of recent research. In: *Yearbook of Medical Informatics*, pp. 128–144 (2008)
20. Moja, L., et al.: Effectiveness of computerized decision support systems linked to electronic health records: a systematic review and meta-analysis. *Am. J. Public Health* **104**(10), 104–116 (2014)
21. Okazaki, N., Tsujii, J.: Simple and efficient algorithm for approximate dictionary matching. In: *Proceedings of the 23rd International Conference on Computational Linguistics*, pp. 851–859 (2010)
22. Bodenreider, O.: The Unified Medical Language System (UMLS): integrating biomedical terminology. *Nucleic Acids Res.* **32**, D267–D270 (2004)
23. Xu, H., Fu, Z., Chen, Y., et al.: Extracting and integrating data from entire electronic health records for detecting colorectal cancer cases. In: *AMIA Annual Symposium Proceedings*, pp. 1564–1572 (2011)
24. D'Avolio, L.W., et al.: Evaluation of a generalizable approach to clinical information retrieval using the automated retrieval console (ARC). *J. Am. Med. Inform. Assoc.* **17**(4), 375–382 (2010)
25. Centre for Effective Practice: Clinically Organized Relevant Exam. http://www.cfpc.ca/uploadedFiles/Resources/Resource_Items/Health_Professionals/CEP_CoreBackTool_2016.pdf. Accessed 20 Aug 2017
26. Neamatullah, I., et al.: Automated de-identification of free-text medical records. *BMC Med. Inform. Decis. Mak.* **8**, 8–32 (2008)
27. Rudolf, C.N.: Clinical records anonymisation and text extraction (CRATE): an open-source software system. *BMC Med. Inform. Decis. Mak.* **17**, 50 (2017)



Classifying Big DNA Methylation Data: A Gene-Oriented Approach

Emanuel Weitschek^{1,3}, Fabio Cumbo^{2,3}, Eleonora Cappelli^{2,3},
Giovanni Felici³, and Paola Bertolazzi³

¹ Department of Engineering, Uninettuno University,
Corso Vittorio Emanuele II 39, 00186 Rome, Italy
emanuel@iasi.cnr.it

² Department of Engineering, Roma Tre University,
Via della Vasca Navale 79, 00146 Rome, Italy
fabio.cumbo@iasi.cnr.it, eleonora.cappelli@uniroma3.it

³ Institute for Systems Analysis and Computer Science, National Research Council,
Via dei Taurini 19, 00185 Rome, Italy
{giovanni.felici,paola.bertolazzi}@iasi.cnr.it

Abstract. Thanks to Next Generation Sequencing (NGS) techniques, public available genomic data of cancer is growing quickly. Indeed, the largest public database of cancer called The Cancer Genome Atlas (TCGA) contains huge amounts of biomedical big data to be analyzed with advanced knowledge extraction methods. In this work, we focus on the NGS experiment of DNA methylation, whose data matrices are composed of hundred thousands of features (i.e., methylated sites). We propose an efficient data processing procedure that permits to obtain a gene-oriented organization and enables to perform a supervised machine learning analysis with state-of-the-art methods. The procedure divides the original data matrices into several sub-matrices, each one containing the sites located within the same gene. We extract from TCGA DNA methylation data of three tumor types (i.e., breast, prostate, and thyroid carcinomas) and we are able to successfully discriminate tumoral from non tumoral samples using function-, tree-, and rule-based classifiers. Finally, we select the best performing genes (matrices) ranking them according to the accuracy of the classifiers and we execute an enrichment analysis of them. Those genes can be further investigated by domain experts for proving their relation to the cancers under study.

Keywords: Classification · DNA methylation · Cancer

1 Introduction

Thanks to the Next Generation Sequencing (NGS) techniques life scientists are able to perform a large number of biological experiments in parallel [12], e.g., genome sequencing (DNA-sequencing) [22], transcriptome profiling (RNA sequencing) [18, 24], microRNA sequencing (miRNA-seq) [41], protein-DNA

interactions (Chip-Seq) [25], Copy Number Variation (CNV) [7], and characterization of the epigenome or chemical changes in the DNA (DNA methylation) [2, 3, 27]. Because of the great advances of NGS techniques, the above-mentioned experiments can be performed at low cost and at high speed even when dealing with human samples [14, 29, 37]. NGS can be applied for case control studies, i.e., specific studies that aim to identify subjects by outcome status at the outset of the investigation, e.g., whether the subject is diagnosed with a disease. Subjects with such an outcome are categorized as *cases*. Once outcome status is identified, controls (i.e., subjects without the outcome but from the same source population) are selected [30].

Among the NGS experiments, DNA methylation stands out specifically for diagnosing diseases. DNA methylation is one of the most studied epigenetic changes in human cells. The changes in DNA methylation patterns are crucial in the development of diseases such as multiple sclerosis, diabetes, schizophrenia, and many forms of cancer [19, 20, 23, 32, 40, 42]. In humans the methylation is a biochemical modification that involves the addition of a methyl group to the level of carbon-5 of cytosine, almost exclusively in the context of the CpG dinucleotide, i.e. cytosine followed by a guanine [2]. It is estimated that 80% of all the CpG islands of a gene is methylated in mammals [16]. Many approaches exploit the high quality and the sensitivity of NGS for the methylation analysis. Most of the methods are based on bisulfite conversion combined with NGS, to determine the percentage of methylated cytosines in a CpG island. This measure is called *beta value* (bv) [11]. The beta value is defined as the ratio of the methylated allele intensity and the overall intensity (i.e. the sum of methylated and unmethylated allele intensities) and is a measure in the range of 0–1, where 1 represents full methylation and 0 no methylation at all. For more details about the DNA methylation experimental techniques the reader may refer to [13, 27].

In this work, we address the issue of analyzing DNA methylation experiments of case control studies in cancer. For this purpose we extract DNA methylation data extracted from The Cancer Genome Atlas (TCGA) [33], focusing on three types of tumors, i.e., Breast Invasive Carcinoma (BRCA), Prostate Adenocarcinoma (PRAD), and the Thyroid Carcinoma (THCA). We select the samples for which we have the experiments conducted on both case and control tissues, and we perform a processing of data in order to map the methylated sites to the genes where they are located. Additionally, we analyze the processed data with supervised machine learning (i.e., classification) algorithms for identifying the case and the control samples. We select the best performing genes for studying the three types of cancer and therefore we identify many potential oncogenes. In the following we use the terminology of TCGA, i.e., *normal* for control samples and *tumoral* for case ones.

This paper is organized as follows. Section 2 illustrates the data and the methods adopted in this work, in Subsect. 2.1 we describe briefly The Cancer Genome Atlas (TCGA), a comprehensive archive of genomic and clinical data related to human cancer. Then, in Subsect. 2.3 we show our data preparation procedure and our gene-oriented data organization. Afterwards, in Subsect. 2.3,

we illustrate the supervised machine learning methods adopted in this work, i.e., Support Vector Machines, Decision Trees, and rule-based classifiers. In Sect. 3 we illustrate and discuss the obtained computational results on the three types of cancer. Finally, in Sect. 4 we delineate the conclusions and present the possible perspectives of our study.

2 Methods

2.1 The Genomic Data Commons

In this work, we consider DNA methylation data extracted from the TCGA2BED [9] repository that contains genomic, clinical and biospecimen data in BED format, previously extracted from the Genomic Data Commons (GDC) [15]. The GDC is online since July 2016 and is an evolution of The Cancer Genome Atlas (TCGA) [33]. The latter has been a breakthrough in cancer research, making available huge amounts of genomic and clinical data of human cancer patients, containing data of more than 10,000 patients with 33 different tumor types (more than 15 TB of genomic and clinical data). The GDC is the result of an initiative funded by the National Cancer Institute (NCI) [17] with the aim of creating a unified data system that can promote the sharing of genomic and clinical data among researchers. The new portal collects, reviews, and makes accessible genomic and clinical data produced by several projects, i.e., The Cancer Genome Atlas (TCGA), Therapeutically Applicable Research to Generate Effective Treatments (TARGET) [10], Cancer Genome Characterization Initiative (CGCI), and Cancer Cell Line Encyclopedia (CCLE). The main aim of the GDC is to allow access to high-quality datasets derived from programs supported by NCI and to recommend guidelines for their organizations. The GDC portal publicly provides dataset of the following genomic experiments of more than 40 tumor types: DNA sequencing, Copy Number Variation, Somatic Mutations, DNA Methylation, Gene Expression Quantification, and miRNA Expression Quantification. Indeed, GDC derives those information from raw data produced by the sequencers with well-known bioinformatics pipelines that have been developed with the ongoing contribution of genomic cancer community experts, and that are described in the GDC Data Harmonization guide [1]. The high quality of data is ensured by a list of best practices that the GDC severely observes: (i) strict controls on the data samples; (ii) implementation of strict data validation procedures; (iii) production of reliable and harmonized derivative data. Moreover, analysis pipelines are implemented using techniques that make them reproducible, interoperable on multiple platforms, and that can be shared with all members of the community. The GDC distinguishes between open access data, which do not require authorization and are generally high-level genomic data, and controlled access data, which instead require authorization to access them.

In this work, we focus on DNA methylation data of TCGA, where each sample is represented with a list of following fields: gene symbols, chromosomes, genomic

coordinates (where the methylations occur), and their beta value (methylation values).

2.2 Data Extraction and Preparation

DNA methylation data of TCGA can be organized as follows. We collect n samples each one with its m features and their class labels (conditions), e.g., normal and tumoral. We represent every sample i by the vector $bv_i = (bv_{i1}, bv_{i2}, \dots, bv_{im}, bv_{ic})$, where $bv_{ij} \in \mathbb{R}$, $i = 1, \dots, n$, $j = 1, \dots, m$ and $bv_{ic} \in \{\textit{normal}, \textit{tumoral}\}$. We build the data matrix with the vectors bv_1, f_2, \dots, bv_n , where the rows represent the samples and the columns the features. In DNA methylation experiments the features are the methylated sites and their values represent the percentages of methylated cytosines in a CpG island. This percentage is called *beta value* (bv). The DNA methylation matrix is represented in Table 1. A DNA methylation experiment extracts more than 450 thousand sites on hundreds of samples. Therefore the DNA methylation matrix is composed of a large number of features (>450 thousand) and is not easily tractable with state of the art data analysis methods, which are not able to handle such big data sets. We propose to analyze these large data matrices by dividing them into S sub-matrices, with S representing the number of genes, where the methylated sites can be mapped. Indeed, each methylated site can be assigned to a specific gene region, where the site is located. The processing procedure is composed of following steps:

1. find the number S of distinct genes where the methylated sites can be mapped;
2. order the methylated sites according to their associated gene symbol;
3. extract the S sub-matrices with n samples and h features ($h \lll n$), whose site are located within the same gene region, i.e. with the same gene symbol.

So we obtain S sub-matrices, one for each gene, with the same format of Table 1, but with only h features (with $2 \leq h \leq 20$). We propose to analyze each sub-matrix with supervised machine learning methods [5, 26, 34, 35, 38], This procedure permits to reduce substantially the number of features in each matrix and to perform a gene-oriented analysis, because each matrix is associated to a gene. We report in Sect. 3 the list of the genes, whose matrices obtain the best performances when analyzed with classification algorithms.

Table 1. DNA-methylation data matrix

Sample	Site ₁	Site ₂	...	Site _{m}	Class
Sa ₁	bv ₁₁	bv ₁₂	...	bv _{1m}	Normal
Sa ₂	bv ₂₁	bv ₂₂	...	bv _{2m}	Tumoral
...
Sa _{n}	bv _{n1}	bv _{n2}	...	bv _{$n$$m$}	Normal

2.3 Supervised Machine Learning

We propose to analyze the processed data with supervised machine learning methods [5,26,34,35,38]. The aims are to distinguish the tumoral from non tumoral samples in an effective way. Classification problems are intended to identify the characteristics that indicate the group to which each sample belongs [31]. A classification model can be used to understand the existing data and to predict to which class a new sample belongs. The performance of a classifier is measured on the generalization ability, i.e., the ability to give to each new experimental observation the correct class.

For additional details about classification the reader may refer to [36]. In this work, we adopt function-, tree- and rule-based classifiers (i.e., Support Vector Machines (SVM) [8], C4.5 [28], RIPPER [6], and CAMUR [5]) in order to evaluate the best performing method and to identify the genes whose DNA methylation data has the most discriminating power.

C4.5 [28] is an algorithm for the generation of decision trees used for classification. A decision tree is a structure similar to a flow chart, where each node denotes a test on an attribute, each branch represents a result of a test, and every leaf is labeled by a class. Indeed a node with outgoing edges is termed test node and the final nodes are the leaves.

Repeated Incremental Pruning to Produce Error Reduction (RIPPER) [6] is an algorithm based on logic formulas, i.e., a combination of significant features with logic operators (and, or, not, <, >, =) in the form of “if-then rules”. It builds a set of rules that identify classes by minimizing the amount of error. The error is defined by the number of training examples miss-classified by the rules.

Classifier with Alternative and Multiple Rule-based models (CAMUR) [5] is a multiple rule-based classifier, which extracts alternative and equivalent classification models. CAMUR combines the RIPPER algorithm with a repeated classification procedure, deleting iteratively the features that appear in the classification models from the dataset in order to extract many classification models. The iterative procedure is stopped when the classification performance is below a given threshold (e.g., 90%) or a given number of iterations has been reached (e.g., 100).

It is worth noting that for the SVM, C4.5, and RIPPER algorithms we use the implementations provided in the Weka package [39], called SMO for SVM, J48 for C4.5, and Jrip for RIPPER, and we perform a 10-fold cross validation sampling procedure for the application of the classifiers.

3 Results and Discussion

The application of the classification algorithms on the S sub-matrices (each one associated to a gene) allows us to select for each tumor the best performing genes according to the obtained accuracy.

A synthetic representation of the results is shown in Table 2, where we report for each tumor (i) the number of genes that perform with an accuracy $\geq 90\%$

for all the three considered classifiers, and the number of analyzed experiments (one half tumoral samples, one half on normal samples). It is worth noting that the total number of considered genes per tumor is always in the range of ~ 25 thousand and the number of considered methylated sites is in the range of >450 thousand, because of the used Illumina DNA methylation technique.

Table 2. Number of samples and of genes obtained from the classification procedures for each tumor

Cancer	Samples	Selected genes
BRCA	192	2392
PRAD	100	103
THCA	112	42

For BRCA the number of extracted genes is still high, see 1 for details regarding the number of genes extracted by each algorithm. This means that the beta value is a good metric to establish the discriminant power of a gene for this particular tumor. Therefore, we further investigate the Breast Invasive Carcinoma (BRCA).

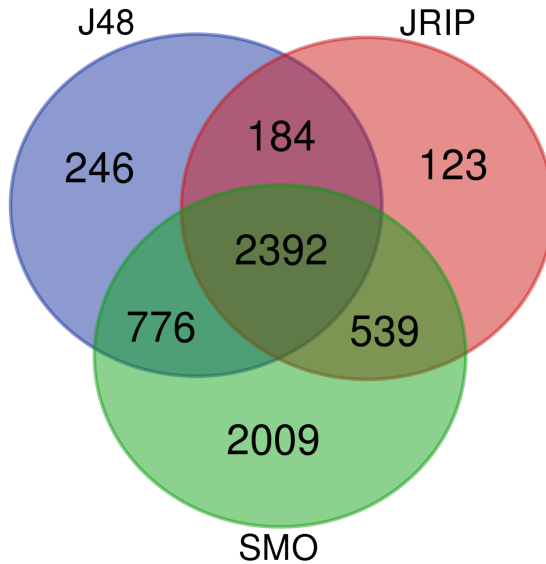


Fig. 1. Number of genes in BRCA where the classifiers (SMO, J48, and Jrip) reach more than 90% of accuracy. All classifiers obtain more than 90% of accuracy in 2392 genes.

In particular, for further reducing the selected genes we apply CAMUR on gene expression data of the same samples and extract a list of logic formulas highlighting the genes that appear in the rules. We subsequently intersect this set of genes with the previously generated set obtained from the other classifiers (SVM, C4.5, and RIPPER) on DNA methylation data generating a new set of 72 common genes. Finally, we highlight a list of logic formulas (shown in Table 3) with a classification rate $\geq 90\%$ that contain the genes in this new set.

Table 3. Examples of classification rules extracted by CAMUR on gene expression data of BRCA with a classification rate $\geq 90\%$, each rule is able to distinguish tumoral from normal samples.

$(SDPR 8436 \geq 10.53)$ and $(SLC44A4 80736 \leq 47.42)$
$(SDPR 8436 \geq 10.53)$ and $(SLC44A4 80736 \leq 51.96)$
$(SDPR 8436 \geq 11.64)$ and $(SLC44A4 80736 \leq 47.69)$
$(TMEM220 388335 \geq 2.51)$ and $(ITIH5 80760 \geq 9.74)$
or $(MYOM2 9172 \geq 29.70)$
$(TMEM220 388335 \geq 2.52)$ and $(LIMS2 55679 \geq 10.72)$
$(TMEM220 388335 \geq 2.52)$ and $(LIMS2 55679 \geq 10.92)$
$(TMEM220 388335 \geq 2.62)$ and $(LIMS2 55679 \geq 10.72)$
or $(SDPR 8436 \geq 17.55)$ and $(A2BP1 54715 \geq 0.019)$
$(TMEM220 388335 \geq 2.62)$ and $(LIMS2 55679 \geq 10.72)$
or $(TRIM59 286827 \leq 1.23)$ and $(A2BP1 54715 \geq 0.019)$

3.1 Gene Enrichment Analysis

In order to validate the classification results, we perform an enrichment analysis on the previous selected genes. We exploit the NCBI [21] Entrez Gene database to find a relationship between the extracted genes and the analyzed tumors. In particular, we highlight all the genes that could potentially cause the development of a neoplastic phenotype in the cell, as shown in Table 4.

In the case of the BRCA, 23 genes are known in literature to be related to the Breast Invasive Carcinoma, 16 genes in the PRAD gene set are related to the Prostate Adenocarcinoma, and only one gene in the THCA gene set, is related to the Thyroid Cancer. The final result of this enrichment analysis is reported in Table 5.

The remaining genes in the three extracted gene sets could be of course considered in future experiments as new targets for those specific diseases. It is worth noting that the highlighted genes are also involved in other diseases (e.g., Obesity, Spinal Muscular Atrophy, Hypotension, etc.).

Table 4. Subset of extracted genes, which are generally related to cancer, for each investigated tumor dataset.

Tumor abbreviation	Cancer related genes
BRCA	A2M, ABCB1, ABCC1, ABCG1, ACOT7, ACOX2, ADAM19, ADAM33, ADAMTS16, ADAMTS17, ADCYAP1R1, ADORA2A, AKAP2, ALDH1A2, AQP1, ATXN1, B4GALNT3, CA12, CD300LG, CDCP1, CPA1, CREB3L1, CRYAB, DENND2D, DST, FAM92A1, FGF1, FHL1, HEPACAM, HOXA7, IGFBP6, IL11RA, INHBA, ITIH5, KIF26B, LGI4, LIMS2, LRRC3B, MEG3, MUC1, PPP1R14A, PRKD1, SDPR, SPRY2, SSTR1, TMEM220, TNXB, TRIM59
PRAD	ADORA3, AKAP2, ARHGEF2, CA9, CASC2, CAV2, CCDC8, CCK, CD8B, CDH23, CHST11, CLIC3, CNTN1, COL3A1, COL4A5, COL4A6, CYBA, CYP2A13, DAB1, DOCK2, EFEMP2, FEV, FZD7, GALNT6, GATA3, GGT5, GJA1, HAPLN3, HIF3A, HVCN1, IL1B, IL2RB, INCA1, KCNJ3, KCTD8, LRRC4, LTC4S, MASP1, MCAM, MIR1258, MIR130B, MIR301B, MIR575, NBR1, NISCH, PDZD2, PFKP, PRR5, PYCARD, RASL10B, RBM38, SALL2, SEPT4, SIX2, SLC2A5, SLC6A2, SND1, TLX1, TMEM106A, TOM1L2, UBE4B, ZNF154, ZNF385B, ZNF577
THCA	AFAP1, BMPR1B, CAMP, CD96, CDH23, CHRN4, CMIP, COX5B, DDAH2, ELOVL5, FCGR3B, IL23R, ITIH2, KIFC3, KLHDC8A, LOH12CR1, MAP3K6, MGAT5, NCOR2, PLA2G3, RARA, SCTR, STRA6, TCL1B, TMEM127, TNFRSF12A, ZBTB20

Table 5. Subset of extracted genes, which are specifically related to the cancer under study, for each investigated tumor dataset.

Tumor abbreviation	Cancer related genes
BRCA	A2M, ABCB1, ABCC1, ACOT7, ACOX2, ADAMTS16, ADAMTS17, ADORA2A, AQP1, CA12, CDCP1, CREB3L1, CRYAB, FGF1, IL11RA, INHBA, ITIH5, KIF26B, LRRC3B, MEG3, MUC1, PRKD1, SDPR
PRAD	ADORA3, AKAP2, CA9, CNTN1, COL4A6, DOCK2, FZD7, GATA3, GJA1, IL1B, MCAM, MIR130B, MIR301B, PDZD2, PYCARD, SND1
THCA	NCOR2

4 Conclusions

In this work, we presented a methodology to efficiently extract a list of relevant genes in DNA methylation data of cancer exploiting consolidated machine learning algorithms. In particular, we considered for each gene the beta values of all its methylated sites aggregated in a matrix. This data representation allowed us to create good performing classification models able to discriminate tumoral and non tumoral samples and to select the best accurate genes. To confirm our methodology, we applied our procedure to three different types of cancers (breast, prostate, and thyroid carcinomas) obtaining promising results. Finally, we executed an enrichment analysis in order to highlight the genes related to the development of a particular tumor as a final validation of our procedure. In this simple approach, each gene is considered independently of the others and some genes that act simultaneously in the tumoral process may be ignored. Therefore, in future we are going to improve the method for considering many methylated sites of different genes simultaneously [4]. As additional future directions, we suggest to extend the number of applied machine learning algorithms for retrieving a stricter list of relevant genes thanks to the combination of their results. This will allow one to drastically reduce the number of genes to guide a smarter design of future experiments. To conclude, we propose to apply our procedure to many other DNA methylation datasets related to different cancer types in order to extend our analysis, further validate the methodology, and discover novel biological insights in tumor studies.

References

1. Genomic data harmonization. <https://gdc.cancer.gov/about-data/data-harmonization-and-generation/genomic-data-harmonization-0>
2. Bird, A.: DNA methylation patterns and epigenetic memory. *Genes Dev.* **16**(1), 6–21 (2002)
3. Bird, A.P.: CpG-rich islands and the function of DNA methylation. *Nature* **321**(6067), 209–213 (1985)
4. Celli, F., Cumbo, F., Weitschek, E.: Classification of large DNA methylation datasets for identifying cancer drivers. *Big Data Res.* (2018). <https://doi.org/10.1016/j.bdr.2018.02.005>
5. Cestarelli, V., Fison, G., Felici, G., Bertolazzi, P., Weitschek, E.: CAMUR: knowledge extraction from RNA-Seq cancer data through equivalent classification rules. *Bioinformatics* **32**(5), 697–704 (2016)
6. Cohen, W.W.: Fast effective rule induction. In: *Proceedings of the Twelfth International Conference on Machine Learning*, pp. 115–123 (1995)
7. Conrad, D.F., et al.: Origins and functional impact of copy number variation in the human genome. *Nature* **464**(7289), 704–712 (2010)
8. Cristianini, N., Shawe-Taylor, J.: *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. Cambridge University Press, Cambridge (2000)
9. Cumbo, F., Fison, G., Ceri, S., Masseroli, M., Weitschek, E.: TCGA2BED: extracting, extending, integrating, and querying the cancer genome atlas. *BMC Bioinform.* **18**(1), 6 (2017)
10. Downing, J.R., et al.: The pediatric cancer genome project. *Nat. Genet.* **44**(6), 619–622 (2012)
11. Du, P., et al.: Comparison of Beta-value and M-value methods for quantifying methylation levels by microarray analysis. *BMC Bioinform.* **11**(1), 587 (2010)
12. Enal Razvi, P.: Next-generation sequencing translating from research towards clinical utility: products in the space and market trends (2013). GENengnews.com. Accessed Feb 2015
13. Handel, A.E., Ebers, G.C., Ramagopalan, S.V.: Epigenetics: molecular mechanisms and implications for disease. *Trends Mol. Med.* **16**(1), 7–16 (2010)
14. Hayden, E.C.: Technology: the \$1,000 genome. *Nature* **507**(7492), 294–5 (2014)
15. Hinkson, I.V., Davidsen, T.M., Klemm, J.D., Kerlavage, A.R., Kibbe, W.A.: A comprehensive infrastructure for big data in cancer research: accelerating cancer research and precision medicine. *Frontiers in cell and developmental biology* **5**, 83 (2017)
16. Jabbari, K., Bernardi, G.: Cytosine methylation and CPG, TPG (CPA) and TPA frequencies. *Gene* **333**, 143–149 (2004)
17. Jensen, M.A., Ferretti, V., Grossman, R.L., Staudt, L.M.: The NCI genomic data commons as an engine for precision medicine. *Blood* **130**, 453–459 (2017). <https://doi.org/10.1182/blood-2017-03-735654>
18. Li, B., Dewey, C.N.: RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. *BMC Bioinform.* **12**(1), 323 (2011)
19. Liggett, T., et al.: Methylation patterns of cell-free plasma DNA in relapsing-remitting multiple sclerosis. *J. Neurol. Sci.* **290**(1), 16–21 (2010)
20. Luk, S.T.C., Tong, M., Ng, K.Y., Yip, K.Y.L., Guan, X.Y., Ma, S.: Identification of ZFP42/REX1 as a regulator of cancer stemness in CD133⁺ liver cancer stem cells by genome-wide DNA methylation analysis. *Nat. Genet.* **77**(13), 4352 (2017)

21. Maglott, D., Ostell, J., Pruitt, K.D., Tatusova, T.: Entrez gene: gene-centered information at NCBI. *Nucl. Acids Res.* **33**(suppl. 1), D54–D58 (2005)
22. McKenna, A., et al.: The genome analysis toolkit: a MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Res.* **20**(9), 1297–1303 (2010)
23. Mill, J., et al.: Epigenomic profiling reveals DNA-methylation changes associated with major psychosis. *Am. J. Hum. Genet.* **82**(3), 696–711 (2008)
24. Mortazavi, A., Williams, B.A., McCue, K., Schaeffer, L., Wold, B.: Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nat. Methods* **5**(7), 621–628 (2008)
25. Park, P.J.: Chip-Seq: advantages and challenges of a maturing technology. *Nat. Rev. Genet.* **10**(10), 669–680 (2009)
26. Polychronopoulos, D., Weitschek, E., Dimitrieva, S., Bucher, P., Felici, G., Almirantis, Y.: Classification of selectively constrained dna elements using feature vectors and rule-based classifiers. *Genomics* **104**(2), 79–86 (2014)
27. Portela, A., Esteller, M.: Epigenetic modifications and human disease. *Nat. Biotechnol.* **28**(10), 1057–1068 (2010)
28. Quinlan, J.R.: *C4.5: Programs for Machine Learning*. Elsevier, New York (2014)
29. Sheridan, C.: Illumina claims \$1,000 genome win. *Nat. Biotechnol.* **32**(2), 115 (2014)
30. Song, J.W., Chung, K.C.: Observational studies: cohort and case-control studies. *Plast. Reconstr. Surg.* **126**(6), 2234 (2010)
31. Tan, P., Steinbach, M., Kumar, V.: *Introduction to Data Mining*. Addison Wesley, Boca Raton (2005)
32. Toperoff, G., et al.: Genome-wide survey reveals predisposing diabetes type 2-related DNA methylation variations in human peripheral blood. *Hum. Mol. Genet.* **21**(2), 371–383 (2012)
33. Weinstein, J.N., et al.: The cancer genome atlas pan-cancer analysis project. *Nat. Genet.* **45**(10), 1113–1120 (2013)
34. Weitschek, E., Felici, G., Bertolazzi, P.: MALA: a microarray clustering and classification software. In: *Database and Expert Systems Applications (DEXA), 2012 23rd International Workshop on Biological Knowledge Discovery*, pp. 201–205. IEEE (2012)
35. Weitschek, E., Felici, G., Bertolazzi, P.: Clinical data mining: problems, pitfalls and solutions. In: *Database and Expert Systems Applications (DEXA) 2013, 24th International Workshop on Biological Knowledge Discovery and Data Mining*, pp. 90–94. IEEE (2013)
36. Weitschek, E., Fiscon, G., Felici, G.: Supervised DNA barcodes species classification: analysis, comparisons and results. *BioData Min.* **7**(1), 1 (2014)
37. Weitschek, E., Santoni, D., Fiscon, G., De Cola, M.C., Bertolazzi, P., Felici, G.: Next generation sequencing reads comparison with an alignment-free distance. *BMC Res. Notes* **7**(1), 869 (2014)
38. Weitschek, E., Velzen, R., Felici, G., Bertolazzi, P.: Blog 2.0: a software system for character-based species classification with DNA barcode sequences. What it does, how to use it. *Mol. Ecol. Resour.* **13**(6), 1043–1046 (2013)
39. Witten, I.H., Frank, E., Hall, M.A., Pal, C.J.: *Data Mining: Practical Machine Learning Tools and Techniques*. Morgan Kaufmann, San Francisco (2016)

40. Yang, X., Gao, L., Zhang, S.: Comparative pan-cancer DNA methylation analysis reveals cancer common and specific patterns. *Brief. Bioinform.* **18**, 764–773 (2016). <https://doi.org/10.1093/bib/bbw063>
41. Zeng, Y., Cullen, B.R.: Sequence requirements for micro RNA processing and function in human cells. *RNA* **9**(1), 112–123 (2003)
42. Zhu, Y., et al.: Quantitative and correlation analysis of the DNA methylation and expression of DAPK in breast cancer. *PeerJ* **5**, e3084 (2017)



Classifying Leukemia and Gout Patients with Neural Networks

Guryash Bahra and Lena Wiese(✉)

Institute of Computer Science, University of Göttingen, Göttingen, Germany
g.bahra@stud.uni-goettingen.de, wiese@cs.uni-goettingen.de

Abstract. Machine Learning is one of the top growing fields of recent times and is applied in various areas such as healthcare. In this article, machine learning is used to study the patients suffering from either gout or leukemia, but not both, with the use of their uric acid signatures. The study of the uric acid signatures involves the application of supervised machine learning, using an artificial neural network (ANN) with one hidden layer and sigmoid activation function, to classify patients and the calculation of the accuracy with k-fold cross validation. We identify the number of nodes in the hidden layer and a value for the weight decay parameter that are optimal in terms of accuracy and ensure good performance.

1 Introduction

In medical data analysis, machine learning is a common procedure used for classification of patients suffering from different diseases. In this paper, our focus is on the classification of patients suffering from either leukemia or gout. One of the common factors in leukemia and gout diseases is the uric acid signature in blood. Uric acid concentration in a healthy human in developed countries ranges from 3.5 mg/dl (in infants) to about 6 mg/dl (in adults) [1, 9, 17]. In patients suffering from gout or leukemia the uric acid concentration increases more than the normal range and is therefore regularly monitored and treated. In gout, a combination of genetic mutations and environmental factors causes uric acid concentration to increase. This further results in formation of uric acid crystals which precipitates into joints and causes painful arthritis [9]. As for leukemia, turnover of white blood cells increases the uric acid concentration. The two diseases have different pathophysiology, which – in combination with their treatments – results in different signatures of uric acid concentrations. In this article, supervised learning is performed on uric acid measurements to classify the two diseases, leukemia and gout.

1.1 Machine Learning Techniques

Machine learning involves building of models from the given dataset which can be utilized to make future predictions. This process is executed in two phases:

(i) calculation of unknown dependencies from the input dataset and (ii) prediction of new outputs using those dependencies.

The two common types of machine learning are supervised and unsupervised learning. Supervised learning involves a labelled set of input data to predict the output. In contrast, unsupervised learning involves unlabelled data; because of this, there is no designated output, which implies that the learning model has to identify patterns in the input data. The work in this article is based on a classification problem of supervised learning, which involves categorizing the data into finite classes. More precisely, uric acid concentrations are classified into two classes, leukemia and gout.

1.2 Objectives

The main objective of our work is to perform supervised machine learning, with neural networks, to estimate the accuracy of the system to distinguish between the patients with either gout or leukemia. To achieve this objective, several tasks have to be performed and these are summarized as follows:

- To identify files to be used from Medical Information Mart for Intensive Care (MIMIC) dataset [5, 6].
- To identify data (patients with either gout and leukemia and their uric acid signatures) required for the study.
- To perform supervised learning with 3-fold cross validation on uric acid measurements.

2 Related Work

A wealth of research is done in healthcare with the use of machine learning. We survey some approaches here. In [2] the authors propose an algorithm, BIG-BIOCL, for the classification of DNA Methylation Datasets for identifying cancer drivers in patients suffering from either breast, kidney or thyroid carcinomas. Supervised learning is used in [16] for cardiovascular risk prediction; the following algorithms are used: random forest, logistic regression, gradient boosting machines and neural networks. The authors conclude that neural networks performed better than the rest. In [8], the authors review different supervised learning methods, Artificial Neural Networks (ANNs), Bayesian Networks (BNs), Support Vector Machines (SVMs) and Decision Trees (DTs), for prognosis of cancer and its prediction. Their paper even highlights the case studies used for machine learning tools to predict cancer susceptibility, cancer recurrence and cancer survival. In [9] unsupervised feature learning is used on uric acid signatures before supervised learning is applied to classify the patients into gout or leukemia. The work described in their paper is implemented on data taken from Electronic Medical Records [13]. In their paper, it is mentioned that the data is noisy, sparse and irregular, therefore, it is smoothened with the use of Gaussian process regression. On this data, deep learning is performed with the use

of Sparse Autoencoders. Furthermore, the features learned from first and second layers of Sparse Autoencoders, are then utilized for the supervised learning classification task using Logistic Regression.

These papers highlight the use of machine learning in healthcare. In particular, classification of diseases is sought-after, and neural networks are widely used for classification.

3 Data Set

MIMIC-III is the third iteration of a large clinical database MIMIC. It comprises of the medical data of patients admitted to critical care units, Coronary Care Unit (CCU), Cardiac Surgery Recovery Unit (CSRU), Medical Intensive Care Unit (MICU), Neonatal Intensive Care Unit (NICU), Surgical Intensive Care Unit (SICU) and Trauma Surgical Intensive Care Unit (TSICU), at the Beth Israel Deaconess Medical Center in Boston [5,6]. According to Goldberger et al. [5], the current version of the database is 1.4 as of September 4, 2016, and consists of health-related records of de-identified 46,520 subjects out of which 38,645 are adults and 7,875 are neonates. The patients are de-identified according to Health Insurance Portability and Accountability Act (HIPAA) standards, which involved removal of 18 fields, such as patients' name, telephone numbers, addresses, etc., as listed in HIPAA. It also involved shifting of the dates, including date of birth, by a random offset, as directed by the HIPAA. The database not only includes the information of the vital sign measurements, medicines administered, laboratory measurements, fluid balance, imaging reports, out-of-hospital mortality but also patients' demographics, nurses' and physicians' notes, procedure and diagnostic codes, and more. Note that the MIMIC database was prepared by compiling data from two data sources, CareVue and Metavision Intensive Care Unit (ICU) databases, used at the hospital.

3.1 Dataset Identification

There are 26 Comma Separated Values (CSV) files in the MIMIC-III data set. And, out of those 26 files, the following 4 files are considered for the case study:

- `D.ICD.DIAGNOSES`: gives the ICD-9 codes for gout and leukemia diagnoses
- `DIAGNOSES.ICD`: identifies the hospital admissions and patients suffering from gout and leukemia
- `D.LABITEMS`: gives the ID for uric acid signatures
- `LABEVENTS`: gives the data about uric acid measurements of the patients identified with gout and leukemia.

There are 78 ICD-9 codes for leukemia and 11 for gout identified from the `D.ICD.DIAGNOSES` table. Then, from the `DIAGNOSES.ICD` table, a total of 2,837 hospital admissions are identified with above diagnoses, and out of which 618 hospital admissions are for leukemia and 2,219 are for gout. These many admissions correspond to 2,259 patients or unique¹ `SUBJECT_IDS`, out of which 454

¹ In R, *deduplicated* function with logical negation operator (!) is used to find unique `SUBJECT_IDS`.

patients suffered from leukemia and 1,805 suffered from gout. Furthermore, 22 patients are common for both the diagnoses, and after removing IDs of those patients, a total of 2,773 hospital admissions are identified for the patients suffering from either leukemia (584 HADM_IDS) or gout (2,189 HADM_IDS) but not both. The 2,773 admissions correspond to 2,215 patients, out of which 1,783 patients suffered from gout and 432 from leukemia. Moreover, the hospital admissions are reduced from 2,773 to 1,076 as there are no records of uric acid measurements for those patients. The number, 1,076, further decreased to 640 as there are no uric acid observations corresponding to those admissions. And finally, these 640 unique admissions correspond to 567 unique patients, out of which 311 suffered from gout and the remaining 256 patients suffered from leukemia.

As for the uric acid signatures, 3 IDs are identified from the D_LABITEMS table. And, corresponding to those IDs, 19,906 observations representing uric acid measurements are identified from the LABEVENTS table. Then, 19,906 observations reduced to 7,076, as the removed observations didn't correspond to the identified SUBJECT_IDS. Furthermore, 7,076 observations reduced to 5,665, as those observations did not correspond to the identified hospital admission IDs.

3.2 Dataset Creation

The data, i.e., uric acid concentrations, are arranged into 567 sequences, grouped according to the patient IDs. These sequences are then broken down to a size of 17 values per row: the first two values are the label (1 for leukemia and 0 for gout) and patient ID, and the remaining 15 values are the uric acid concentrations. Note that the sizes of sequences are unequal. Therefore, there can be multiple rows of data of a single patient, and each row in turn is treated as a new sequence. And, for sequences with less than 15 data values, 0 value is used for the remaining part of the sequence. This resulted in a total of 813 sequences. The sequences are then shuffled with the use of `sample` function provided by R [15].

3.3 K-Fold Cross-validation

To create training and testing sets used for the learning, the k-fold cross validation method is employed. In k-fold cross validation, the data is randomly divided into k subsets of equal size and a single subset is referred to as fold. Of the k folds, $k - 1$ folds are combined to form the training set and the remaining fold is used as the testing set, and the accuracy is calculated for the training and the testing sets which describes the stability of the model. This is then repeated for k iterations, and for every iteration, the testing set comprises of a fold used exactly once. Moreover, to use the model for new predictions and to estimate the overall accuracy of the model, consider the classifier for which the highest accuracy is achieved for the testing set.

As described in the previous paragraph, first the data is divided into equal subsets. Therefore, 813 sequences (from Sect. 3.2) are divided into 3 equal subsets of size 271 each. Then, supervised learning (as described in Sect. 4.2) is performed on these subsets for three iterations. For each iteration, the testing set is formed

with a single subset used exactly once and the remaining two subsets are used as the training set (of size 542). The accuracy (calculated as in Sect. 4.2) is reported for every iteration.

4 Method

4.1 Neural Networks

Neural networks are one of several supervised learning classification techniques, and are based on the concept of perceptrons [7, 12]. Neural networks are conceptualized as in the following paragraphs.

Forward Propagation. In this work, a 3-layered neural network is used, where the first layer, L_1 , is the input layer, the second layer, L_2 , is the hidden layer, and the last layer, L_3 , is the output layer. Note that the output of one layer is the input of the next layer, and there are s_l number of nodes in each layer l .

Activation unit, a_i^l , is used to define the output of the i th unit in layer l . Therefore, for the L_1 layer, $a_i^{(1)} = x_i$, where x is the input data. As for the layers L_2 and L_3 , the nodes are computational and therefore, activations are calculated as a function of input vector x , weights matrix W and a bias vector b , as given by the Eq. 1.

$$a_i^{(l)} = f\left(W_{ij}^{(l-1)} a_j^{(l-1)} + b_i^{(l-1)}\right) \quad (1)$$

In Eq. 1, W_{ij}^{l-1} is the weight or the parameter of the connection between the j th unit in layer $l-1$ and the i th unit in layer l , bias unit b_i^l corresponds to the i th unit in layer l . Function $f : R \rightarrow R$ is the activation function, and is usually defined with sigmoid function as shown by the Eq. 2; it produces an output ranging from (0, 1).

$$f(z) = \frac{1}{1 + \exp(-z)} \quad (2)$$

As the nodes are calculated starting from the layer L_1 up to layer L_3 in the network, this step is called forward propagation.

An important identity to note is that the derivative $f'(z)$ of sigmoid function $f(z)$ (in Eq. 2) is given by the Eq. 3, and is used later in the section.

$$f'(z) = f(z)(1 - f(z)) \quad (3)$$

Cost Function. The cost function we use is known as the squared-error cost function. For a given training set $\{(x^{(1)}, y^{(1)}), \dots, (x^{(r)}, y^{(r)}), \dots, (x^{(m)}, y^{(m)})\}$ of m training examples, the cost function is given as Eq. 4,

$$J(W, b) = \left[\frac{1}{m} \sum_{r=1}^m J(W, b; x^{(r)}, y^{(r)}) \right] + \frac{\lambda}{2} \sum_{l=1}^{n_l-1} \sum_{i=1}^{s_l} \sum_{j=1}^{s_{l+1}} \left(W_{ij}^{(l)} \right)^2 \quad (4)$$

where the first term is the average sum-of-squares error term and the second term is the *regularisation* term (or the *weight decay* term) which decreases the value of the weights and avoids overfitting. Note that the regularisation term is not applied to bias b [10]. And, λ in Eq. 4 is the *weight decay parameter*.

To minimize the cost function $J(W, b)$ as a function of weights matrix W and bias vector b , every parameter $W_{ij}^{(l)}$ and $b_i^{(l)}$ is initialized to a random value near to 0. And then an optimization algorithm, for example gradient descent, is applied for minimization.

Gradient Descent Algorithm. Gradient descent updates the parameters per iteration as mentioned in Eq. 5, where α is the learning rate and $\frac{\partial}{\partial W_{ij}^{(l)}} J(W, b)$ and $\frac{\partial}{\partial b_i^{(l)}} J(W, b)$ are derivatives of the overall cost function $J(W, b)$. Parameters are updated as

$$\begin{aligned} W_{ij}^{(l)} &= W_{ij}^{(l)} - \alpha \frac{\partial}{\partial W_{ij}^{(l)}} J(W, b) \\ b_i^{(l)} &= b_i^{(l)} - \alpha \frac{\partial}{\partial b_i^{(l)}} J(W, b) \end{aligned} \quad (5)$$

Back Propagation Algorithm. To calculate the partial derivatives, $\frac{\partial}{\partial W_{ij}^{(l)}} J(W, b)$ and $\frac{\partial}{\partial b_i^{(l)}} J(W, b)$, as mentioned in Eq. 5, back propagation algorithm is applied and is described as follows. The first step is to calculate the error term δ for every computational node, starting from layer L_3 , or the output layer, to layer L_2 , in the network. As stated in [10], “the error term measures how much the node is responsible for any errors in the output”. Finally, the parameters, W and b , which minimize the cost function, $J(W, b)$, are calculated with the gradient descent algorithm.

4.2 Implementation of Supervised Learning Using Neural Networks

The steps carried out to perform supervised learning (in Octave [4]) are described in the following paragraphs.

Define s_l and λ . To begin, the nodes s_l in all the layers (in Eq. 4) and the weight decay parameter λ (in Eq. 4) are defined. According to our dataset (from

Sect. 3.2), the input size is 15, that is, the number of nodes (s_1) in layer 1 (L_1) is 15. This is because, from Sect. 3.2, out of 17 values per row, 15 values are the uric acid measurements. The number of output nodes is 1, as the label (leukemia or gout) per row is single-valued (from Sect. 3.2). The number of nodes in hidden layer s_2 and the value of weight decay parameter λ is varied in order to assess the changes (positive, negative, or no change) in the result.

Initialization of Weights W and Biases b . The weights in matrix W are to be initialized to a value close to 0 [3] and therefore, are randomly initialized from the interval $[-0.5, 0.5]$ [11]. The biases b are initialized to 0.

Cost Function Calculation. The calculation of cost function $J(W, b)$ (as in Sect. 4.1) is implemented according to the pseudo-code below.

1. **compute activation matrix $a^{(l)}$**

Activation matrix is computed according to Eq. 1.

a. for layer $l = 2$: $a^{(2)} = f(W^{(1)} * (data)^T) + b^{(1)}$

Note: $(data)^T$ is transpose of $data$ and f is sigmoid function (described by Eq. 2).

b. for layer $l = 3$: $a^{(3)} = f(W^{(2)} * a^{(2)}) + b^{(2)}$

2. **calculate weight regularisation term**

All the weights in the layers, $l = 1, 2$, are added, and multiplied with $\frac{\lambda}{2}$ (as described by Eq. 4).

3. **calculate cost function**

The cost function $J(W, b)$ is determined.

a. $cost = 0$

b. for $i = 1 \dots m$: $cost = cost + \frac{1}{2}(a^{(3)}(i) - y(i))^2$

c. $J(W, b) = \frac{1}{m}cost + weight\ regularisation\ term$

Note: the number of training examples (from Sect. 3.3) is $m = 542$, $cost$ is a temporary variable, y is label with value either 0 (for gout) or 1 (for leukemia), and $weight\ regularisation\ term$ is from the previous step.

Subsequently, the Limited-memory-Broyden-Fletcher-Goldfarb-Shanno (L-BFGS) method is used for minimization of the cost function. The L-BFGS algorithm is implemented by the function `minFunc` (by Schmidt [14]).

Calculation of Derivatives. The calculation of partial derivatives, $\frac{\partial}{\partial W_{ij}^{(l)}}J(W, b)$ and $\frac{\partial}{\partial b_i^{(l)}}J(W, b)$, as described in Sect. 4.1, is implemented according to the pseudo-code below.

1. **compute error terms $\delta^{(l)}$**

a. for the output layer $l = 3$: $\delta^{(3)} = -(y - a^{(3)}) \cdot a'^{(3)}$

Note: a' is the derivative of the sigmoid activation function. On using the identity of sigmoid function, as in Eq. 3, $\delta^{(3)}$ in Step a is defined as:

$$\delta^{(3)} = -(y - a^{(3)}) \cdot a^{(3)} \cdot (1 - a^{(3)})$$

b. for layers $l = 2$: $\delta^{(2)} = (W^{(2)})^T \delta^{(3)} \cdot a'^{(3)}$

Note: $(W)^T$ is transpose of weight matrix W and a' is again the derivative of the sigmoid activation function.

2. compute partial derivatives of the overall cost function $J(W, b)$

a. for layer $l = 2$: $\frac{\partial}{\partial W^{(2)}} J(W, b) = \frac{1}{m} \delta^{(3)} (a^{(2)})^T + \lambda W^{(2)}$ and $\frac{\partial}{\partial b^{(2)}} J(W, b) = \frac{1}{m} \delta^{(3)}$

b. for layer $l = 1$: $\frac{\partial}{\partial W^{(1)}} J(W, b) = \frac{1}{m} \delta^{(2)} data + \lambda W^{(1)}$ and $\frac{\partial}{\partial b^{(1)}} J(W, b) = \frac{1}{m} \delta^{(2)}$

Here, for the input layer L_1 , $a^{(1)} = data$ as mentioned in Sect. 4.1. Note that in Steps **a** and **b**, δ is the error term computed in the previous step.

Update Parameters W and b . The weights matrix W and the bias vector b for layers L_1 and L_2 , which minimize the cost function $J(W, b)$, are recalibrated after every iteration of the L-BFGS algorithm. In other words, W and b are equal to final values of the partial derivatives of the overall cost function, $\frac{\partial}{\partial W^{(l)}} J(W, b)$ and $\frac{\partial}{\partial b^{(l)}} J(W, b)$ respectively.

Calculate Accuracy. To calculate the accuracy of the machine learning model on the training and the testing sets, forward propagation (as described in Sect. 4.1) is performed such that for layers $l = 2, 3$: $a^{(l)} = f(W^{(l-1)} * a^{(l-1)}) + b^{(l-1)}$. Note that the activation matrix of layer 1, $a^{(1)} = data$ (either training or testing), f is sigmoid function, and W and b are calculated using the L-BFGS algorithm.

Then, the activation vector of the output layer, $a^{(3)}$, is used to assign labels to the data (either training set or testing). If the value of an element in $a^{(3)}$ is greater or equal to 0.5, then, label 1 (corresponding to leukemia) is assigned to the element, else label 0 (corresponding to gout) is assigned. These assigned labels then form the prediction vector of size 542×1 in case of training set and of 271×1 in case of testing set.

Each element in prediction vector is then compared with the corresponding actual label of the data. If the labels are the same, 1 is assigned to a comparison vector; if labels are not the same, then 0 is assigned to the comparison vector. Furthermore, the average is calculated for the comparison vector, which is of size 542×1 in case of training set and of 271×1 in case of testing set. The average multiplied with 100 gives the accuracy of the model in percent.

5 Results

This section describes the results of supervised learning. The results represent the neural network model's ability to distinguish between patients suffering from either gout or leukemia. The accuracies are determined for 5 different cases, resulting from the change in the values of weight decay parameter λ and number

of hidden layer nodes s_2 (as in Sect. 4.2). Case 1 is where $s_2 = 10$ & $\lambda = 0.0001$; Case 2 is $s_2 = 25$ & $\lambda = 0.0001$; Case 3 is $s_2 = 5$ & $\lambda = 0.0001$; Case 4 is $s_2 = 5$ & $\lambda = 0.00001$; Case 5 is $s_2 = 5$ & $\lambda = 0.000001$.

The accuracy is computed three times (I1–I3) per case; this is because weights in W are randomly initialized (see Sect. 4.2) and therefore, give slightly different values for each iteration. For the final accuracy with 3-fold cross validation (measured in percent), the accuracies are averaged out (Avg).

Table 1. Accuracies (in %) of neural network with 3-folds cross validation.

Case		Cross validation on original dataset							
		Test set: fold 1		Test set: fold 2		Test set: fold 3		Average	
		Train	Test	Train	Test	Train	Test	Train	Test
Case 1: $s_2 = 10$, $\lambda = 0.0001$	I1	88.74	84.50	90.59	81.54	91.32	78.22	90.22	81.42
	I2	88.74	86.34	90.03	83.39	90.77	76.01	89.84	81.91
	I3	88.56	85.97	90.40	83.39	90.95	78.22	89.97	82.52
	Avg	88.68	85.60	90.34	82.77	91.01	77.48	90.01	81.95
Case 2: $s_2 = 25$ $\lambda = 0.0001$	I1	89.48	85.23	90.77	82.65	91.51	78.22	90.58	82.03
	I2	89.48	86.71	91.32	80.81	91.69	79.70	90.83	82.41
	I3	89.48	85.23	91.14	81.91	91.88	78.96	90.83	82.03
	Avg	89.48	85.72	91.07	81.79	91.69	78.96	90.74	82.15
Case 3: $s_2 = 5$ $\lambda = 0.0001$	I1	85.60	84.87	87.82	82.28	89.66	81.91	87.69	83.02
	I2	85.60	85.97	88.56	83.02	89.66	78.59	87.94	82.52
	I3	85.42	85.23	88.37	83.02	89.29	78.22	87.69	82.15
	Avg	85.54	85.35	88.25	82.77	89.53	79.57	87.77	82.56
Case 4: $s_2 = 5$ $\lambda = 0.00001$	I1	86.71	86.71	88.37	83.02	90.40	79.33	88.49	83.02
	I2	87.26	83.02	87.26	83.39	86.71	83.02	87.07	83.14
	I3	88	87.08	88.56	80.81	89.66	77.49	88.74	81.79
	Avg	87.32	85.60	88.06	82.40	88.92	79.94	88.1	82.65
Case 5: $s_2 = 5$ $\lambda = 0.000001$	I1	86.16	85.60	85.60	83.39	87.63	80.81	86.46	83.26
	I2	85.42	85.60	88	83.02	90.22	78.59	87.88	82.40
	I3	86.16	85.60	88.19	82.65	90.22	77.49	88.19	81.91
	Avg	85.91	85.60	87.26	83.02	89.35	78.96	87.51	82.52

From Table 1 we can observe that (although the highest average training set accuracy is 90.74 in case 2) the highest average testing set accuracy is 82.65 in case 4. Hence the settings of case 4 seem to be the best out of all cases.

We implemented the steps to carry out the supervised learning presented in the previous sections in Octave [4]. Data preprocessing was done in R. We measured the runtime of the neural network model learning phases for all 5 cases. Executions are run on a Ubuntu 16.04.2 LTS system with 8 GB RAM, 64 bit intel core i5 processor and 1 TB of hard disk. Table 2 shows that case 4 indeed provides a good average execution time.

Table 2. Execution time (in seconds) for k-fold cross validation.

Case	Runtime (s)
Case 1: when $s_2 = 10$ & $\lambda = 0.0001$	30.03
Case 2: when $s_2 = 25$ & $\lambda = 0.0001$	179.76
Case 3: when $s_2 = 5$ & $\lambda = 0.0001$	16.08
Case 4: when $s_2 = 5$ & $\lambda = 0.00001$	22.07
Case 5: when $s_2 = 5$ & $\lambda = 0.000001$	27.72

6 Discussion and Conclusion

In our experiment a neural network was designed with one hidden layer to classify gout and leukemia patients based on their uric acid measurements. The optimal settings that we identified comprise the lowest number of nodes in the hidden layer as well as a medium value for the weight decay parameter. These settings also provide a good runtime.

Our experiment starts with identifying the tables from the MIMIC-III database. Then, from those tables, patients with gout and leukemia diseases, and their corresponding uric acid measurements, are identified. The data is then cleansed, and is further used for supervised learning. The neural network (in Sects. 4.1 and 4.2) for the supervised learning is designed using one hidden layer and sigmoid activation function. Accuracy, is then calculated to measure the effectiveness of the model.

In future work we will investigate whether using more layers improves the accuracy. Using tanh activation function, instead of sigmoid activation function, and observing its effect on the accuracy is another enhancement we plan to study. Moreover, verification with a larger dataset will be necessary to validate our results.

References

1. Alvarez-Lario, B., MacArron-Vicente, J.: Is there anything good in uric acid? QJM: Int. J. Med. **104**, 1015–1024 (2011)
2. Celli, F., Cumbo, F., Weitschek, E.: Classification of large DNA methylation datasets for identifying cancer drivers. Big Data Res. (2018). <https://www.sciencedirect.com/science/article/pii/S2214579617302708?via%3Dihub>
3. Changhau, I.: Weight Initialization in Artificial Neural Networks (2017). <https://isaacchanghau.github.io/2017/05/24/Weight-Initialization-in-Artificial-Neural-Networks/>
4. Eaton, J.W., Bateman, D., Hauberg, S., Wehbring, R.: GNU Octave version 4.2.0 manual: a high-level interactive language for numerical computations (2016). <http://www.gnu.org/software/octave/doc/interpreter>
5. Goldberger, A.L., et al.: PhysioBank, PhysioToolkit, and PhysioNet: components of a new research resource for complex physiologic signals. Circ. Electron. **101**, e215–e220 (2000)

6. Johnson, A.E., et al.: MIMIC-III, a freely accessible critical care database. *Sci. Data* **24**, 1600355 (2016)
7. Kotsiantis, S.B.: Supervised machine learning: a review of classification techniques. In: Maglogiannis, I.G., Karpouzis, K., Wallace, M. (eds.) *Emerging artificial intelligence applications in computer engineering: real word AI systems with applications in eHealth, HCI, Information Retrieval and Pervasive Technologies*, pp. 3–24. IOS Press (2007)
8. Kourou, K., Exarchos, T.P., Exarchos, K.P., Karamouzis, M.V., Fotiadis, D.I.: Machine learning applications in cancer prognosis and prediction. *Comput. Struct. Biotechnol. J.* **13**, 8–17 (2015)
9. Lasko, T.A., Denny, J.C., Levy, M.A.: Computational phenotype discovery using unsupervised feature learning over noisy, sparse, and irregular clinical data. *PLOS ONE* **8**(8) (2013). <http://journals.plos.org/plosone/article?id=10.1371/journal.pone.0066341>
10. Ng, A., Ngiam, J., Foo, C.Y., Mai, Y., Suen, C.: UFLDL Tutorial (2013). http://deeplearning.stanford.edu/wiki/index.php/UFLDL_Tutorial
11. Nguyen, D., Widrow, B.: Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights. In: *IJCNN International Joint Conference on Neural Networks*, vol. 3 (1990)
12. Nielsen, M.A.: *Neural Networks and Deep Learning*. Determination Press (2015). <http://neuralnetworksanddeeplearning.com/>
13. Roden, D.M., et al.: Development of a large-scale de-identified DNA biobank to enable personalized medicine. *Clin. Pharmacol. Ther.* **84**, 362–369 (2008)
14. Schmidt, M.: minFunc: unconstrained differentiable multivariate optimization in Matlab (2005). <https://www.cs.ubc.ca/~schmidtm/Software/minFunc.html>
15. Team, R.C.: *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna (2014). <https://www.r-project.org/>
16. Weng, S.F., Reps, J., Kai, J., Garibaldi, J.M., Qureshi, N.: Can machine-learning improve cardiovascular risk prediction using routine clinical data? *PLoS ONE* **12**(4), 1–14 (2017)
17. Wilcox, W.: Abnormal serum uric acid levels in children. *J. Pediatr.* **128**, 731741 (1996)



Incremental Wrapper Based Random Forest Gene Subset Selection for Tumor Discernment

Alia Fatima^(✉), Usman Qamar, Saad Rehman,
and Aiman Khan Nazir

National University of Sciences and Technology (NUST), Islamabad, Pakistan
{alia.fatimal6, aiman.nazir16}@ce.ceme.edu.pk,
{usmanq, saadrehman}@ceme.nust.edu.pk

Abstract. High-dimensional cancer related dataset permits the researchers to timely diagnose and facilitate in effective treatment of the cancer. Biomedicine application process on the thousands of features. It is challenging to extract the precise statistics from this high-dimensional dataset. This paper presents the Incremental Wrapper based Random Forest Gene Subset Selection of Tumor discernment that mechanisms on the principle of incremental wrapper based feature subset selection with random forest classification algorithm and this algorithm also works as performance validator. Incremental wrapper based feature subset selection is a technique to pick out a finest conceivable subset of genes from the high-dimensional data with low computational cost. Random Forest will increase the overall performance as it works better in cancer related high-dimensional dataset. The efficacy of the random forest classification algorithm as performance validator will significantly improve by working on a selective discriminative subset of prognostic genes as compare to the raw data. We evaluate the proposed methodology on the six publicly available cancer related high dimensional datasets and found that the proposed methodology outperform as compare to standard random forests.

Keywords: Cancer classification · Random forest · IWSS
Incremental wrapper based gene subset selection

1 Introduction

Cancer is the most serious illness in which certain cells of the body grow in an uncontrolled way. It can be cured if timely diagnosed. Nowadays, researchers are extensively working on the early diagnosis and treatment of the cancer [1]. Diverse technologies are used in this respect in which one of the most focused technique is the analysis of the disease related information. Intensive research carries on the analysis of microarrays data [2, 3]. Biology and biomedicine applications are extensively using for this purpose. These applications are generating a large number of genes which is further used for the diagnosis of the disease such as high dimensional datasets are used for the discernment of the cancer. High dimensionality is the fundamental problem for extracting and analyzing the vital information from gene-expression data [4]. These high dimensional datasets have thousands of genes.

These datasets are used in two perspectives: first to accurately diagnosis the disease and second to distinguish the particular set of genes which are responsible for the disease [4]. Feature selection techniques are used for selecting the prognostic genes. As, most of the classification algorithms give, the better results by developing the model on fewest number of genes [2].

Feature selection algorithm comprises of two parts, first one is a search technique for new feature subset and the second one is an evaluation measure to score the given feature subset. These techniques are used to increase the efficacy of the classifiers. Microarray datasets are high dimensional datasets with a large number of genes. These all genes are not relevant. Irrelevant and redundant features decreases the performance of the classifier [5].

Bioinformatics community has the deepest interest in Random forest (RF) algorithm [6] for the classification of high-dimensional and microarray dataset from the past few years [7, 8]. Recent work [8] demonstrate that random forest has better performance as compared to other best performing classifier in the cancer microarray gene expression domain. Irrelevant features present in high dimensional data set to deteriorate the performance of the learning algorithm [9].

The effective feature selection method can be used to lessen this problem by selecting a subset of discriminative features from the complete feature set [10, 11]. There are three groups for feature selection method: filter, wrapper and embedded [12]. Filter methods use the intrinsic properties of the training sets to evaluate the features and selecting a valuable feature subset. The generalization ability of the methods is better along with the computational complexity. These methods are flexible to work with diverse classifier. The wrapper method evaluates the quality of the feature subset accordingly to the embedded classifier. These methods gain the better classification performance as compared to proceeding one [13, 14]. The embedded method works on the combine qualities of the preceding two methods.

In this study, we will present a novel methodology for the discernment of the high dimensional cancer dataset. The proposed methodology will alleviate the curse of dimensionality and will increase the performance of the random forest in the high dimensional dataset for the accurate discernment of tumor. Remaining paper is arranged as follows: Sect. 2 presents literature review, Sect. 3 presents the proposed methodology, Sect. 4 presents the experimental results and paper is concluded in Sect. 5.

2 Literature Review

2.1 Random Forest

Random forest is an ensemble which is composed by a group of classification trees. Breiman first introduced the concept of random forest in early 2001 [6]. Random forest is developed by the following procedure:

n number of instances are randomly selected from the total N number of instances. These n instances sample sets are used as training dataset for developing the decision tree model.

m number of variables are selected from the total M number of variables. These variables are selected randomly. These m number of variables are used at each node for splitting the node on the basis of best splitting criteria. m remains constant throughout the growth of the trees. Trees are grown on the maximum possible size and there is no pruning.

New objects are classified on the basis of input vector. This input vector is evaluated on the basis of each already developed a decision tree model in the forest. Each tree gives a classified class result and final result selected on the basis of the majority voting.

Forest error rate depends on following two attributes according to the original work.

Forest error rate rise with the increase of the correlation value or similarity between any pair of trees in the forest.

A tree is considered a good classifier with a low error rate.

2.2 Incremental Wrapper Based Feature Subset Selection

Ruiz et al. [15] proposed the incremental wrapper based feature subset (IWSS) algorithm. IWSS algorithm consists of two phases. In the first phase, it uses the feature ranking algorithm for scoring function. The scoring function assigns the score to the individual feature on the basis of computing the score from the values of each feature and class label. Features are ranked in the list in decreasing order as, feature with the highest score is first in the list and so on.

In the second phase, algorithm selects the feature with the highest score and make it the best feature subset after computing the accuracy with the selected learner. It computes the accuracy of the next feature subset which is developed by inducing the next highest scoring feature in the existing subset. This subset is selected as the best feature subset if its accuracy is greater than the previously computed accuracy. This Process is repeated until the last feature is evaluated.

Input: D training U-measure, L-classifier Output: Best Subset
<pre> List R={} For each gene gi ∈ D Score = compute (gi, U, D) append gi to R according to Score BestClassif = 0 BestSubset = ∅ For i =1 to N TempSubset = BestSubset ∪ {gi} (gi ∈ R) TempClassif = WrapperClassif(TempSubset, L) if(TempClassif > BestClassif) BestSubset = TempSubset BestClassif = TempClassif </pre>

Algorithm. 1. Incremental Wrapper Based Feature Subset Selection

2.3 OneRAttributeEval

OneRAttributeEval evaluates the worth of an attribute by using the OneR classifier.

2.4 OneR

OneR [16] is a classification algorithm. It is a short form of One Rule. Holte proposed it in 1993. It is a simple for humans to interpret, but accurate classification algorithm, which performs well on most commonly used datasets. It generates one rule for each predictor in the data. Then, it selects the rule with the smallest total error as its “one rule”. A frequency table is constructed for each predictor against the target to create a rule for a predictor.

OneR Algorithm.

- For each predictor,
- For each value of that predictor, make a rule as follows;
 - Count how often each value of target (class) appears
 - Find the most frequent class
 - Make the rule assign that class to this value of the predictor
- Calculate the total error of the rules of each predictor
- Choose the predictor with the smallest total error.

3 Proposed Methodology

Random forest performs well on high-dimensional dataset as compare to other classifiers. In this work, we are proposing a novel methodology for the cancer classification on the basis of high dimensional dataset. This work proposes to use the incremental wrapper based feature subset selection with OneRAttributeEval and Random Forest for the improved and accurate results of prognosis of cancer. This methodology comprises of two phases. In the first phase, it incrementally selects the feature subset and in a second phase, it evaluates the accuracy of the classifier with a selected subset of features for tumor discernment.

Phase 1: Incremental wrapper based feature subset is used to select the relevant features by eliminating the redundant features. It incrementally selects the feature subset and OneRAttributeEval algorithm evaluates the worth of an attribute by using the OneR Classifier and features are ranked on the basis of their worth. The performance of the features is evaluated on the basis of the algorithm accuracy. Random forest is used to evaluate the given subset of the features.

Phase 2: In this phase, a selected subset of features is evaluated and accuracy of the dataset is evaluated by using the random forest (Fig. 1).

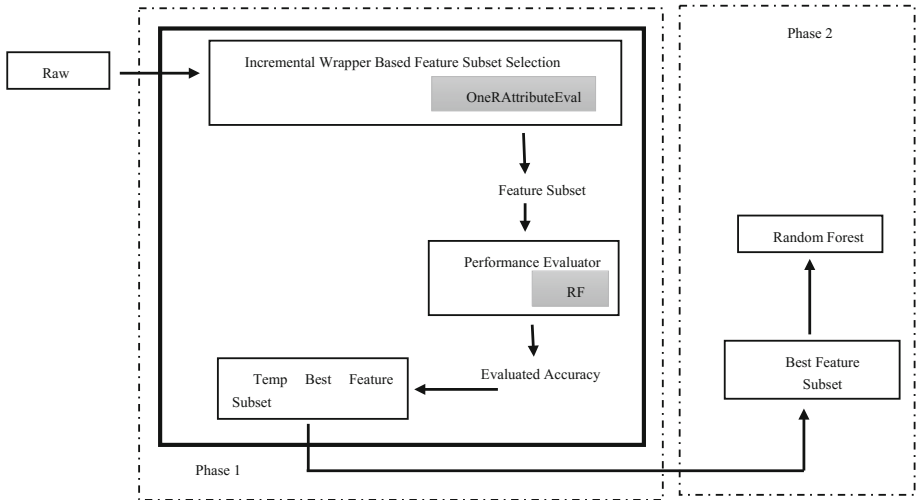


Fig. 1. Framework of proposed approach: wrapper based RF gene subset selection for tumor discernment

4 Experimental Results

We have performed the experimental evaluation of our proposed approach on Weka 3-8. Our evaluation environment worked on the machine which has i3 core processor, 6 GB RAM, 64 bit operating system and Windows 8.1. Table 1 presents the features of the selected cancer related datasets, such as no. of attributes and no. of instances.

Table 1. Dataset used.

Dataset name	No. of attribute	No. of instances
Colon	2000	62
GCM	16064	190
Leukemia	7130	72
Lung-Cancer	56	32
Lymphoma	4027	96

Leukemia dataset is obtained from [17], Lung-Cancer datasets are obtained from UCI Repository [18], and Colon, Lymphoma and GCM datasets are obtained from [19]. We used accuracy parameter as the performance measure of our proposed methodology.

Accuracy is used to predict the class label. Accuracy is defined as,

$$Accuracy = \frac{TruePositive + TrueNegative}{TruePositive + TrueNegative + FalsePositive + FalseNegative}$$

The experimental results of our proposed approach are shown in Table 2 on the basis of the above mentioned cancer related datasets.

Table 2. Results of the proposed approach.

Dataset name	Selected no. of attribute	Accuracy
Colon	4	87.096%
GCM	23	62.1053%
Leukemia	4	91.67%
Lung-Cancer	3	68.75%
Lymphoma	13	85.4167%

We compared the results of our proposed approach with the standard Random Forest Algorithm. Comparison of the standard Random Forest Algorithm and our proposed Approach is shown in Table 3.

Table 3. Comparison of the proposed approach and standard random forest.

Dataset name	Standard RF	Proposed approach
Colon	83.87%	87.096%
GCM	61.0526%	62.1053%
Leukemia	90.27%	91.67%
Lung-Cancer	46.75%	68.75%
Lymphoma	84.375%	85.4167%

Above mentioned results demonstrate that our proposed methodology performs well in cancer related high-dimensional dataset as compared to the standard Random Forest by selecting the minimum prognostic genes.

5 Conclusion

Cancer is found to be a most killing disease over the globe. Cancer diagnosis and treatment related biomedicine applications work on the high dimensional dataset. These datasets consist of the number of redundant features. It is important to remove the redundant and irrelevant features from the high dimensional dataset to get the valuable statistics. This paper presents a novel methodology: Incremental wrapper based random forest gene subset selection for tumor discernment. This methodology comprises of two phases. In the first phase, minimum relevant prognostic subset of genes is selected and in the second phase, Random Forest is used for the classification of the dataset as, it is found to perform well in the classification of disease related high dimensional dataset. The performance of the proposed methodology is evaluated on the basis of six high dimensional cancer related datasets. Accuracy is used as the evaluation measure of the performance. We compare the results of the proposed approach with the standard

Random forest and found that proposed methodology is giving the accurate results as compare to standard random forest with the less number of genes. In the future, we will prefer to work on the embedded random forest feature subset selection to decrease the computational cost and time complexity.

References

1. Ahmad, F., Isa, N.A.M., Hussain, Z., Osman, M.K., Sulaiman, S.N.: A GA-based feature selection and parameter optimization of an ANN in diagnosing breast cancer. *Pattern Anal. Appl.* **18**, 861–870 (2015)
2. Mishra, D., Sahu, B.: Feature selection for cancer classification: a signal-to-noise ratio approach. *Int. J. Sci. Eng. Res.* **2**, 1–7 (2011)
3. Deng, L., Pei, J., Ma, J., Lee, D.L.: A rank sum test method for informative gene discovery. In: *Proceedings of the Tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 410–419 (2004)
4. Dasgupta, S., Saha, G., Mondal, R.: A comparison between methods for generating differentially expressed genes from microarray data for prediction of disease. In: *Proceedings of the 2015 Third International Conference on Computer, Communication, Control and Information Technology (C3IT)* (2015)
5. Hua, J., Tembe, W.D., Dougherty, E.R.: Performance of feature-selection methods in the classification of high-dimension data. *Pattern Recogn.* **42**, 409–424 (2009)
6. Breiman, L.: Random forests. *Mach. Learn.* **45**, 5–32 (2001)
7. Wu, B., et al.: Comparison of statistical methods for classification of Ovarian cancer using mass spectrometry data. *Bioinformatics* **19**, 1636–1643 (2003)
8. Díaz-Uriarte, R., De Andres, S.A.: Gene selection and classification of microarray data using random forest. *BMC Bioinform.* **7**, 3 (2006)
9. Shu, W., Shen, H.: Incremental feature selection based on rough set in dynamic incomplete data. *Pattern Recogn.* **47**, 3890–3906 (2014)
10. Prabhakar, S., Jain, A.K.: Decision-level fusion in fingerprint verification. *Pattern Recogn.* **35**, 861–874 (2002)
11. Gheyas, I.A., Smith, L.S.: Feature subset selection in large dimensionality domains. *Pattern Recogn.* **43**, 5–13 (2010)
12. You, W., Yang, Z., Ji, G.: PLS-based recursive feature elimination for high-dimensional small sample. *Knowl. Based Syst.* **55**, 15–28 (2014)
13. Kohavi, R., John, G.H.: Wrappers for feature subset selection. *Artif. Intell.* **97**, 273–324 (1997)
14. Inza, I., Larrañaga, P., Blanco, R., Cerrolaza, A.J.: Filter versus wrapper gene selection approaches in DNA microarray domains. *Artif. Intell. Med.* **31**, 91–103 (2004)
15. Ruiz, R., Riquelme, J.C., Aguilar-Ruiz, J.S.: Incremental wrapper-based gene selection from microarray data for cancer classification. *Pattern Recogn.* **39**, 2383–2392 (2006)
16. Holte, R.C.: Very simple classification rules perform well on most commonly used datasets. *Mach. Learn.* **11**, 63–91 (1993)
17. Cancer program data sets (2010). Broad Institute. <http://www.broadinstitute.org/cgi-bin/cancer/datasets.cgi>
18. Frank, A., Asuncion, A.: UCI machine learning repository (2010). <http://archive.ics.uci.edu/ml>
19. Dataset repository in ARFF (weka) (2010). BioInformatics Group Seville. <http://www.upo.es/eps/big5/datasets.html>



Protein Identification as a Suitable Application for Fast Data Architecture

Roman Zoun¹(✉), Gabriel Campero Durand¹, Kay Schallert²,
Apoorva Patrikar³, David Broneske¹, Wolfram Fenske¹, Robert Heyer²,
Dirk Benndorf², and Gunter Saake¹

¹ Working Group Databases and Software Engineering, University of Magdeburg,
39104 Magdeburg, Germany

{roman.zoun, campero, david.broneske,
wolfram.fenske, gunter.saake}@ovgu.de

² Chair of Bioprocess Engineering, University of Magdeburg,
39104 Magdeburg, Germany

{kay.schallert, robert.heyer, dirk.benndorf}@ovgu.de

³ Accenture GmbH, Kronberg im Taunus, Germany
apoorva.patrikar@ovgu.de

Abstract. Metaproteomics is a field of biology research that relies on mass spectrometry to characterize the protein complement of microbiological communities. Since only identified data can be analyzed, identification algorithms such as X!Tandem, OMSSA and Mascot are essential in the domain, to get insights into the biological experimental data. However, protein identification software has been developed for proteomics. Metaproteomics, in contrast, involves large biological communities, gigabytes of experimental data per sample, and greater amounts of comparisons, given the mixed culture of species in the protein database. Furthermore, the file-based nature of current protein identification tools makes them ill-suited for future metaproteomics research. In addition, possible medical use cases of metaproteomics require near real-time identification. From the technology perspective, Fast Data seems promising to increase throughput and performance of protein identification in a metaproteomics workflow. In this paper we analyze the core functions of the established protein identification engine X!Tandem and show that streaming Fast Data architectures are suitable for protein identification. Furthermore, we point out the bottlenecks of the current algorithms and how to remove them with our approach.

Keywords: Fast Data · Bioinformatics · Metaproteomics
Proteomics · Cloud computing · Protein identification

1 Introduction

Proteomics is the biological research of all proteins of a particular species, which brings insights about the functionality of an organism. In the proteomics

Supported by de.NBI.

workflow, the prepared biological sample is measured using a mass spectrometer. It measures the mass of the molecules of the biological input sample. The identification can be done with so called protein identification engines. These are software tools, which compare the measured mass spectra with already identified proteins from a database and create a similarity score for each comparison. Possible protein search engines are X!Tandem, OMSSA and Mascot.

The algorithms use a file of the experimental spectra data and also a textual representation of protein databases to generate a result file of identified spectra data. Since the mass spectrometry devices are constantly upgraded, the size of the measured data has increased from Megabytes to Gigabytes. This leads to an increased number of comparisons [19]. However, the current performance of protein identification software is still sufficient for the proteomics research field, as the protein databases belong to one taxonomy from the given experiment sample.

Metaproteomics, a further important research field of biology, extends the proteomics workflow to a biological community instead of only one species. The main task of metaproteomics is to analyze microbial communities, such as biogas plants or human gut and extract bio indicators for such communities [20]. This research brings promising future tasks like optimizing biogas plants, improving the production of compressed natural gas (CNG) and diagnostic use cases in a clinical environment, too [15, 24, 30].

Similar to proteomics, the metaproteomics workflow relies on mass spectrometry and uses the same search engines for protein identification, but on bigger input data [10, 11]. In metaproteomics, the protein databases are exponentially bigger because they need to cover all the possible species in the biological sample. For example, the database of identified proteins of a human in UniProt TrEMBL¹ contains around 140 thousand proteins, whereas the number of entries for all organisms exceeds 100 millions, more than 700 times as much (around 40 GB) [2]. Additionally protein databases with known proteins grow because findings by biologists expand. Hence, the performance of locally executed protein identification in a metaproteomics workflow reaches its limits. Furthermore, the possible future use cases of metaproteomics in the clinical environment for patient diagnostics need near real time processing.

The current X!Tandem needs several hours for one metaproteomics search and uses main memory to store all the experimental spectra data at once. Since the spectra can reach up to 20 GB per experiment with modern mass spectrometer devices, metaproteomics searches, where each experimental spectra is scored against possible thousands of similar matches (thus increasing significantly the memory footprint), are not feasible on a local system. Additionally, the resulting identifications are stored during the runtime in the memory, which increases the RAM use a lot.

¹ The mission of UniProt is to provide the scientific community with a comprehensive, high-quality and freely accessible resource of protein sequence and functional information [2].

Brian Pratt et al. tackle the performance problem of X!Tandem and implement parallel X!Tandem using Hadoop and show, that big data technologies improve performance [25]. But the parallel approach needs a preparation time for partitioning the data and uploading it to the system.

Fast data architectures constitute an alternative approach to process incoming data, which promises real time analysis of sensor data [13,17,28]. In this paper we take a closer look at the X!Tandem protein identification tool and analyze the feasibility of an X!Tandem algorithm on a Fast Data Architecture.

We found, that X!Tandem already exhibits streaming behavior for the proteins, it streams the theoretical spectra. For a central fast data architecture the data processing step has to change to streaming experimental spectra data. Since the spectra data is user dependent and the protein database is static data, that is used by several users, we recommend an X!Tandem fast data architecture with streaming experimental spectra data and a persistent protein database. The new suitable data processing pipeline does not change the logic of the software, only the way of pairwise processing. Besides performance, this method brings some positive effects such as removing redundant peptides and enabling further analyses of measured spectra to the pipeline.

This paper is structured as follows, the Sect. 2 shows the functionality of the protein search algorithm X!Tandem. In Sect. 3 Fast Data is explained. In Sect. 4 the performance of the components of X!Tandem is presented. In Sect. 5 we explain the new Fast Data architecture of X!Tandem components and in Sect. 6 we reveal the conclusion of the new architecture. Section 7 contains related work and future work can be found in Sect. 8.

2 State of the Art Protein Identification Using X!Tandem

Before we start with the explanation of the protein identification algorithm, we need a brief overview on its input data. The input of the protein identification is on the one side the experimental spectral data from a biological sample, and on the other side the protein database that contains information of already known proteins.

2.1 Protein Identification Input: Experimental Spectra

A spectrum is a mass of fragmented ions represented as an intensity to mass-to-charge-ratio plot. With the help of those measurements, the mass of the ions can be calculated.

Figure 1 shows mass spectra from a sample [15]. For each peak of the MS spectrum, an MS2 spectrum can be measured based on fragmentations as an MS/MS spectrum. This data is the outcome of the mass spectrometry, the so called experimental spectra [9]. Typically a Mascot Generic File (MGF) format is used for the textual representation of MS/MS spectra (see Listing 1.1) [9,21,27]. The attributes needed for protein identification are the charge (line 4), the pepmass (line 3) and the list of the peaks (line 5–10).

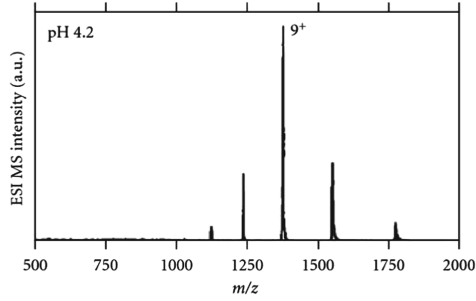


Fig. 1. An example of a mass spectrum [4]

```

1 BEGIN IONS
2 TITLE=Spectrum000107 RHPYFYAPELLYYANK +2 y- and b-series
3 PEPMASS=1023.01766 2000000
4 CHARGE=2+
5 147.11285 100000 1+
6 261.15578 100000 1+
7 ... ..
8 END IONS

```

Listing 1.1. MGF example

2.2 Protein Identification Input: Protein Database

Protein identification is based on the comparison of the experimental spectral data with already known proteins. One amino acid is represented by a specific character. A peptide is a sequence of amino acids and a protein is a sequence of peptides. Protein digestion is needed for separating the proteins to peptides, which splits the protein at specific points into peptides. Since, the mass of the peptide values (amino acids) is known, a theoretical spectrum can be calculated for each peptide. This theoretical spectrum has only perfect values without any measurement artifacts or noise. With the transformation of the proteins to spectra, a comparison between experimental and theoretical spectrum can take place. A protein contains the protein sequence and meta information. The list of proteins in the file serves as a protein database. Typically, a FASTA format is used for the textual representation of proteins (see Listing 1.2) [6].

```

1 >mg|Testsequence_1 Methyl-coenzyme
2 MPMYEDRIDLYGADGKLEEDVPLEAVSPLKNPTIANLVSDVKRSV
3 GAFERMHLLGLAYQGLNANNLLFDLVKENSkgTVGTVIASLVERAI
4 IGSVYSEIDYFREPIVNVARGAAEIKDQL

```

Listing 1.2. Textual representation of one protein in a FASTA format

2.3 Protein Identification Algorithm: A Pairwise Comparison

The process of protein identification is a pairwise comparison. For each experimental spectrum, a protein from the database in the FASTA file is digested and the digested part gets transformed into a theoretical spectrum. The theoretical spectrum and the experimental spectrum from the bio-sample are compared using a similarity function. If the similarity is higher than a specific threshold, the best similarity score is stored as a match. In this way, one experimental spectrum is identified. Later, the proteins with the most matched peptides are identified [22].

2.4 X!Tandem Components

X!Tandem [8] is a well known protein database search algorithm. Other algorithms like Sequest [12] and Mascot [7] are also used for the same purpose, with some differences in their internal working. In our work we focus on X!Tandem, since it is an open source library, that is recommended by biologists [3].

X!Tandem uses text files for all input data. The algorithm loads the experimental spectra complete in the main memory, after it has loaded the protein sequences batchwise. After all the data is loaded, X!Tandem calculates the score for each pair. The similarity is calculated using the dot product of an experimental spectra and theoretical spectra [26]. Additionally X!Tandem provides some features to reduce the amount of comparisons by using convolution filters to reduce the amount of peaks and some parametrized values added, like modification of proteins². Everything is calculated in the scorer, before the similarity function. Since a theoretical spectra is calculated from a peptide, the tool calculates statistical confidence for the peptide-spectrum-match to identify the proteins using an expectation value.

We observe, that the protein database is static: it does not change and it takes time to load it completely before usage. The experimental spectra data is loaded completely into the memory and gets processed one by one, overwriting the best match during the pairwise comparison. After all comparisons the expectation value needs to be calculated to identify the proteins.

To guarantee the usage of protein identification for future metaproteomics workflow an upgrade of the engine is needed. Therefore, the adoption of cloud computing techniques, emphasizing scalability and provisioning resources according to service level expectations, seems promising.

3 Fast Data: The Real Time Big Data

The big data systems evolve to stream oriented systems, where data is processed in mini-batches or streams as it arrives into the system. These so called fast data applications process the data continuously, considering possibly infinite data streams [29]. Fast data focuses on real-time production of data using

² Mutation in the protein sequence change an amino acid.

streaming messaging [1, 17]. The idea of incoming data changes some analytical behavior. The streaming data should be independent, since only a specific time window of the data is available. A popular fast data implementation is the so called SMACK³ stack. The combination of Kafka as a streaming backbone of the system, Cassandra as a distributed storage and Apache Spark as a mini batch processor is a popular combination of streaming applications [29]. In Fig. 2 we show a fast data architecture with possible components. Incoming data can enter directly as Kafka messages, or via HTTP directly (2), using special sockets (1) or be managed by microservices (3). The incoming data is managed in a distributed cluster (4). The messages can communicate directly with the persistence layer (5), mostly to store them without processing or further analysis. To process the data, stream processing engines are used (6 and 8). To store the results or get data for calculation tasks, the processing engines access the persistence component (7, 9 and 10). The whole system needs a cloud operating system schedule the diverse components (11).

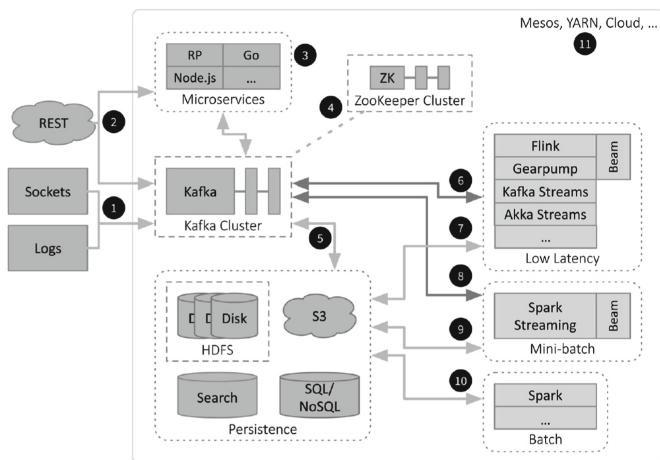


Fig. 2. An example Fast Data streaming architecture [29].

Having in mind that the fast data architecture is a well suited stream processing engine, we want to answer the question: Is protein identification a suitable application for the streaming fast data architecture? First of all we need a detailed view on the current state of X!Tandem.

4 Improvable Components of X!Tandem

As described before, X!Tandem consists of a component to load and digest the proteins, a spectra component to load and loop through the spectra data and

³ Spark, Mesos, Akka, Cassandra and Kafka as streaming pipeline for real time data processing [13].

the scorer, which calculates the score and the expectation value of the proteins. The first step is to load all spectra. We ran X!Tandem 100 times with a protein database (6,158,917 entries) and spectra data (33,227 entries) on our system⁴ to evaluate the current state. The spectrum loader component needs on average 219.6 milliseconds to load one spectrum into the memory. The protein are loaded batch wise (default 1,000 proteins per batch) and each spectrum goes over the batch and calculates the score. The protein and the spectrum loader map the textual representation to a programmatic object and prefilter them to reduce the amount of comparisons. The protein loader needs on average 0.0026 ms to load one protein and each scoring takes on average 644.3 ms. The whole search took on average 5.6 h. The current generation of mass spectrometers can produce more than 300 thousands spectra for one experiment and as mentioned before, a big protein database has more than 100 million entries. Searches on this data would take over 24 h. Additionally, the results, which are stored in an XML format, are not usable for analytical queries or further analysis.

Overall the current X!Tandem approach loads the spectra data into memory and “stream” batch wise the proteins and calculates the scores pairwise, using mapping and filter functions at the runtime. X!Tandem is processing the data like it is already on a fast data architecture, so what benefits does a real integration of the protein search engine to the fast data architecture bring?

5 Streamification of X!Tandem

The integration of X!Tandem would bring benefits for usability, performance, efficient storage and for further processing of the data. The fast data architecture brings a central cloud solution with a central database management system (DBMS). We observed, that the protein data is static and should be stored centrally and accessible for all users. This extends collaboration between biologists and allows reuse and non redundant storage of protein data. The ingesting streaming data will be the experimental spectra instead of the proteins. Streaming these data gives a possibility of further calculation with the experimental spectrum during the protein search such as DeNovo or clustering [14, 16, 18]. A better option would be to digest the proteins, and store the distinct peptide information only once. Making digestion results persistent in this way would create a non-redundant peptide database and reduce the amount of comparisons. The measured spectra are independent from each other and there is no need to load all of them into the memory. Processing of the data needs a stream processing engine for prefiltering, mapping and scoring functions. In metaproteomics, the identification runs against a huge database. Hence, once loaded peptides can be used for multiple queries of different users without loading the peptide every time.

The integration of X!Tandem into a fast data architecture would bring an application that streams spectra batch wise and loops them through all proteins in the database (Fig. 3). The spectra can be streamed from a user using a file

⁴ CPU 48x Intel Xeon E5-2650 v4; 512 GB RAM.

upload stream (2) or a directly connected mass spectrometer (1). The spectra are collected in a message queue (3). The stream processing engine consumes the messages, maps the spectra into programmatic objects and prefilters them (4). In the next step the consumed batch loops through all the peptide sequences from the database (5) and the calculated score and the expectation value get stored in the database (5). All the components are managed by a cloud operating system (6).

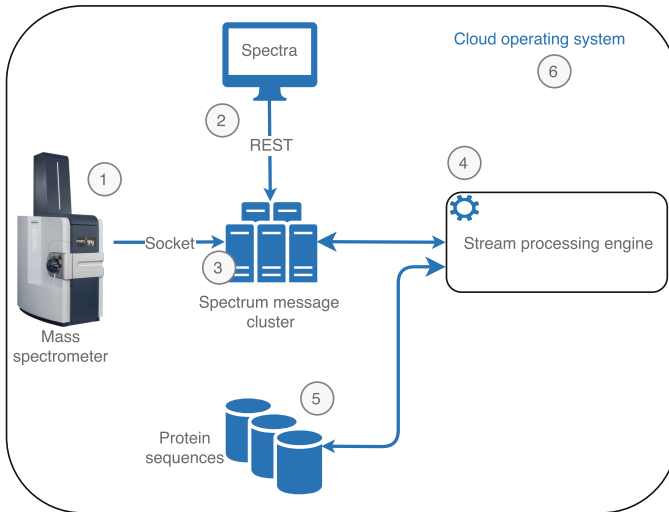


Fig. 3. The general fast data architecture for X!Tandem algorithm.

The next step of the research will be an implementation on the popular SMACK stack. The evaluation and a comparison will show if our expectations for further performance gains are fulfilled.

6 Conclusion

We show in our work pros and cons of the current file-based version of X!Tandem. Also we show, that the current workflow of the algorithm already has a streaming behavior and we make the case that performance benefits could be achieved by using modern fast data technologies. We point out that an integration on a fast data architecture increases not only performance but also usability and enables further analyses besides protein identification. The state of the art of fast data shows clearly the benefits of streaming near real time application, and also our proof of concept is promising. The next research step is a full implementation using the combination of Apache Spark, Apache Kafka and Cassandra and an evaluation against the current X!Tandem.

7 Related Work

Brian Pratt et al. implemented parallel X!Tandem using Hadoop MapReduce on Amazon Web Services, which is the first parallel implementation of X!Tandem to exploit the scalability and fault tolerance of Hadoop to create large on-demand compute clusters on commodity hardware [25]. Another work implemented the algorithm using GPUs to increase the performance and parallel the comparisons [5]. In our work, we propose to implement the X!Tandem protein search on a mini services Fast Data streaming architecture.

8 Future Work

In future work we will implement the X!Tandem protein identification engine using a SMACK stack and also other techniques to improve performance. Besides using different tools, an effective approach for caching of protein sequences will be evaluated. Additionally, we plan to integrate further steps into the pipeline, like clustering the spectra data and preparing the data for visualizations such as chord and krona [23, 31]. Furthermore, we focus on a direct connection of a mass spectrometer with the streaming system.

Acknowledgment. The authors sincerely thank Xiao Chen, Sebastian Krieter, Andreas Meister and Marcus Pinnecke for their support and advice. This work is partly funded by the de.NBI Network (031L0103), the DFG (grant no.: SA 465/50-1), the European Regional Development Fund (grant no. 11.000sz00.00.017 114347 0), the German Federal Ministry of Food and Agriculture (grants nos. 22404015) and dedicated to the memory of Mikhail Zoun.

References



1. Ahmad, Y., Çetintemel, U.: Streaming applications. In: Liu, L., Tamer Özsu, M. (eds.) *Encyclopedia of Database Systems*, pp. 2847–2848. Springer, Heidelberg (2009). https://doi.org/10.1007/978-0-387-39940-9_374
2. Apweiler, R., et al.: UniProt: the universal protein knowledgebase. *Nucleic Acids Res.* **32**, 115–119 (2004)
3. Balgley, B.M., Laudeman, T., Yang, L., Song, T., Lee, C.S.: Comparative evaluation of tandem MS search algorithms using a target-decoy search strategy. *Mol. Cell. Proteomics* **6**(9), 1599–1608 (2007)
4. Banerjee, S., Mazumdar, S.: Electrospray ionization mass spectrometry: a technique to access the information beyond the molecular weight of the analyte. *Int. J. Anal. Chem.* **2012** (2012). <https://doi.org/10.1155/2012/282574>
5. Baumgardner, L., Shanmugam, A., Lam, H., Eng, J., Martin, D.: Fast parallel tandem mass spectral library searching using GPU hardware acceleration. *J. Proteome Res.* (2011). <https://doi.org/10.1021/pr200074h>
6. National Center for Biotechnology Information: Fasta format, November 2002. https://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE_TYPE=BlastDocs&DOC_TYPE=BlastHelp

7. Cottrell, J.S., London, U.: Probability-based protein identification by searching sequence databases using mass spectrometry data. *Electrophoresis* **20**(18), 3551–3567 (1999)
8. Craig, R., Beavis, R.C.: A method for reducing the time required to match protein sequences with tandem mass spectra. *Rapid Commun. Mass Spectrom.* **17**(20), 2310–2316 (2003). <https://doi.org/10.1002/rcm.1198>
9. Deutsch, E.W.: File formats commonly used in mass spectrometry proteomics. *Mol. Cell. Proteomics* **11**(12), 1612–1621 (2012)
10. Duncan, M.W., Aebersold, R., Caprioli, R.M.: The pros and cons of peptide-centric proteomics. *Nat. Biotechnol.* (2010). <https://doi.org/10.1038/nbt0710-659>
11. Elias, J., Gygi, S.: Target-decoy search strategy for mass spectrometry-based proteomics. *Methods Mol. Biol.* **604**, 55–71 (2010). https://doi.org/10.1007/978-1-60761-444-9_5
12. Eng, J.K., McCormack, A.L., Yates, J.R.: An approach to correlate tandem mass spectral data of peptides with amino acid sequences in a protein database. *J. Am. Soc. Mass Spectrom.* **5**(11), 976–989 (1994)
13. Estrada, R.: *Fast Data Processing Systems with SMACK Stack*. Packt Publishing, Birmingham (2016)
14. Griss, J., et al.: Recognizing millions of consistently unidentified spectra across hundreds of shotgun proteomics datasets. *Nat. Methods* (2016). <https://doi.org/10.1038/nmeth.3902>
15. Heyer, R., Kohrs, F., Reichl, U., Benndorf, D.: Metaproteomics of complex microbial communities in biogas plants. *Microb. Technol.* **8** (2015). <https://doi.org/10.1111/1751-7915.12276>
16. Seidler, J., Zinn, N., Boehm, M.E., Lehmann, W.D.: De novo sequencing of peptides by MS/MS. *Proteomics* (2009). <https://doi.org/10.1002/pmic.200900459>
17. Kipf, A., Pandey, V., Boettcher, J., Braun, L., Neumann, T., Kemper, A.: Analytics on fast data: main-memory database systems versus modern streaming systems. In: 20th International Conference on Extending Database Technology (2017)
18. Kokaly, R., et al.: USGS spectral library version 7. Technical report, U.S. Geological Survey Data Series 1035 (2017). <https://doi.org/10.3133/ds1035>
19. Lubeck, M., et al.: PasefTM on a timstof pro defines new performance standards for shotgun proteomics with dramatic improvements in MS/MS data acquisition rates and sensitivity. Technical report, Bruker Daltonik GmbH (2017)
20. Maron, P.A., Ranjard, L., Mougel, C., Lemanceau, P.: Metaproteomics: a new approach for studying functional microbial ecology. *Microb. Ecol.* **53**, 486–493 (2007)
21. McDonald, W.H., et al.: MS1, MS2, and SQT-three unified, compact, and easily parsed file formats for the storage of shotgun proteomic spectra and identifications. *Rapid Commun. Mass Spectrom.* **18**(18), 2162–2168 (2004). <https://doi.org/10.1002/rcm.1603>
22. Million, R., Franchin, C., Tessari, P., Polati, R., Cecconi, D., Arrigoni, G.: Pros and cons of peptide isoelectric focusing in shotgun proteomics. *J. Chromatogr. A* **1293**, 1–9 (2013). <https://doi.org/10.1016/j.chroma.2013.03.073>
23. Ondov, B.D., Bergman, N.H., Phillippy, A.M.: Interactive metagenomic visualization in a web browser. *BMC Bioinform.* **12**(1), 385 (2011). <https://doi.org/10.1186/1471-2105-12-385>
24. Petriz, B.A., Franco, O.L.: Metaproteomics as a complementary approach to gut microbiota in health and disease. *Front. Chem.* (2017). <https://doi.org/10.3389/fchem.2017.00004>

25. Pratt, B., Howbert, J.J., Tasman, N.I., Nilsson, E.J.: MR-Tandem: parallel X!Tandem using hadoop mapreduce on Amazon web services. *Bioinformatics* (2012). <https://doi.org/10.1093/bioinformatics/btr615>
26. Craig, R., Beavis, R.C.: A method for reducing the time required to match protein sequences with tandem mass spectra. *Rapid Commun. Mass Spectrom.* 17, 2310–2316 (2003)
27. Matrix Science: Data file format (2016). http://www.matrixscience.com/help/data_file_help.html
28. Wampler, D.: Fast data: big data evolved. White Paper (2015)
29. Wampler, D.: *Fast Data Architectures for Streaming Applications*, 1st edn. O'Reilly Media, Sebastopol (2016)
30. Zhang, J., Liang, Y., Yau, P., Pandey, R., Harpalani, S.: A metaproteomic approach for identifying proteins in anaerobic bioreactors converting coal to methane. *Int. J. Coal Geol.* 146, 91–103 (2015)
31. Zoun, R., Schallert, K., Broneske, D., Heyer, R., Benndorf, D., Saake, G.: Interactive chord visualization for metaproteomics. In: 28th International Workshop on Database and Expert Systems Applications (DEXA), pp. 79–83, August 2017. <https://doi.org/10.1109/DEXA.2017.32>



Mining Geometrical Motifs Co-occurrences in the CMS Dataset

Mirto Musci^(✉)  and Marco Ferretti 

Department of Computer Engineering,
University of Pavia, Via Ferrata 5, 27100 Pavia, PV, Italy
mirto.musci@unipv.it

Abstract. Precise and efficient retrieval of structural motifs is a task of great interest in proteomics. Geometrical approaches to motif identification allow the retrieval of unknown motifs in unfamiliar proteins that may be missed by widespread topological algorithms. In particular, the Cross Motif Search (CMS) algorithm analyzes pairs of proteins and retrieves every group of secondary structure elements that is similar between the two proteins. These similarities are candidate to be structural motifs. When extended to large datasets, the exhaustive approach of CMS generates a huge volume of data. Mining the output of CMS means identifying the most significant candidate motifs proposed by the algorithm, in order to determine their biological significance. In the literature, effective data mining on a CMS dataset is an unsolved problem.

In this paper, we propose a heuristic approach based on what we call protein “co-occurrences” to guide data mining on the CMS dataset. Preliminary results show that the proposed implementation is computationally efficient and is able to select only a small subset of significant motifs.

Keywords: Proteins · Secondary structure · Geometrical motifs
Cross Motif Search · Data mining

1 Introduction

At the primary level, a protein is made by one or more amino-acid chains. Groups of amino-acids with recurrent spatial configurations are called Secondary Structure Elements (SSE). SSEs are the basic building blocks of the secondary level: the most common ones are alpha helices and beta strands. *Structural motifs* are recurrent spatial patterns made of several SSEs. A thorough, automated analysis of the secondary structure is instrumental to improve the understanding of protein functionalities. In particular, the identification of structural motifs is pivotal for evolutionary studies, pharmaceuticals, and more. Such analysis has the potential to generate high volumes of data when applied to the ever-growing Protein Databank (PDB).

The focus of this paper is to address the challenges of mining the biological data generated by *Cross Motif Search* (CMS) [1–9], a computer-vision based algorithm for motif identification, which relies on a simple geometrical model for the protein.

1.1 State of the Art

There are several established tools for motif identification: DALI [10], SSM [11], MASS [12], PROMOTIF [13], and many others. Most of them depend on a *topological* description of the protein structure, where the interconnection between each element in the structure is stored in a graph, in order to leverage on graph algorithms for the identification.

Cross Motif Search (CMS) is an algorithm for the retrieval and identification of secondary structure motifs using a *geometrical approach* [1–9]. CMS relies on the definition of a geometrical model for the secondary structure. In literature, different approaches based on hybrid topological and geometrical approaches have been described, such a ProSmoS [14]. The authors of [15] apply an approach similar to CMS to the identification of patterns in the primary structure.

A geometrical approach takes into consideration all spatial information, and makes no assumption on the topology of any candidate motifs. The result is that geometrical algorithms are computationally more intensive than topological ones. Moreover, the inherent assumptions of topological models make them extremely efficient tools for the identification of motifs in proteins that share many similarities (i.e. have been classified in the same evolutionary family). However, they struggle on proteins belonging to *different families*. Indeed, CMS is of particular interest, because it *allows the identification of unknown geometrical motifs belonging to proteins of different families*. The justification of this claim is discussed in much detail in [1, 6, 7].

CMS is freely available as a command-line tool, or through *MotifVisualizer* (MV), a Graphical User Interface (GUI), developed to facilitate cooperation with communities other than computer scientists, such as biologists [9].

1.2 CMS Algorithm

CMS operates (“runs”) on two proteins at a time (a *pair*): the first one is called *source protein*; the other is called *search protein*. Both source and search proteins are modelled as geometrical objects: *each SSE in each chain* is represented as a simple *line segment* in the 3D space (Fig. 1).

CMS uses *triplets* of SSEs as its basic geometrical primitive, and then applies to them a variant of the well-known Generalized Hough Transform (GHT). The algorithm works *exhaustively*, in a *recursive* way: *each possible triplet in the source protein is looked for in the search protein*. Then, all positive triplet matches are extended to create *quadruplets* of SSEs. Every valid quadruplet is then looked for in the search protein thanks to the GHT, and so on, recursively. Thus, a valid candidate motif is modelled as a cloud of line segment in the protein space, with the caveat that every line segment must lie in the same amino-acid chain.

Note that each CMS run can be personalized using different *tolerance parameters*: put in simple words, they represent how close two structures are required to be, in order to be identified as similar by the GHT.

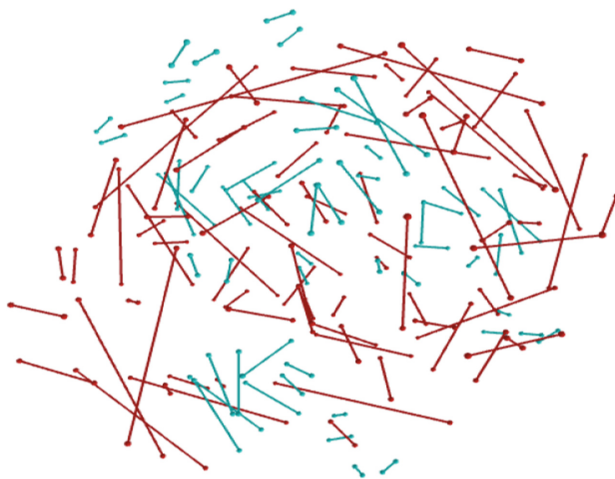


Fig. 1. The secondary structure of a protein, shown by MotifVisualizer [9]. Secondary structure elements are represented as simple line segments for both helices (red) and strands (blue). (Color figure online)

The algorithm can work on multiple pairs of proteins at the same time: a set of proteins $\{P_i\}$ is processed by simply applying CMS to each pair (P_i, P_j) . For this reason, and for its exhaustiveness, CMS runs can be very intensive on the CPU; hence the algorithm has been parallelized using OpenMP on each protein pair comparison, and then with a hybrid MP/MPI approach on multiple pairs [4].

1.3 CMS Data Flow

The CMS algorithm needs *two XML files as its input*. These files describe the source and search proteins as a list of structural elements. To provide such input, it is necessary to parse and convert a DSSP file (Dictionary of Secondary Structure of Protein [16]), which in turn is processed from the well-known Protein Data Bank (PDB) format. Conveniently, several databases of DSSP files are available [17].

As discussed above, a typical CSM run is performed on *unfamiliar* proteins. In order to do so, the SCOP database (Structural Classification Of Protein [18, 19]) is used as a guidance in retrieving the data input. The entire process for the generation of the input file has been streamlined and automated thanks to the *SCOPEXplorer* tool [5].

Similarly, the output of a CMS run on a pair of proteins produces a report, with the same XML format (Fig. 2). Each report lists the geometrical similarities identified between the pair of proteins subjected to the analysis and all tolerance parameters of the GHT; a report is produced only if at least one common structure has been identified.

A *CMS dataset* is defined as the set of all reports produced by CMS on a given set of proteins with a given set of parameters. If the set contains N proteins, a CMS dataset contains at best $N(N - 1)$ reports, one for each possible pair.

The parallel implementation of CMS allows running the algorithm on large sets of proteins, up to the entire PDB if required. However, this leads to significant challenge: the statistical analysis, or data mining of the output. The crucial step is the post processing of output reports: to determine if a geometrical similarity is indeed a biologically significant motif, some kind of scoring or selection should be performed on the CMS dataset. Such approach would need to efficiently identify statistically significant patterns in the data set, and discard singular and trivial occurrences.

In theory, a manual examination by domain experts would be the best possible approach; but if N is large, or the analysis needs to be repeated on many different datasets, the manual approach is unfeasible. However, if some data mining process would be able to select a small set of candidates with some interesting properties, manual evaluation becomes feasible on such restricted set.

1.4 Structure of the Paper

In order to address the data mining challenge on the CMS dataset, we present a heuristic approach based on feature selection and on the identification of co-occurrences between different candidate motifs (Sect. 2). In Sect. 3 we describe its implementation in Python and SQL. Section 4 shows some preliminary results of applying the proposal on the CMS dataset. Section 5 concludes the paper.

2 Methodology

As discussed in the introduction (Sect. 1.2), CMS is able to identify all geometrical similarities between two proteins. Each similarity is described by two sets of SSEs: the first set belongs to the source protein; the second set belongs to the search protein. Remember that a group of SSEs is represented as a set of linear segments.

Each similarity can be seen as a *candidate motif*; the main problem of data mining, is to *identify which candidates are the most significant*. Clearly, the “significance” of a similarity can be evaluated only within the context of a set of proteins and not in a vacuum.

In order to determine the significance of a similarity, we define a *heuristic score*, based on a weighted sum of a *few selected features*. Given a dataset, each feature concurs to give a value to each similarity identified by a CMS on the entire dataset. The intuition is that a candidate motif with a high score has a good probability of being “significant” and thus of high biological interest.

2.1 Feature Selection

We have identified four features based on which candidate motifs can be ranked. In the implementation, different weights can be assigned to each feature. The selected features are:

- *Number of occurrences*: that is the number of times that the same structure has been retrieved by CMS in a given dataset. When compared with the total number of pairs in the dataset, is a good indication of the significance of the candidate.

- *SCOP category of compared proteins*: that is the deepest common SCOP category containing both the source and the search protein of the candidate. A common structure among similarly classified proteins is expected, and it is likely to be less significant than a similarity between two proteins in different SCOP classes, which could point-out to previously unknown protein functions.
- *Cardinality*: that is the number of SSEs in the candidate motif. Clearly, large motifs are more interesting from a biological point-of view. Moreover, the GHT algorithm is very susceptible to the accumulated errors that even small spatial turbulences could produce, so that large candidates are rare and points to geometrical structure that are very similar to each other.
- *Number of co-occurrences*: is the most complex feature, and we believe it is crucial for stating the significance of a candidate; it is described in the following section.

2.2 Co-occurrences

Thanks to the properties of the GHT, we know that two similar sets of SSEs satisfy a *weak equivalence relation*, with the following properties:

Reflexive property. A group of SSEs is similar to itself. That is, a CMS run on a protein and itself will return exactly a similarity if the protein contains 3 SSEs and $O(N^2)$ similarities if contains N SSEs.

Symmetric property. A CMS run on sets S_1 and S_2 will produce the same report of a CMS run on sets S_2 and S_1 . Put in other words, source and search proteins are interchangeable.

However, the transitive property is not satisfied in general. The reason is that CMS use a variable degree of tolerance in order to match geometrical structures with small differences in their conformations, due to the natural variability of the same structure conformation across different proteins.

Example 1. Let us consider proteins A , B and C , and let us define A_i as any set of n SSEs belonging to A , ordered in some arbitrary but consistent fashion. Suppose that CMS is run on the protein pairs (A, B) , (A, C) and (B, C) . The output reports show that both (A, B) and (A, C) have at least a match. That is, both (A, B) and (A, C) have a common similarity. Moreover, let us say that A_i shows up in both reports, so that (A_i, B_j) are similar in (A, B) and (A_i, C_k) are similar in (A, C) . However, nothing in the algorithm guarantees that B_j and C_k are similar in general; it is even possible that a CMS run on (B, C) does not identify any similarity at all!

Some structures, however, are “special” in the sense that they satisfy a partial transitive property, which only holds for a limited subset of proteins.

Partial Transitive Property. Let us consider a structure $P_i^{(j)}$ belonging to the protein $P^{(j)}$, and let us suppose that a CMS run on the sets of proteins $\{P^{(1)}, \dots, P^{(j)}, P^{(k)}, \dots, P^{(n)}\}$ identifies similarities between $P_i^{(j)}$ and $\{P_i^{(1)}, \dots, P_i^{(n)}\}$.

We say that $P_i^{(j)}$ satisfies a partial transitive property with a degree $k > 1$ if $P_i^{(l)}$ is similar to $\{P_i^{(1)}, \dots, P_i^{(k)}\} - \{P_i^{(l)}\}$ for any l in $\{1, k\}$.

Put in other words, a structure is partially transitive if it participates to a subset of k similarities that satisfy the transitive property.

In analogy to the number of occurrences feature described above, we say that a similarity has a number of co-occurrences in a set of proteins equal to the maximum value of k for which the partial transitive property defined above holds. Clearly, similarities with a high number of co-occurrences are the most interesting from a motif identification point-of-view, while similarities with no co-occurrences should be discarded, even if they have a high number of occurrences.

Example 2. Given three proteins A , B and C of the previous example, A_i has a co-occurrence with a score of 3 if the following pairs of structures are similar: (A_i, B_j) , (A_i, C_k) , (B_j, C_k) . Note that, for simplicity, we are ignoring the scenario where A_i is similar to multiple structures in either B or C .

3 Implementation

Three new components have been added to the CMS family: a XML parser (*CMS Parser*) to insert the output data from the CMS dataset into a MySQL database, a SQL application (*Co-Occurrences Calculator*) that queries the database in order to compute the number of co-occurrences, and a visualization tool integrated with MotifVisualizer.

The GUI enables the user to browse the MySQL database, check the co-occurrences scores and evaluate different combinations of the weights for the heuristic function described in Sect. 2.1. The three components have been implemented with a combination of Python and SQL.

3.1 CMS Parser

The first component is command-line tool, called CMS Parser (CMSP). For each XML report in the CMS dataset, CMSP parses it in order to insert its data into a SQL database. XMLP implements a FTP client, in order to download the CMS reports stored on a remote FTP server. In a previous work [5] we tried a direct port of the XML report in a non-relational database (*BerkleyDB*). The Motif Match Analyzer tools described in [5] performed quite well on a small testing dataset, but it did not scale well on an entire CMS dataset.

As mentioned before, each protein in the dataset comes from a different super-family protein. XMLP has been implemented in Python with the *ElementTree* module. Then, the data is inserted into an SQL table with the following columns:

[Source protein, Search protein, Source chain,
Source Motif, Search chain, Search Motif]

Following the notation introduced in Sect. 2.2, each row represents a similarity between two structures; that is a pair of structures (A_i, B_j) belonging to some chains of the source protein A and the search protein B . This data can be parsed directly from a

Then, in order to list every possible co-occurrence, we create a *second view*, using a *SQL inner-join* between the first view and the Reports table. The goal is to check whether the structures B_j and C_k of a generic row in the first view are present in the Reports table and thus are valid candidates. If the query condition is satisfied, than a *row of the second view represents a single co-occurrence of A_i* .

Finally, in order to obtain the number of co-occurrences of A_i , we count the number of rows in the second view that share the same structure A_i . The result of this last query is inserted in a table, called Ratings. Ratings has the following simple structure:

[Protein, Chain, Motif, Co-Occurrences]

To ensure reasonable performances, we have built *SQL indexes* on both conditions for self-join query; this allow reducing the execution time of the query by two orders of magnitude, at the cost of a larger memory footprint.

3.3 Visualization Tool

In order to browse the results in the Ratings table, apply the feature selection described in Sect. 2.1, and visualize interesting candidates in MotifVisualizer, we implemented a *Python GUI to the data mining*. Figure 3 shows the interface of this visualization tool.

The tool allows the user to select any protein, any chain or any SSE structure, check their co-occurrences ratings, and apply filters based on the features described in Sect. 2.1 (number of occurrences, cardinality, and SCOP depth).

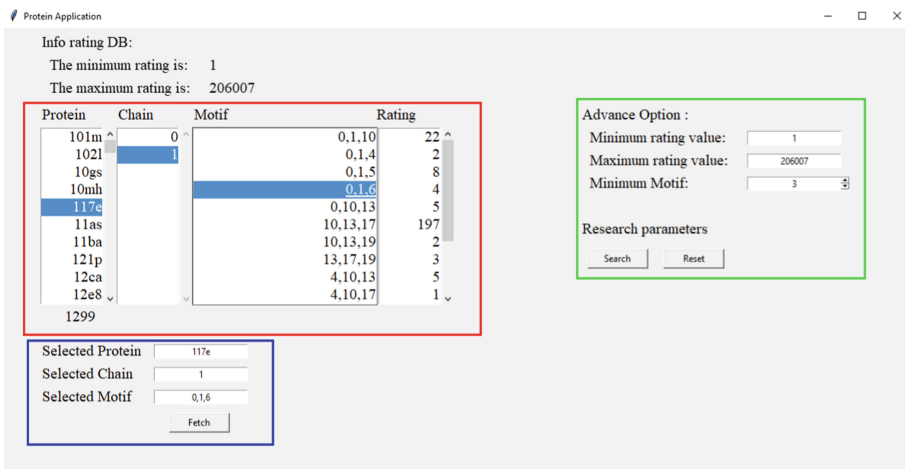


Fig. 3. The interface of the visualization tool. The red box shows a view of the Ratings table (Sect. 3.2). The user can navigate the view and select proteins, chains or structures of interest. The text fields in the green area allow the user to apply search filters to the main viewport. In-depth information on the selection are shown in the blue area. The “fetch” button generates an XML file on the fly.

The most important feature of the visualization tool is the ability to generate XML files of the selected structure on the fly, retrieving every candidate motif in the CMS dataset that has produced a valid co-occurrence. Note that the number of generated files is equal to the co-occurrences rating of the structure. Finally, the tool calls MotifVisualizer [9] in order to visualize significant candidates within a 3D environment.

4 Results

We tested the data-mining technique proposed in Sect. 2, on the protein dataset described in [6]. For the reasons discussed in Sect. 1.1, every protein in the dataset comes from a different SCOP super-family [18]. From each super-family, we select the first protein in lexicographic order. In total, the dataset contains 1,470 proteins.

We ran CMS on the dataset, using the default (strict) tolerances for the geometrical identification. CMS produced 303,402 reports, for a total of 37.7 GB of uncompressed text data. Overall, CMS was able to identify more than 6 million candidate motifs (6,051,079 rows in the Reports table); however, the vast majority of them have no co-occurrences, and thus are of no biological significance. Then, we run both CMSP and COC on the dataset.

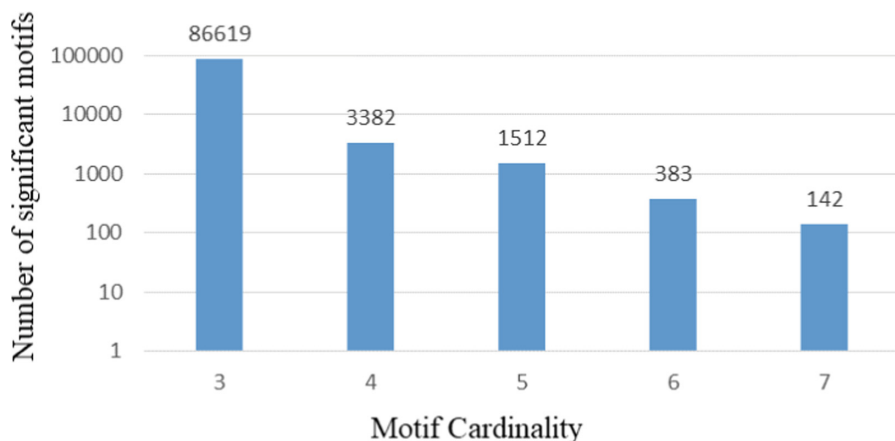


Fig. 4. The number of significant motifs (logarithmic scale) identified by the heuristic data mining of the run on the CMS dataset described in [6]. Motifs are divided for their cardinality (Sect. 2.1).

As expected, CMSP is able to process all the reports in the matter of few minutes. COC runs in a couple of hours on an average workstation. The memory footprint of the SQL database is almost 8 GB. SQL indexes occupy 659 MB.

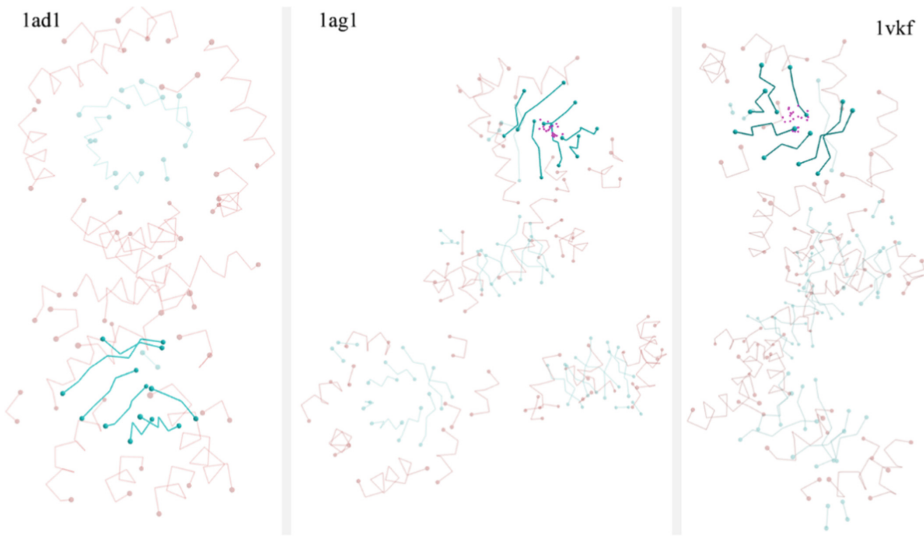


Fig. 5. A significant motif with three occurrences and valid co-occurrences. The motifs are displayed in MotifVisualizer [9] thanks to the XML generated on the fly by the visualization tool described in Sect. 3.3.

Figure 4 shows the number of significant motifs, i.e. motifs that contains at least a structure with a number of co-occurrences greater than 1. Significant motifs are partitioned by their cardinality. The graphs gives an evidence of the large amounts of irrelevant data in the dataset; indeed only 92,000 motifs were deemed significant on a total of 6 million ($\sim 1.6\%$ of the total).

Moreover, Fig. 4 shows that most significant structures have a low (i.e. 3–4) cardinality, and thus can be discarded. At the end, we have mined 2,037 candidates: a manageable number for visual inspection, that can be further reduced with a careful use of the features described in Sect. 2.1.

Figure 5 shows one of the largest significant motifs (cardinality of 7) identified in the dataset. In particular, the figure shows a candidate with 3 occurrences: it was originally found in protein 1AD1, 1AG1 and 1VKF. The triplet, obviously, is a valid co-occurrence according to COC. Note that the PDB entry for 1VKF annotates the apparent asymmetry of the entire conformation. On the other hand, our results clearly shows a pattern, which has also been found in two other proteins. This previously unknown symmetry is a perfect indication of the kind of results we expect from CMS.

5 Conclusions

Geometrical algorithms such as CMS can retrieve biological significant motifs in unfamiliar proteins; unfortunately, mining the huge volume of candidate identified by a CMS run, in order to isolate the significant motifs, is very challenging.

In this paper, we have proposed a method based on a heuristic feature selection to mine the CMS dataset. The most critical feature is the number of co-occurrences, to which a formal definition and a method of computation has been provided. Three software components integrated in CMS have been implemented in Python and SQL in order to data-mine the CMS dataset.

The implementation has been tested on a CMS dataset of 1,470 proteins, belonging to different SCOP super-families. Preliminary experiments shows that it is possible to isolate significant motifs in the CMS dataset in a way that is also computationally efficient. However, future partnerships with biologists will be necessary to understand if the mined candidates are indeed significant in the biological sense.

References

1. Ferretti, M., Musci, M.: Geometrical motifs search in proteins: a parallel approach. *Parallel Comput.* **42**, 60–74 (2015)
2. Cantoni, V., et al.: Protein motif retrieval by secondary structure element geometry and biological features saliency. In: 25th International Workshop on Biological Knowledge Discovery and Data Mining (BIOKDD 2014), pp. 23–26 (2014)
3. Cantoni, V., Ferretti, M., Musci, M., Nugrahaningsih, N.: Structural motifs identification and retrieval: a geometrical approach. In: *Pattern Recognition in Computational Molecular Biology: Techniques and Approaches*, 1st edn. Wiley, Hoboken (2016)
4. Drago, G., Ferretti, M., Musci, M.: CCMS: a greedy approach to motif extraction. In: Petrosino, A., Maddalena, L., Pala, P. (eds.) *ICIAP 2013*. LNCS, vol. 8158, pp. 363–371. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-41190-8_39
5. Argentieri, T., Cantoni, V., Musci, M.: Extending cross motif search with heuristic data mining. In: 28th International Workshop on Biological Knowledge Discovery and Data Mining (BIOKDD 2017) (2017)
6. Ferretti, M., Musci, M.: Entire motifs search of secondary structures in proteins: a parallelization study. In: *Proceedings of the 20th European MPI Users' Group Meeting*, New York, NY, USA, pp. 199–204 (2013)
7. Ferretti, M., Musci, M., Santangelo, L.: MPI-CMS: a hybrid parallel approach to geometrical motif search in proteins. *Concurr. Comput. Pract. Exp.* **27**(18), 5500–5516 (2015)
8. Ferretti, M., Musci, M., Santangelo, L.: A hybrid OpenMP and OpenMPI approach to geometrical motif search in proteins. In: *IEEE International Conference on Cluster Computing (IEEE Cluster 2014)*, pp. 298–304 (2014)
9. Argentieri, T., Cantoni, V., Musci, M.: Motifvisualizer: a interdisciplinary GUI for geometrical motif retrieval in proteins. In: *Biological Knowledge Discovery and Data Mining - 27th International Conference on Database and Expert Systems Applications (2016)*
10. Holm, L., Sander, C.: Protein structure comparison by alignment of distance matrices. *J. Mol. Biol.* **233**, 123–138 (1993)
11. Krissinel, E., Henrik, K.: Secondary-structure matching (SSM), a new tool for fast protein structure alignment in three dimensions. *Acta Crystallogr. D Biol. Crystallogr.* **60**, 2256–2268 (2004)
12. Dror, H., Nussinov, B.R., Wolfson, H.: Mass: multiple structural alignment by secondary structures. *Bioinformatics* **19**(1), i95–i104 (2003)

13. Hutchinson, E.G., Thornton, J.M.: PROMOTIF—a program to identify and analyze structural motifs in proteins. *Protein Sci.* **5**(2), 212–220 (1996)
14. Shi, S., Chitturi, B., Grishin, N.V.: ProSMoS server: a pattern-based search using interaction matrix representation of protein structures. *Nucleic Acids Res.* **37**(Web Server issue), W526–W531 (2009)
15. Kumar, S., Nei, M., Dudley, J., Tamura, K.: MEGA: a biologist-centric software for evolutionary analysis of DNA and protein sequences. *Brief Bioinform.* **9**(4), 299–306 (2008)
16. Kabsch, W., Sander, C.: Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers* **22**(12), 2577–2637 (1983)
17. Touw, W.G., et al.: A series of PDB related databases for everyday needs. *Nucleic Acids Res.* **43**(Database issue), D364–D368. <ftp://cmbi.ru.nl/pub/molbio/data/dssp/>
18. Murzin, A.G., Brenner, S.E., Hubbard, T., Chothia, C.: Scop: a structural classification of proteins database for the investigation of sequences and structures. *J. Mol. Biol.* **247**, 536–540 (1995)
19. Hubbard, T.J.P., Murzin, A.G., Brenner, S.E., Chothia, C.: Scop: a structural classification of proteins database. *Nucleic Acids Res.* **25**, 236–239 (1997)



Suitable Overlapping Set Visualization Techniques and Their Application to Visualize Biclustering Results on Gene Expression Data

Haithem Aouabed¹, Rodrigo Santamaría², and Mourad Elloumi¹(✉)

¹ Laboratory of Technologies of Information and Communication and Electrical Engineering (LaTICE), National High School of Engineers of Tunis (ENSIT), University of Tunis, Tunis, Tunisia

haithem.abdi@gmail.com, mourad.elloumi@gmail.com

² Departamento de Informática y Automática, Universidad de Salamanca, Salamanca, Spain

rodri@usal.es

Abstract. Biclustering algorithms applied in classification of genomic data have two main theoretical differences compared to traditional clustering ones. First, it provides bi-dimensionality, grouping both genes and conditions together, since a group of genes can be co-regulated for a given condition but not for others. Second, it considers group overlaps, allowing genes to contribute to more than one activity. Visualizing biclustering results is a non-trivial process due to these two characteristics. Heatmaps-based techniques are considered as a standard for visualizing clustering results. They consist on reordering rows and/or columns in order to show clusters as contiguous blocks. However, for biclustering results, this same process cannot be applied without duplicating rows and/or columns. Moreover, a variety of techniques for visualizing sets and their relations has been published in the past recent years. Some of them can be considered as an ideal solution to visualize large sets with high number of possible relations between them. In this paper, we firstly review several set-visualizing techniques that we consider most suitable to satisfy the two mentioned features of biclustering and then, we discuss how these new techniques can visualize biclustering results.

Keywords: Biclustering · Visualization · Sets
Set-visualizing techniques · Overlaps

1 Introduction

Several genomic data analysis workflows imply identifying groups of biological entities (e.g. genes) that exhibit similar behavior under certain conditions. Indeed, analysis of DNA gene expression data has the potential to find the characteristics of a particular disease by comparing related tissues and cells, identify

new genes then study their functions and expressions in different conditions and, help in the discovery of more effective drugs. One of the dominant methods to analyze genomic abundance data is clustering [7] which consists in a selection of genes (rows) with similar expression patterns under the whole set of conditions (columns). However, for specific biological aims, the detection of new patterns is feasible only when the analysis carry out the grouping in both dimensions simultaneously: genes and conditions. In fact, this allows finding subgroups of genes that co-expressed under certain experimental conditions and may not have any co-expression with the other conditions. These nuggets of local patterns can be discovered based on biclustering [6, 18], a non-supervised technique expanded at a constant pace in the recent years (see [19] for a survey). Biclustering exceeds traditional clustering techniques due to its two main characteristics; *bi-dimensionality* and *overlaps*.

Visualizing biclustering results is an interesting process to infer patterns from the expression data [20]. The most popular techniques to visualize a single bicluster are heatmaps and parallel coordinates. In a heatmap [7], genes are posted as rows while conditions are posted as columns with a color that represents the gene expression level. This technique is implemented in several tools which include BiVoc [9], BicAT [3], BicOverlapper [26], Furby [28], Bicluster Viewer [10], BiGGES TS [8], Expander [27] and biclust [15]. In order to draw a bicluster, rows and columns are rearranged and usually placed in the upper left corner. In parallel coordinates method [13], conditions are visualized as vertical axes and genes as lines, which join corresponding expression values. This prospect is used by some software tools like BiVisu [5], BicAT, Bicluster Viewer, BicOverlapper, biclust and Expander. To visualize multiple biclusters, most widespread proposed techniques are also based on heatmaps. By reordering and replication of rows and columns of each bicluster, a global view of all biclustering results is possible. Jin et al. [14], BiVoc and Bicluster Viewer are examples of tools that implement this method. Usually, scalability is the principal drawback of these tools. Recently, there are some more sophisticated visualization prospects implemented in already cited tools; BicOverlapper used a force-directed graph as technique to show overlapping biclusters. With an *overview + detail* approach, bicluster details are drawn in a separate view for best analysis. Overlaps can be detected easily but with a high number of biclusters and rate of overlapping, obtaining a readable overview of the biclustering results as a whole will be more/less difficult especially with some geometrical constraints [2, 28]. In Furby, biclusters are considered as nodes and represented by heatmap matrix while overlapping genes or samples between biclusters are depicted by edges. However, drawn edges link a set of genes and columns that are shared between each pair of biclusters, so the rate of crossing between these edges will be high which leads to a cluttered result view [2]. Moreover, a variety of techniques for visualizing sets and relations between them, have been published in the past recent years. With different capacities to support scalability issues and different approaches to represent relations, some of them can be a solution to visualize biclustering results.

Our aim in this paper is to review several set-visualizing techniques that we consider most suitable to satisfy characteristics of biclustering and then, we discuss how these new techniques can visualize biclustering results. For an exhaustive review about set visualization techniques, see [2].

2 Euler and Venn Diagrams

The oldest methods to visualize sets and their intersections are *Euler* and *Venn* diagrams. They were invented by the English logician and philosopher, John Venn in the 18th century and used as a common means of teaching set theory and logical relations in classrooms [4, 16]. Sets are represented by closed curves in the plane, often circles, and set relations are illustrated by curve overlaps. All possible relations between sets including intersection, inclusion and exclusion can be depicted because there are no restrictions about the way of representing overlaps. Venn diagrams which are a particular form of Euler diagrams, they represent all possible intersections between sets either they are empty or not. Based on an area-proportional approach where the drawn areas characterize the size of a set and its intersections, both methods are intuitive and simple to depict set relations. Euler diagrams are extensively used in such areas like genetics and proteomics [22]. For a global overview of Euler diagrams, see [24].

2.1 Untangling Euler Diagrams

Among several variations techniques of Euler diagrams that use closed curves to present sets and relations between them but add more sophisticated improvements, we notice *Untangling* Euler diagrams [23]. This variation is invented to address three basic issues of traditional Euler diagrams that are the complexity of the regions used to draw sets, the suitable approach to draw elements of sets in which sufficient space is provided inside the regions, and the scalability [23]. Untangling Euler diagrams are apt to break complex set intersections with a strict hierarchy that can be easily drawn by readable and rectangular areas with consideration of elements and control of their positions. Intersections between sets are represented by additional bundle links to reduce cluttering. This topologically Euler-like variation was adapted in two variations; Compact Rectangular Euler Diagram (ComED) and Euler Diagram with Duplications (DupED) (see Fig. 1).

ComED consists in splitting sets involved in intersections into multiple rectangular parts. To assure connectivity and continuity of each set, these parts are connected by hyper-edges that their crossings not contain any shared elements between participated sets. The rectangular parts are disposed in a strict containment hierarchy that reveals several set relations (intersection, inclusion, etc.). The second approach (i.e. DupED) gives separate rectangular regions for each set to avoid drawing intersections. Duplication is restrained to elements belonging to multiple sets. Links are drawn between multiple instances of each shared element. While DupED has a high potential to assess the number of sets,

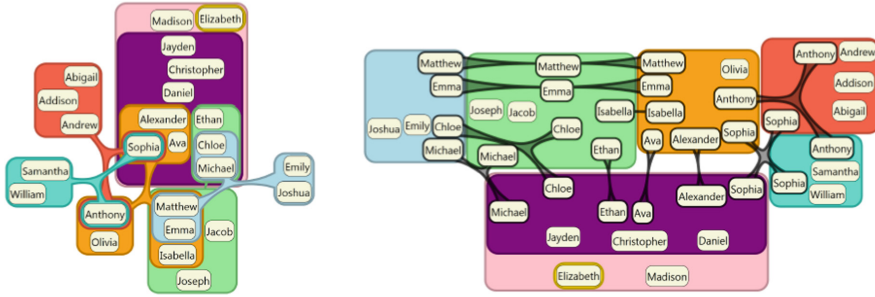


Fig. 1. Untangling Euler diagrams visualization: ComED (left) and DupED (right) [23].

their interactions and their sizes, ComED is usually the preferred useful method because, if the number of sets grow significantly, it scales better in terms of visual complexity [2].

3 Matrix-Based Techniques

A matrix is composed by n rows and m columns. Sets and their specific memberships can be easily visualized as matrix (i.e. elements as rows and sets as columns). Due to its intuitive visualization approach, matrix techniques can increase the readability of the visualized data. Overlaps between sets can be depicted either by adapting reordering and duplication principles [30], using a separate matrix [17] or combine matrix with other visualization techniques like node-link diagrams [11]. Matrix techniques are well adapted in several domains especially in biology and bioinformatics, for example, in genomic data analysis.

3.1 OnSet

OnSet or PixelLayers [25] considers set visualization with a new and different way. The matrix of elements is presented as a *whole* grid where each element is characterized by a specific and unique location. Each element is drawn by a very small square named *pixel*. Then, each set is represented by a separate matrix that contains only elements that belong to it, which take the state of *colored* while other elements are fixed in *blank* state. To know individually shared elements between the sets, it is enough to hover over its corresponding position with the mouse. Comparing all elements between two or more sets is depicted by drag and drop interactions. The result is an aggregated matrix. Two logical operations are supported; AND and OR. The former highlights intersections while the latter displays unions. Figure 2 shows an example where the sets are generated from biochemical data [25].

the corresponding filled circles are also connected with a line. In order to address several essential analysis tasks such as comparison between two sets or between some elements participated in a specific intersection of sets, intersections can be aggregated according to some aggregation semantics or sorted according to some sorting measures. Elements in UpSet can be visualized in different ways such as tables, scatter plots, histograms, etc. depending on the characteristics of data. To visualize more complicated data like gene expression data, the proposed method enables the usage of extended specific APIs. Also, logical queries can be defined either on set view or element view to answer specific customized interests.

4 Aggregation-Based Techniques

Aggregation-based techniques [2] consist on addressing scalability by *aggregating* multiple elements that belong to specific set overlaps into a single visual form, usually a *bar*. Thereby, each set is divided into segments that represent different degrees of its elements. A degree of an element depicts the number of sets it belongs to. These techniques enable exploring details about certain elements on demand instead of showing elements of a set all together. Analyst interactions are needed to infer overlaps between sets.

4.1 Radial Sets

Radial Sets [1] is a Aggregation-based visualization technique that aggregate elements in the sets in order to address scalability issues. Sets are depicted by *separate non-overlapping regions* that are arranged *radially* on a circle. These regions can be scaled to reflect the size of its corresponding set. Elements are grouped by their degrees (i.e. number of participating sets in the overlap) and visualized as *histogram bars* inside the regions. Besides, overlaps between sets can be represented in different ways depending on the degree of intersections. Overlaps of degree two are depicted by edges connecting overlapping sets after reordering them based on a heuristic algorithm. However, for higher degrees, overlaps are visualized as bubbles combined with hyper-edges that connect overlapping sets. The size of a bubble designates the size of the overlap. Hyper-edges can be visualized on demand by hovering over the corresponding bubble with the mouse. Aggregated attribute values or measurements of aggregated elements can be encoded by the usage of colors either in the histogram bars, the bubbles or the edges. In addition, Radial Sets allow depicting details of selected elements with summarized tables and histograms. Figure 4 illustrates the logic of this method.

5 Biclustering Visualization Based on Set Visualization Techniques

Visualizing biclustering results is not a trivial process especially when considering the two characteristics of biclustering algorithms; bi-dimensionality and overlaps. Defined visualization methods have to take into consideration these issues.

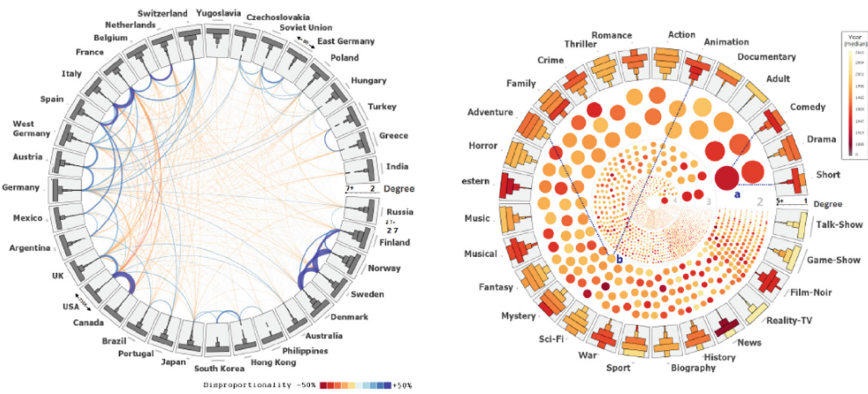


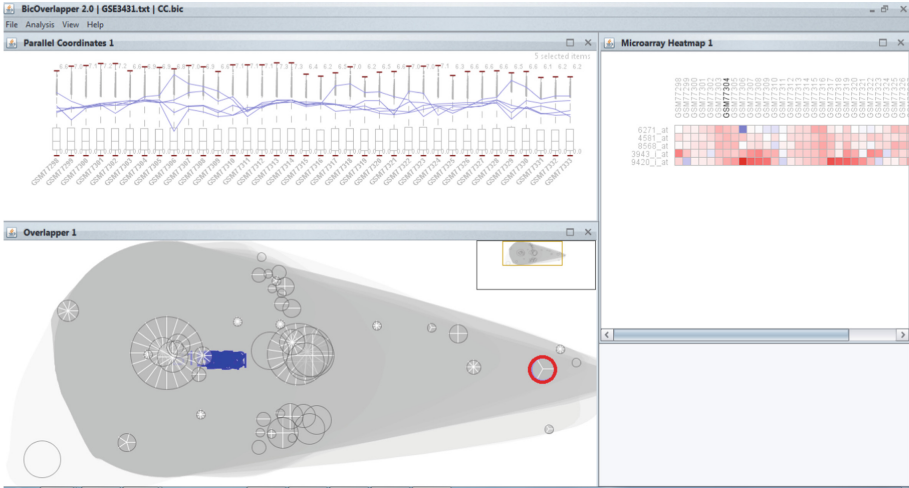
Fig. 4. Radial Sets visualization. Overlaps are shown by edges (on the left) or by bubbles (on the right) [1].

Besides, the different visualization set-to-set relationship techniques described in Sects. 2, 3 and 4 give even several possibilities to better visualize biclustering results. Considering the dimensions of gene expression data that usually vary from hundreds to thousands of genes and from tens to hundreds of conditions [19], Set-to-set relationship approaches can have the potential to resolve problems of scalability (i.e. raise the total number of biclusters that can be shown) and complexity (i.e. show results in simple and readable way especially for overlaps) when drawing biclusters. In fact, a combination of untangling Euler diagrams in its ComED version with heatmap matrix can be a possible solution. As mentioned in [23], untangling Euler diagrams can depict up to 100 sets or more with a total number of elements that can reach to several thousands. In fact, genes and conditions exclusive for a bicluster or shared by several biclusters can be shown via heatmaps while overlaps can be depicted by colored rectangular parts. However, like traditional Euler diagrams, a high rate of overlaps is a severe limit of this method. Besides, UpSet or OnSet techniques can be used in such case. The former can show a number of sets between 40 to 50, which is a typical number of biclustering results from gene expression data. The supported number of elements is interesting also, which can reach 50000 [17] while the latter scales to a large number of sets that surpass 50 but the largest size of each depicted matrix cannot exceed 45×45 (i.e. a total of 2000 elements) [25]. These characteristics render UpSet a good approach to visualize big biclusters with an important number of genes and conditions while OnSet, despite its scalability limitations, can give us a deep vision of biclustering overlaps (i.e. intersections or union) due to its drug and drop principal. As possible propositions, UpSet can depict overlaps between biclusters based on its intersections matrix. Indeed, each column can be considered as *bicluster name* and rows could refer to possible combinations of overlapped biclusters. Clicking on each row can open

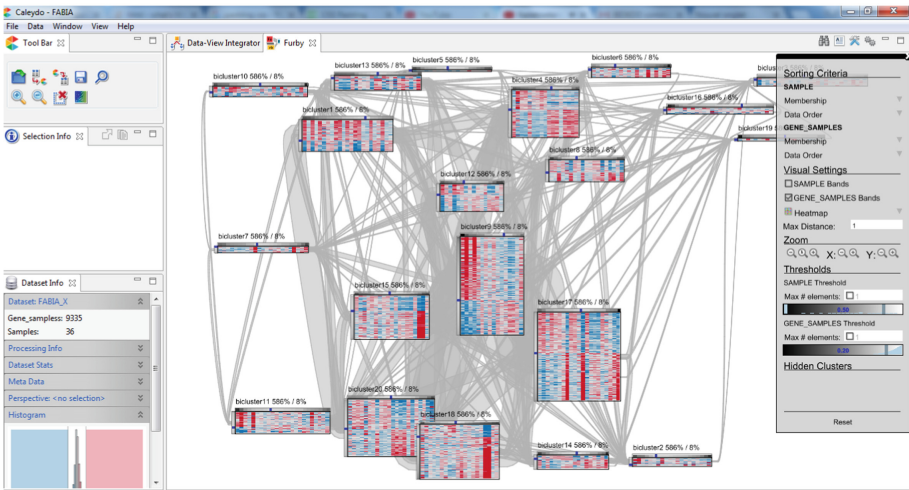
another view that shows genes and conditions in a heatmap matrix or as parallel coordinates. Considering the principle of OnSet, each bicluster can be depicted by a separate matrix where each expression level of gene i under column j can be considered as an element. The number of constructed matrices corresponds to the number of resulting biclusters. Overlaps can be deducted by drag and drop interactions. Radial Sets has the capacity to show 30 sets in the same view at once. The number of elements handled by this method can reach 1 million due to the frequency-based aggregations and to the relative analysis possibilities [1]. As in the UpSet case, Radial Sets can depict biclusters with high sizes. Indeed, Radial Sets can represent biclusters as *uniformly-shaped non overlapping regions* [1] in a circle way while overlaps can be drawn by *bubbles* and *hyper-edges*. To guarantee more scalability, gene expression levels can be visualized by separate heatmaps or parallel coordinates views.

We conclude based on this review that, despite its limitations to depict a lot of overlaps, untangling Euler diagrams surpass all other set-to-set relationship techniques on the number of possible sets that can be visualized at once. We mention that OnSet is not a scalable method to visualize biclusters with big sizes as its top size matrix is 45×45 . Also, the methods of UpSet and Radial Sets inhibit the depiction of containment relations between the sets and their elements because they use separate representations for sets, overlaps between sets and elements of each set [1] and also they are not very scalable to draw important number of sets (maximum of 50 for UpSet and 30 for Radial Sets). As a conclusion, we believe that a good combination of one of the techniques described in this review with traditional biclustering visualization techniques like heatmaps or parallel coordinates can alleviate the biclustering visualization challenge. In fact, we mention two methods that based their approaches on such a combination in the literature. The first one is BicOverlapper that uses a Venn-like representation where biclusters are depicted as *hulls* and overlaps are shown by *intersections of hulls*. Groups of genes and conditions either of separate biclusters or specific overlap are represented by *glyphs*. A glyph is a circle divided in sectors whose numbers depict the number of biclusters where genes and conditions belong to. Genes and conditions specific to a bicluster or overlap between biclusters are pictured by heatmaps and parallel coordinates in a separate view while the second biclustering visualization technique is Furby that combined *matrix-like* representation with *node-link* diagrams. It keeps heatmap matrix to show genes and conditions of each bicluster. Overlaps are depicted by edges between shared genes and conditions for each pair of biclusters. A real usage case of the two methods is depicted in Fig. 5.

We note that the scalability and clarity of visualization are limited for the two techniques especially when the number of results and their overlaps remarkably increase.



(a) BicOverlapper results visualization of 20 biclusters generated with Bimax [21] biclustering algorithm. Overlap between biclusters 4, 7 and 20 is selected (red glyph in the bottom left). The corresponding genes and conditions are shown by heatmaps (top right) and parallel coordinates (top left).



(b) Furby results visualization of 20 biclusters generated with FABIA [12] biclustering algorithm.

Fig. 5. Biclustering results from the yeast gene expression profile along three cell cycles, from experiment GSE3431 [29]. (Color figure online)

6 Conclusion

In contrast to clustering where results are obtained by applying algorithms on the whole of one dimension (genes or conditions) of the gene expression matrix

which simplify the visualization process, biclustering needs more sophisticated visualization techniques in order to facilitate the analysis process that leads to extract nuggets of knowledge required by a bioinformatician. In the light of our review, set and set relation techniques open important opportunities in this field of research. A sophisticated combination between traditional visualization techniques like heatmaps or parallel coordinates and one of the novel set visualization techniques already described, can be a solid solution for the visualization of biclustering results.

References

1. Alsallakh, B., Aigner, W., Miksch, S., Hauser, H.: Radial Sets: interactive visual analysis of large overlapping sets. *IEEE Trans. Vis. Comput. Graph.* **19**(12), 2496–2505 (2013)
2. Alsallakh, B., Micalef, L., Aigner, W., Hauser, H., Miksch, S., Rodgers, P.: Visualizing sets and set-typed data: state-of-the-art and future challenges. In: *Eurographics Conference on Visualization (EuroVis) State of the Art Reports*, pp. 1–21 (2014)
3. Barkow, S., Bleuler, S., Prelic, A., Zimmermann, P., Zitzler, E.: BicAT: a biclustering analysis toolbox. *Bioinformatics* **22**(10), 1282–1283 (2006)
4. Baron, M.E.: A note on the historical development of logic diagrams: Leibniz, Euler and Venn. *Math. Gaz.* **53**(384), 113 (1969)
5. Cheng, K.O., Law, N.F., Siu, W.C., Lau, T.H.: BiVisu: software tool for bicluster detection and visualization. *Bioinformatics* **23**(17), 2342–2344 (2007)
6. Cheng, Y., Church, G.M.: Biclustering of expression data. In: *Proceedings of International Conference on Intelligent Systems for Molecular Biology*, pp. 93–103 (2000)
7. Eisen, M.B., Spellman, P.T., Brown, P.O., Botstein, D.: Cluster analysis and display of genome-wide expression patterns. *Proc. Natl. Acad. Sci.* **95**(25), 1–6 (1998)
8. Gonçalves, J.P., Madeira, S.C., Oliveira, A.L.: BiGGEsTS: integrated environment for biclustering analysis of time series gene expression data. *BMC Res. Notes* **2**, 1–11 (2009)
9. Grothaus, G.A., Mufti, A., Murali, T.M.: Automatic layout and visualization of biclusters. *Algorithms Mol. Biol.* **1**(1) (2006)
10. Heinrich, J., Seifert, R., Burch, M., Weiskopf, D.: BiCluster viewer: a visualization tool for analyzing gene expression data. In: *Bebis, G., et al. (eds.) ISVC 2011. LNCS, vol. 6938*, pp. 641–652. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-24028-7_59
11. Henry, N., Fekete, J.D., McGuffin, M.J.: NodeTrix: a hybrid visualization of social networks. *IEEE Trans. Vis. Comput. Graph.* **13**(6), 1302–1309 (2007)
12. Hochreiter, S., et al.: FABIA: factor analysis for bicluster acquisition. *Bioinformatics* **26**(12), 1520–1527 (2010)
13. Inselberg, A.: The plane with parallel coordinates. *Vis. Comput.* **1**(2), 69–91 (1985)
14. Jin, R., Xiang, Y., Fuhry, D., Dragan, F.F.: Overlapping matrix pattern visualization: a hypergraph approach. In: *Proceedings of IEEE International Conference on Data Mining, ICDM*, pp. 313–322 (2008)
15. Kaiser, S., et al.: BiClust: BiCluster Algorithms. R package version 1.0.2 (2013)

16. Euler, L.: *Lettres a une princesse d'Allemagne sur divers sujets de physique et de philosophie*. Leonhard Euler: Free Download & Streaming: Internet Archive, vol. 1, 1 edn (1772)
17. Lex, A., Gehlenborg, N., Strobel, H., Vuilleumot, R., Pfister, H.: UpSet: visualization of intersecting sets. *IEEE Trans. Vis. Comput. Graph.* **20**(12), 1983–1992 (2014)
18. Madeira, S.C., Oliveira, A.L.: Biclustering algorithms for biological data analysis: a survey. *IEEE Trans. Comput. Biol. Bioinforma.* **1**(1), 24–45 (2004)
19. Padilha, V.A., Campello, R.J.G.B.: A systematic comparative evaluation of biclustering techniques. *BMC Bioinform.* **18**, 55 (2017)
20. Pontes, B., Giráldez, R., Aguilar-Ruiz, J.S.: Biclustering on expression data: a review. *J. Biomed. Inform.* **57**, 163–180 (2015)
21. Prelić, A., et al.: A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics* **22**(9), 1122–1129 (2006)
22. Reid, R.J.D., et al.: Selective ploidy ablation, a high-throughput plasmid transfer protocol, identifies new genes affecting topoisomerase I-induced DNA damage. *Genome. Res.* **21**(3), 477–486 (2011)
23. Riche, N.H., Dwyer, T.: Untangling Euler diagrams. *IEEE Trans. Vis. Comput. Graph.* **16**(6), 1090–1099 (2010)
24. Rodgers, P.: A survey of Euler diagrams. *J. Vis. Lang. Comput.* **25**(3), 134–155 (2014)
25. Sadana, R., Major, T., Dove, A., Stasko, J.: OnSet: A visualization technique for large-scale binary set data. *IEEE Trans. Vis. Comput. Graph.* **20**(12), 1993–2002 (2014)
26. Santamaría, R., Therón, R., Quintales, L.: BicOverlapper 2.0: visual analysis for gene expression. *Bioinformatics* **30**(12), 1785–1786 (2014)
27. Shamir, R., et al.: EXPANDER - an integrative program suite for microarray data analysis. *BMC Bioinform.* **6**, 1–12 (2005)
28. Streit, M., Gratzl, S., Gillhofer, M., Mayr, A., Mittrecker, A., Hochreiter, S.: Furby: fuzzy force-directed bicluster visualization. *BMC Bioinform.* **15**(6), S4 (2014)
29. Tu, B.P., Kudlicki, A., Rowicka, M., McKnight, S.L.: Logic of the yeast metabolic cycle: temporal compartmentalization of cellular processes. *Science* **310**(5751), 1152–1158 (2005)
30. Wilkinson, L., Friendly, M.: The history of the cluster heat map. *Am. Stat.* **63**(2), 179–184 (2009)

Technologies for Information Retrieval (TIR)

Preface

15th International Workshop on Technologies for Information Retrieval, TIR 2018

In recent years, information channels have become increasingly distributed, opinionated, and they vary in their information quality and trustworthiness of the stated facts and arguments. The development of advanced retrieval solutions requires the understanding and the combination of methods from different research areas, including machine learning, (big) data mining, natural language processing, artificial intelligence, user interaction and modelling, and web engineering.

The TIR workshop series provides a platform for presenting and discussing new solutions, novel ideas, and prototypes related to the technologies for information retrieval.

The TIR workshop was held for the fifteenth time. In the past, it was characterized by a stimulating atmosphere and it attracted high quality contributions from all over the world. We are glad that we could compile a program that is both interesting and scientifically demanding. The TIR 2018 Program Committee once again did a great job and provided us with detailed and professional reviews. Without the reviewers' work, the organization of this workshop would not have been possible – thank you sincerely.

Christin Seifert
Michael Granitzer
Benno Stein

Organization

Program Committee

Christin Seifert (Co-chair)	University of Passau, Germany
Michael Granitzer (Co-chair)	University of Passau, Germany
Benno Stein (Co-chair)	Bauhaus-Universität Weimar, Germany
Mikhail Alexandrov	Autonomous University of Barcelona, Spain
Alberto Barrón-Cedeño	Quatar Computing Research Institute, Qatar
Richard Chbeir	UPPA University, France
Mario Döller	University of Applied Science FH Kufstein Tirol, Austria
Marc Franco-Salvador	Universitat Politècnica de València, Spain
Ingo Frommholz	University of Bedfordshire, UK
Christian Gütl	Technical University Graz, Austria
Chaker Jebari	IBRI College of Applied Sciences, Oman
Johannes Jurgovsky	University of Passau, Germany
Roman Kern	Know-Center Graz, Austria
Nedim Lipka	Adobe Research, USA
Adrián Pastor López-Monroy	National Institute of Astrophysics, Optics and Electronics, Mexico
Mihai Lupu	Vienna University of Technology, Austria
Matthias Lux	Klagenfurt University, Austria
Paolo Rosso	Universitat Politècnica de València, Spain
Harald Sack	Karlsruhe Institute of Technology, Germany
Marina Santini	Santa Anna IT Research Institute, Sweden
Ralf Schenkel	University of Trier, Germany
Jörg Schlötterer	University of Passau, Germany
Efstathios Stamatatos	University of the Aegean, Greece
Nadine Steinmetz	Technische Universität Ilmenau, Germany
Eloisa Vargiu	Barcelona Digital, Spain
Henning Wachsmuth	Bauhaus-Universität Weimar, Germany
Rob Warren	Carleton University, Canada



Can We Quantify Domainhood? Exploring Measures to Assess Domain-Specificity in Web Corpora

Marina Santini¹(✉), Wiktor Strandqvist^{1,2}, Mikael Nyström^{1,2},
Marjan Alirezai³, and Arne Jönsson^{1,2}

¹ RISE SICS, Linköping, Sweden

{marina.santini,mikael.nystrom,arne.jonsson}@ri.se,
wiktors.strandqvist@gmail.com

² Linköping University, Linköping, Sweden

{mikael.nystrom,arne.jonsson}@liu.se

³ Örebro University, Örebro, Sweden

marjan.alirezaie@oru.se

Abstract. Web corpora are a cornerstone of modern Language Technology. Corpora built from the web are convenient because their creation is fast and inexpensive. Several studies have been carried out to assess the representativeness of general-purpose web corpora by comparing them to traditional corpora. Less attention has been paid to assess the representativeness of specialized or domain-specific web corpora. In this paper, we focus on the assessment of domain representativeness of web corpora and we claim that it is possible to assess the degree of domain-specificity, or *domainhood*, of web corpora. We present a case study where we explore the effectiveness of different measures - namely the Mann-Withney-Wilcoxon Test, Kendall correlation coefficient, Kullback-Leibler divergence, log-likelihood and burstiness - to gauge domainhood. Our findings indicate that burstiness is the most suitable measure to single out domain-specific words from a specialized corpus and to allow for the quantification of domainhood.

1 Introduction

Web corpora are text collections made of documents that have been retrieved and downloaded from the web. Building web corpora is convenient because the whole process of corpus creation is automated, fast and inexpensive, while the construction of traditional corpora (like the British National Corpus a.k.a. BNC) is normally expensive, time-consuming and require considerable amount of human expertise to decide the ideal combination of documents to store in the corpus. Needless to say that these investments in time, financial resources and human knowledge are well paid-off because traditional corpora are high-quality and long-lasting collections (e.g. the Brown corpus created in the 60's is still used today).

Web corpora are also invaluable, but often time restrictions and limited funding make the manual evaluation of web corpora painstakingly hard and impractical. For this reason, several studies have focussed on quantitative methods and statistical measures to automatically assess the quality of web corpora (e.g. see [8]). Pioneerily, before the start of the web corpora era, Kilgarriff had already spotted this need when he stated: “Measures are needed not only for theoretical and research work, but also to address practical questions that arise wherever corpora are used: is a new corpus sufficiently different from available ones, to make it worth acquiring? When will a grammar based on one corpus be valid for another? How much will it cost to port a Natural Language Processing (NLP) application from one domain, with one corpus, to another, with another?” [14]. More recently, substantial research has been carried out to show that general-purpose web corpora show the same qualities as traditional general-purpose corpora (e.g. see [2,9]). When compared with general-purpose corpus evaluation, the evaluation of domain-specific web corpora is less advanced (see Sect. 2). We would like to start filling this gap because specialized web corpora are widely used in several linguistic disciplines (e.g. in translation studies and lexicography) and they are an important building block of language technology applications (e.g. machine translation, terminology extraction and lexicon induction). Both in linguistics and in language technology, the quality of the results depends on the domain representativeness of the web corpus itself. The research question we would like to start addressing in this paper is then the following: “is it possible to quantify automatically the *domainhood* of web corpora?”. The answer to this question is not straightforward. We make the argument that *a domain-specific web corpus (whatever its domain granularity) should be gauged against a list of core terms that well represent the domain of interest*. We stress that domainhood is only one dimension of the overall corpus quality assessment, and that “corpus quality” is a relative (and not absolute) concept, since the corpus quality should be defined in relation to the purpose for which a corpus has been built and the kind of linguistic phenomena it is meant to represent. In this paper, we present a case study where we compare a medical domain-specific web corpus to a general-purpose traditional reference corpus. We apply well-established measures based on word frequency lists. The measures that we explore are: the Mann-Whitney-Wilcoxon Test, Kendall correlation coefficient, Kullback–Leibler divergence, log-likelihood and burstiness.

2 Related Work

How and to what extent can we assess the quality of web corpora? It seems that a notion like “corpus quality” has become too vague when referring to text collections produced by recent technology. An obvious rebuttal would be: what’s your definition of “corpus quality”?

At the beginning of corpus linguistics, when the collections were manually built and each text was discussed by experts, the quality of a corpus was assessed as the overall representativeness of a corpus for a certain language. This was the

main idea behind the construction of the Brown corpus in the 60's, the British National Corpus in the 90's, and of all the other national general-purpose corpora built in the last decades. Much effort was put into the definition of the parameters that can guarantee the “representativeness” of language use (e.g. see [3]).

Nowadays, when we talk about web corpora, it seems more appropriate to talk about “qualities” rather than a monolithic notion of “quality”. “Qualities” can be defined as dimensions of variation. Domain, genre, style, register, medium, etc. are well-known dimensions of language variation. Although many researchers have worked on the design and assessment of web corpora, no standard metrics has been agreed upon for the automatic quantitative assessment of the different “qualities” of web corpora. In this study, we focus on the dimension of “domain”, that we define as the “subject field” or “area” in which a web document is used. Our aim is somewhat similar to SPARTAN, a technique for constructing specialized corpora from the web by systematically analysing website contents [22]. However, our purpose is not to analyze the domain-specificity of websites as a whole, rather we focus on web documents about chronic diseases. In our experiments, we rely on measures that are well-established and allow for experimental reproducibility, and in this section we provide a short overview of studies where these measures were used.

In his seminal article, Kilgarriff motivates his review of approaches to corpus comparison by asking two crucial questions: “how similar are two corpora?” and “in what ways do two corpora differ?” [14]. Kilgarriff focuses on comparison techniques based on word frequencies, although he acknowledges (as we do) that “a full comparison between any two corpora would of course cover many other matters” [14]. He reviews various statistical measures reaching the conclusion that the Mann-Withney (also known as Wilcoxon) ranks test is a “suitable technique”. He also reviews log-likelihood (a measure based on entropy [7]) and acknowledges that it is “mathematically a well-grounded and accurate measure of surprisiness” but it is more difficult to interpret than the Mann-Withney-Wilcoxon ranks test.

Less skeptical about log-likelihood’s interpretability are Rayson & Garside, who show that this measure can be safely used as a “quick way to find the differences between corpora” and it is more robust than other measures because it is insensitive to corpus size [18].

The Kendall correlation coefficient helps determine whether the observed patterns of two corpora show significant correlations [10].

Another possibility is to use the Kullback–Leibler (KL) distance to assess the “randomness” or “unbiased-ness” of general-purpose corpora and the “bias-ness” of domain-specific corpora, as in [5], where the authors compare domain-specific suparts of the BNC against the whole BNC corpus, showing that KL divergence reliably indicates the difference between them.

Recently, Strandqvist et al. [21] have profitably applied three corpus profiling measures - namely, rank correlation (Kendall and Spearman), Kullback–Leibler divergence, and log-likelihood - to compare the domain-specificity of two medical

web corpora, one bootstrapped with hand-picked term seeds, and the other one bootstrapped with automatically extracted term seeds.

Burstiness has been used in information retrieval and in terminology extraction [13], and more recently for corpus evaluation [20]. Burstiness is a measure that can be utilized for inducing specialized lexicon that is not evenly distributed in a corpus, but appears “in bursts” [23]. Burstiness indicates “how peaked a word’s usage is over a particular corpus of documents” [17], and “bursty words are topical words that tend to appear frequently in documents when some topic is discussed, but do not appear frequently across all documents in a collection” [12]. While bursty words are feared and filtered out when assessing general-purpose corpora [20], we think that they could give a good indication of domain-specificity, and for this reason we include burstiness in our experiments.

3 What’s in a Specialized Corpus? A Case Study

Since “words are not selected at random” [14], we assume that the words included in a corpus represent its content and language use. In our experiments, we use two corpora of Swedish texts, namely a reference corpus and a specialized corpus.

The reference corpus is the Stockholm-Umeå corpus [11] (henceforth SUC, a.k.a. the National Swedish Corpus), while the specialized corpus is a medical web corpus (henceforth *eCare_Sv_01* [19]) which was recently bootstrapped from the web using 155 SNOMED CT¹ terms (unigrams and bigrams) in Swedish indicating chronic diseases. The size of the SUC amounts to 1 million words, whereas the size of *eCare_Sv_01* is approximately 700 000 words. Both corpora are relatively small.

The 155 SNOMED CT terms that we used as seeds were chosen by a domain expert and they well-represent the domain of interest, since they indicate chronic diseases that are classified as such in the SNOMED CT ontology. To build the corpus, we used these terms as queries in Google.se search engine. The web corpus was downloaded with BootCat [1] (Customized URLs option). Using regular search engines (like Google, Yahoo or Bing) and term seeds (as queries) to build a corpus is handy, but it also has some caveats that depend on the design or distortion of the underlying search engine [22]. These caveats affect the content of web corpora since it might happen that irrelevant documents are included in the collection, especially when searching for very specialized terms. Since manual and qualitative inspections are often prohibitive, the automatic assessment of the domain-specificity of a corpus crawled from the web is potentially very useful.

For the evaluation, we used a gold standard term list containing the tokenized SNOMED CT term seeds, without duplicates. This means that SNOMED CT terms like “kronisk anemi” (en: chronic anemia) and “kronisk artrit” (en: chronic arthritis), in the gold standard will be represented by three entries, namely “kronisk”, “anemi” and “artrit”. All in all, the gold standard term list includes 165 terms².

¹ SNOMED CT browser is available at <http://browser.ihtsdotools.org/>.

² The lists of the selected 155 SNOMED CT terms and the tokenized gold standard (165 entries) are available here: <http://santini.se/eCareCorpus/home.htm>.

It makes sense to use domain-specific terms for both bootstrapping the web corpus and for evaluating its domainhood because the terms used as seeds (source terms) should be found in non-trivial proportions in the final corpus to be sure that the corpus is representative of the domain.

4 Metrics

In the experiments presented in this paper we take the bag-of-words approach and we used metrics based on word frequency lists. We report results and discussion for the following measures: Mann-Whitney-Wilcoxon Test, Kendall correlation coefficient (τ), Kullback–Leibler (KL) divergence, log-likelihood and burstiness.

Word Frequency Lists. A word frequency list (a.k.a. unigram lists) can be seen as a “compact representation of a corpus, lacking much of the information in the corpus but small and easily tractable” [16]. (We used the R packages “tm” and “quanteda” to create frequency lists).

Mann-Whitney-Wilcoxon Test. The Mann-Whitney-Wilcoxon Test (non-parametric test) helps decide whether the distributions of the two corpora are identical without assuming them to follow the normal distribution. (We used the R function “wilcox.test()” to calculate the test).

Kendall Correlation Coefficient (τ). Kendall correlation coefficient (τ) is a non-parametric measure of correlation between two rankings. τ is a probability value which indicates the difference between the probability that the observed data are in the same order and the probability that the observed data are not in the same order. There are several variations of Kendall’s τ (they differ only in the way that they handle ties). (We used the R function “cor.test()” with method = “kendall” to calculate the test).

Kullback–Leibler (KL) Divergence. The convenience of KL divergence (a.k.a. relative entropy) lies in its ability to quantify how “distant” an estimation of a distribution may be from the true distribution. The KL divergence is non-negative and equal to zero if the two distributions are identical. In our context, the closer the value is to 0, the more similar two corpora are. (We used the R function “KL.empirical()”, (\log_2), package “entropy” to compute KL divergence).

Log-Likelihood (LL). Log-likelihood (a.k.a. G^2) [7] can be used for corpus profiling [18]. It is a measure based on a contingency table and compares the expected values in two corpora under observation. The larger the LL score of a word, the more different its distribution in the two corpora.

Burstiness. Burstiness helps identify words that are important in certain documents, but that are unevenly distributed in the corpus as a whole. Several burstiness formulas exist. In the experiments reported in this paper, we use Church & Gale’s formula [4], including the modification proposed by [12] (i.e. the use of relative frequencies rather than absolute frequencies).

5 Experiments

In this section, we present the experiments and discuss the results.

5.1 Corpus Profile: Sorted Frequency Lists

We computed the relative frequencies of SUC and *eCare_Sv_01* after having removed stopwords (we used the standard Swedish stop list in R). Then we divided each frequency of occurrence by the total number of words in the corpus and we normalized by multiplying by 1 million to get the frequencies of words per millions (wpm) [15]. When visually compared, the sorted frequency lists immediately show the different composition of the two corpora. Common words appear at the top of SUC, while the top ranked words in *eCare_Sv_01* are domain-specific words such as *kronisk* (en: chronic), *behandling* (en: treatment), *patienter* (en: patients), as shown in Table 1.

Table 1. Sorted frequencies (wpm)

Rank	SUC	Freq	eCare	Freq
1	också (<i>also</i>)	2266.12	kronisk (<i>chronic</i>)	4224.16
2	andra (<i>other</i>)	1938.1	behandling (<i>treatment</i>)	4132.86
3	finns (<i>exist/be</i>)	1614.37	hos (<i>at (locative)</i>)	3669.21
4	år (<i>year</i>)	1588.68	patienter (<i>patients</i>)	2741.92

5.2 Rank Correlation

In Fig. 1, we compare the rankings (with ties) of the top 1000 words of the two corpora. The cut-off at 1000 is arbitrary and it is simply used to skim off low frequencies. Figure 1 shows that the relation between the top 1000 words in the two corpora tends to be negative, i.e. when the rank of an *eCare_Sv_01* word is high, the rank of the same word is low in the SUC and vice versa (see the aggregation of points that are parallel to x and y axes). However, several words (see the clouds of points in the middle) show a positive relation, i.e. when the rank increases in *eCare_Sv_01*, it also increases in SUC. The solid line in Fig. 1 depicts the LOWESS smoother (a.k.a. Locally Weighted Scatterplot Smoothing). The LOWESS function (R function: “lines(lowess())”) creates a smooth line through the scatter plot to help detect the relationship between

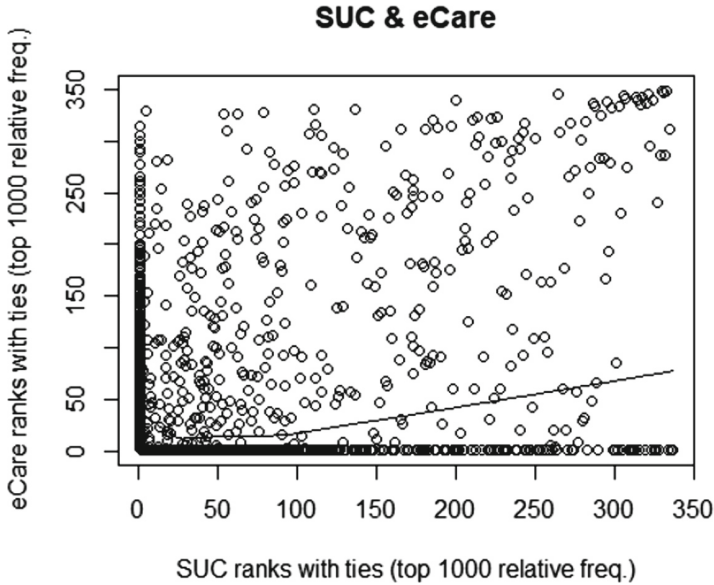


Fig. 1. Frequency scatterplot

variables and foresee trends [10]. In our scatterplot, the LOWESS smoother shows a slight positive relationship. Although informative, the scatterplot does not tell us whether a statistically-significant correlation exists between the two corpora. For this reason, we ran two non-parametric rank correlation measures, namely the Mann-Whitney-Wilcoxon Test and Kendall τ and we measured their statistical significance at $p < 0.05$, two-sided.

Mann-Whitney-Wilcoxon Test. For the Mann-Whitney-Wilcoxon Test, the p-value of the test is 0.019, which is less than the significance level of $p = 0.05$. We can conclude that the median rank of a word in SUC is significantly different from the median rank in *eCare_Sv_01*.

Kendall τ . For Kendall τ , the p-value of the test is 0.000000003122 (p-value in R: 3.122e-09) which is less than the significance level $p = 0.05$.

Both tests indicate that the distribution of the words in *eCare_Sv_01* is significantly different from the distribution of words in SUC.

5.3 Kullback–Leibler (KL) Divergence

A smoothing value of 0.01 was applied to the normalized relative frequencies (wpm). The KL divergence between SUC and *eCare_Sv_01* is 5.80, which (unsurprisingly) indicates a large divergence between a general-purpose SUC and the domain-specific *eCare_Sv_01*.

5.4 Log-Likelihood (a.k.a. G^2)

We computed log-likelihood (LL) on smoothed frequencies. A LL score of 3.8415 or higher is significant at the level of $p < 0.05$ and a LL score of 10.8276 is significant at the level of $p < 0.001$ [6]. We used these thresholds to cut-off the list of LL scores (sorted by decreasing values). We selected the LL scores with a value of 3.8415 or higher and with a value of 10.8276 or higher. We got a list of 1542 records in the first case, and a list of 1514 records in the second case. We computed Precision@ against the gold standard for both lists. An identical value was returned for both, i.e. 0.048. The values for Precision@ are somewhat similar to the values returned by the Jaccard and Dice coefficients (i.e. 0.036 and 0.069 respectively). The intersection between the selected LL scores and the gold standard is 58, i.e. 35.15%.

This experiment provided some useful insights. First, since the words in the frequency lists are not lemmatized, some mismatches are caused by differing morphological forms. This problem can be addressed by lemmatizing the top-ranked frequent words before matching them against the gold standard. Second, it was naive to expect that the top-ranked words were only chronic illnesses. Some words often co-occur with chronic diseases and are domain-specific, like “patient” or “treatment”, but they are not in the gold standard because they do not designate a chronic disease. Arguably, the current gold standard is too selective. Last but not least, we realized that the LL scores do not show the “direction” of comparison. For example, a word like “anemi” could, in principle, have a higher distribution in SUC (reference corpus) rather than in *eCare_Sv_01* (target corpus). For this reason we think that log-likelihood might not be the best approach to detect the degree of a domainhood of a target corpus.

5.5 Burstiness

As mentioned above, burstiness singles out content words that tend to appear in some documents, but that are not spread out evenly across the whole corpus. This characterization well fits *eCare_Sv_01* where each chronic disease is discussed in some of the documents, but not in all of them. We calculated burstiness separately for *eCare_Sv_01* and for SUC and compared the results. For each corpus, we sorted the burstiness values by decreasing order and we took the top 2105 bursty words. The expectation is that *eCare_Sv_01* should contain many words listed in the gold standard, while SUC should show a limited overlap with the golden standard. This expectation is indeed met. 90 terms out of 165 (i.e. 54.5%) are top-ranked in the list of *eCare_Sv_01*’s bursty words. It is also to be noted that the range of values of bursty words differs a lot across the two corpora. In *eCare_Sv_01*, the first top-ranked 2105 words have burstiness values ranging from 0.1 to 0.005, while in SUC burstiness values range from 0.044 to 0.0022 for the same range. Table 2 reports the number of words in common with the gold standard, i.e. the intersection, (col.2), the Jaccard coefficient (col.3), the Dice coefficient (col. 4) and Precision@2105 (col. 5). Certainly, the values of the two coefficients and the value

of Precision@ do not make justice of the magnitude of the overlap since their calculation takes into account the number of unmatched items, which in our case are many because the gold standard list is much shorter than the list of ranked words. These are the bursty 90 words that *eCare_Sv_01* shares with the gold standard: *andningsinsufficiens, anemi, artrit, artropati, atelektas, atrofisk, basalcellscancer, beryllios, blefarit, bronkiolit, clonorchiasis, continua, cystica, cystit, cystitis, dakryocystit, depression, dermatit, dysfagi, eksem, emfysem, exoftalmus, eksplosivitet, faryngit, fluoros, gastrit, giktartrit, gingivit, glomerulonefrit, glossit, hemikrania, hepatit, hyperglykemi, hyperkapni, hypernatremi, hyponatremi, infektionssjukdom, intermittent, jaccouds, juvenil, kammartakykardi, kartagensers, kolecystit, kolit, konjunktivit, kontaktdermatit, kronisk, krupp, laryngotrakeit, lipidnefros, lungembolism, lungemfysem, mastit, mastocytos, mastoidit, meningokockemi, metrit, missfall, mycetom, neutropeni, njursvikt, obliterativ, orkit, osteomyelit, ozena, pankreatit, paraplegi, parodontit, paronyki, perikardit, polyserosit, postkardiotomisyndrom, prostatit, psoriasisartrit, rhinitis, rinit, schizofreni, schnitzlers, sicca, silikos, spondyloartropati, syndrom, synovit, testistorsion, tics, trakeit, trakeobronkit, tyreoidit, urtikaria, vulvit.*

Table 2. Comparison between bursty words and the gold standard

	Intersection	Jaccard	Dice	Precision@2105
SUC	1	0.000440	0.00088	0.00001
eCare	90	0.04128	0.07929	0.03590

5.6 Discussion

We have seen that the comparison between sorted frequency lists does give a coarse but grounded idea of the domain-specificity of *eCare_Sv_01*. Both correlation tests confirm that the distributions of the word frequencies of the two corpora are not positively correlated and this difference is statistically significant. This finding is further supported by the value returned by the KL divergence, which indicates a large distance between SUC and *eCare_Sv_01*.

However, frequency profiling, rank correlation tests and KL divergence do not tell us how representative the *eCare_Sv_01* corpus is of the domain it is meant to represent.

Sorted LL scores single out words with different distributions in the two corpora. However, these values do not allow us to measure the degree of domain-specificity on an individual corpus.

Burstiness gives a much clearer indication of the domain topics that are discussed in *eCare_Sv_01*. The intersection between *eCare_Sv_01*'s bursty words and the gold standard is an encouraging 54%, a value that can be increased with some additional preprocessing, for example by lemmatizing the bursty words before evaluating them against the gold standard, and by including in the gold standard medical expressions such as “patient”, “treatment”, that do not indicate chronic diseases but are indeed domain-specific.

6 Conclusion and Future Work

In this paper, we explored measures to assess domainhood in web corpora. Domainhood indicates the degree of domain-specificity of a specialized corpus. We carried out comparative experiments where the domainhood of a traditional general-purpose national corpus (SUC) and a specialized medical corpus (*eCare_Sv_01*) were measured against a gold standard list of chronic diseases that represents the target domain. Although the outcome of this experiment is intuitive, we empirically showed that the intuition is supported by counts. Our findings indicate that burstiness is a suitable measure to assess the domain-specificity of a corpus.

We are currently extending these experiments to other languages and additional corpora. We are also working on a new gold standard and on the implementation of additional burstiness formulas.

Acknowledgement. This research was supported by E-care@home, a “SIDUS - Strong Distributed Research Environment” project funded by the Swedish Knowledge Foundation. Project website: <http://ecareathome.se/>.

References

1. Baroni, M., Bernardini, S.: BootCat: bootstrapping corpora and terms from the web. In: LREC (2004)
2. Baroni, M., Bernardini, S., Ferraresi, A., Zanchetta, E.: The wacky wide web: a collection of very large linguistically processed web-crawled corpora. *Lang. Resour. Eval.* **43**(3), 209–226 (2009)
3. Biber, D.: Representativeness in corpus design. *Literary Linguist. Comput.* **8**(4), 243–257 (1993)
4. Church, K.W., Gale, W.A.: Poisson mixtures. *Nat. Lang. Eng.* **1**(2), 163–190 (1995)
5. Ciaramita, M., Baroni, M.: A figure of merit for the evaluation of web-corpus randomness. In: Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics (2006)
6. Desagulier, G.: *Corpus Linguistics and Statistics with R*. Springer, Heidelberg (2017). <https://doi.org/10.1007/978-3-319-64572-8>
7. Dunning, T.: Accurate methods for the statistics of surprise and coincidence. *Comput. Linguist.* **19**(1), 61–74 (1993)
8. Ferraresi, A., Zanchetta, E., Baroni, M., Bernardini, S.: Introducing and evaluating ukWaC, a very large web-derived corpus of English. In: Proceedings of the 4th Web as Corpus Workshop (WAC-4) Can We Beat Google, pp. 47–54 (2008)
9. Fletcher, W.H.: Implementing a BNC-compare-able web corpus. *Building and Exploring Web Corpora*, pp. 43–56 (2007)
10. Gries, S.T.: Elementary statistical testing with R. In: Krug, M., Schlüter, J. (eds.) *Research Methods in Language Variation and change* (2013)
11. Gustafson-Capková, S., Hartmann, B.: *Manual of the Stockholm Umeå corpus version 2.0*. Stockholm University (2006)
12. Irvine, A., Callison-Burch, C.: A comprehensive analysis of bilingual lexicon induction. *Comput. Linguist.* **43**(2), 273–310 (2017)

13. Katz, S.M.: Distribution of content words and phrases in text and language modelling. *Nat. Lang. Eng.* **2**(1), 15–59 (1996)
14. Kilgarriff, A.: Comparing corpora. *Int. J. Corpus Linguist.* **6**(1), 97–133 (2001)
15. Kilgarriff, A.: Simple maths for keywords. In: *Proceedings of the Corpus Linguistics Conference*, Liverpool, UK (2009)
16. Kilgarriff, A.: Comparable corpora within and across languages, word frequency lists and the KELLY project. In: *Proceedings of the 3rd Workshop on Building and Using Comparable Corpora*, pp. 1–5 (2010)
17. Pierrehumbert, J.B.: Burstiness of verbs and derived nouns. In: Santos, D., Lindén, K., Ng’ang’a, W. (eds.) *Shall We Play the Festschrift Game?*, pp. 99–115. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-30773-7_8
18. Rayson, P., Garside, R.: Comparing corpora using frequency profiling. In: *Proceedings of the workshop on Comparing Corpora*, pp. 1–6. Association for Computational Linguistics (2000)
19. Santini, M., Jönsson, A., Nyström, M., Alireza, M.: A web corpus for eCare: collection, lay annotation and learning-First results. In: *Proceedings of the 2nd International Workshop on Language Technologies and Applications (LTA17)*. FedCSIS (2017)
20. Sharoff, S.: Know thy corpus! Exploring frequency distributions in large corpora. In: Diab, M., Villavicencio, A. (eds.) *Essays in Honor of Adam Kilgarriff*. Text Speech and Language Technology Series. Springer, Heidelberg (2017)
21. Strandqvist, W., Santini, M., Lind, L., Jönsson, A.: Towards a quality assessment of web corpora for language technology applications. In: *Proceedings of TISLID18 - Languages For Digital Lives and Cultures*. Ghent University, Belgium (2018)
22. Wong, W., Liu, W., Bennamoun, M.: Constructing specialised corpora through analysing domain representativeness of websites. *Lang. Resour. Eval.* **45**(2), 209–241 (2011)
23. Zhao, Z., Mei, Q.: Questions about questions: an empirical analysis of information needs on twitter. In: *Proceedings of the 22nd International Conference on World Wide Web*, pp. 1545–1556. ACM (2013)



A Case Study of Closed-Domain Response Suggestion with Limited Training Data

Lukas Galke¹^(✉), Gunnar Gerstenkorn², and Ansgar Scherp³

¹ University of Kiel, Kiel, Germany
lga@informatik.uni-kiel.de

² University of Potsdam, Potsdam, Germany
gerstenkorn@uni-potsdam.de

³ University of Stirling, Stirling, Scotland, UK
ansgar.scherp@stir.ac.uk

Abstract. We analyze the problem of response suggestion in a closed domain along a real-world scenario of a digital library. We present a text-processing pipeline to generate question-answer pairs from chat transcripts. On this limited amount of training data, we compare retrieval-based, conditioned-generation, and dedicated representation learning approaches for response suggestion. Our results show that retrieval-based methods that strive to find similar, known contexts are preferable over parametric approaches from the conditioned-generation family, when the training data is limited. We, however, identify a specific representation learning approach that is competitive to the retrieval-based approaches despite the training data limitation.

1 Introduction

Natural language processing digital assistants and tools provide helpful information or automatically suggest responses in a conversation. Most of these assisting tools operate in an open domain without any assumptions. We investigate a similar setting in a closed domain: users of a digital library asking librarians for support regarding the search for literature. Since libraries become more and more digital, these support requests are often made in a digital context, i. e., a chat system. However, many of these requests are repetitive and responding may become tedious. We envision that librarians can benefit from a tool that suggests appropriate responses given the specific request. This way, the repetitive work can be reduced to a minimum of selecting one of few suggested responses. We assume that when operating in a closed domain, finding a well-suited response candidate is easier than in an open domain. However, the restriction to a closed domain also limits the amounts of available training data. That motivates us to re-evaluate different techniques for response suggestion in a setting with limited training data. The amount of training data is a crucial factor for machine learning methods. Thus, we strive to identify those methods that specifically perform well with small amounts of available training data.

We operate on real-world chat transcripts from ZBW—Leibniz Information Centre for Economics¹ obtained through QuestionPoint². QuestionPoint is a platform, that many digital libraries use to process support requests. The transformation from chat transcripts to structured training data involves multiple steps, difficulties, and design decisions that we describe in this paper. Subsequently, we compare retrieval-based, representation learning, and sequence-to-sequence approaches for response suggestion [16, 19]. The core task in response suggestion is to find the most probable response to a given question, which can be formalized as: $r^* = \arg \max_{r \in R} P(r | q)$ for a question q and a set of possible responses R [16].

Evaluating different approaches for response suggestion on real-world data is challenging because the responses of training and test data are different. We therefore chose to evaluate the different approaches with the translation metric BLEU [15]. The overlap of word n-grams is used to assess the quality of a suggested response compared to the true response in the data.

In summary, the contributions of this paper are the following:

- A text-processing pipeline that generates question-answer pairs from raw (XML) chat transcripts. We highlight the difficulties and design decisions in constructing such pipelines.
- An empirical evaluation of retrieval-based, conditioned-generation, and representation learning approaches for response suggestion in a closed-domain scenario with small amounts of training data.
- Evidence, that retrieval-based approaches are preferable in scenarios with limited amount of training data. We show that representation learning approaches are less prone to lack of training data than sequence-to-sequence architectures.

After giving a brief overview on the related work in Sect. 2, we describe the generic text-processing pipeline in Sect. 3. In Sect. 4, we describe the employed models for response suggestions, before we depict our experiments in Sect. 5. The results are presented in Sect. 6 and discussed in Sect. 7, before we conclude in Sect. 8.

2 Related Work

In the following, we briefly review the related work on conditioned-generation and representation learning approaches that are relevant for response suggestion scenarios.

Ritter et al. [16] proposed to use statistical machine translation approaches for conditioned response generation. The source sequence is encoded into a latent representation, which is in turn used to generate the target sequence. The models were trained on Twitter utterances and responses. Vinyals et al. [21]

¹ <http://zbw.eu>.

² <https://www.oclc.org/en/questionpoint.html>.

extended this sequence-to-sequence based approach by taking the chat history into account. Al-Rfou et al. [2] explored a prediction-based approach that exploits input, context, and author information on a Reddit dataset of 2.1 billion utterances. Wu et al. [23] investigated the use of dynamic per-question vocabularies. Xu et al. [24] jointly trained an adversarial discriminator to penalize non-informative answers.

Dedicated representation learning approaches for response suggestion aim to learn (joint or separate) representations for the questions and the responses. The training objective comprises learning a representation that leads to a high similarity score for the correct response and lower scores for other responses, given a certain question. The similarity is typically computed by dot-product or cosine similarity of latent representations such as in *StarSpace* [22] and *DSSM* [9]. Both *StarSpace* and *DSSM* are a general representation learning approaches that use cosine similarity as scoring function. All variants have in common that the training is performed by negative sampling in favor of (hierarchical) softmax [14]. Google Research has conducted two studies considering a Smart Reply mechanism for Gmail [8, 10]. The first work relied on a sequence-to-sequence (i.e., conditioned-generation) approach [10], while the second work uses an approach that is purely based on representation learning [8].

From the related work, we observe that conditioned-generation and representation learning for response suggestion are well-studied topics. However, most approaches consider open-domain applications, e.g., Twitter, Reddit, or Gmail, where large amounts of training data is available. In this paper, we are instead targeting closed-domain scenarios such as a support system of a library, in which considerably less training data is available.

3 Text-Processing Pipeline

In this section, we describe our text-processing pipeline starting at raw XML query log data. After extracting the transcripts (Sect. 3.1), we join consecutive utterances (Sect. 3.2), and then generate question-answer pairs (Sect. 3.3). Finally, we conduct a language detection step (Sect. 3.4), before the pairs are supplied to the response suggestion models.

3.1 Transcript Extraction

We operate on XML exports from QuestionPoint³. QuestionPoint is a dedicated platform, on which librarians answer support requests of the library’s patrons (the user). The XML transcripts consist of an ordered sequence of utterances. Each utterance is associated with an agent. The agent is either a librarian or the user. The transcript opens with the user’s initial question. It also comprises status messages, such as “Patron’s screen name: X ”, “Library ended chat session.”, “Set Resolution: Y ”. These status messages are filtered via regular expressions.

³ <https://www.oclc.org/en/questionpoint.html>.

We further clean all utterances from HTML tags such as `
` and `<p>`. Our data consists of 3,246 XML transcripts of chat sessions regarding the literature search service EconBiz of ZBW.

3.2 Joining Consecutive Utterances

It is possible that two consecutive utterances were made by the same responsible agent. From inspecting the data, we learn that this is often the case when a domain expert first replies with a rather short answer such as “Hello, let me first read the question” or “I will look it up in our catalogue”. Afterwards, the domain expert provides the actual answer. We assume that it is beneficial, when the actual answer is pursued as soon as possible. Otherwise, all considered approaches would tend to yield rather generic yet uninformative suggestions. We therefore decide to consistently concatenate consecutive utterances from the same agent.

3.3 Question-Answer Pair Generation

At the current point, we have a set of transcripts of alternating (library and patron) utterances. The goal is to create a set of paired training data from these utterances. We considered multiple approaches for generating paired training data from the sequences of alternating utterances. The generation of training data is not trivial because each librarian’s response may not only refer back to the preceding utterance of the user, but also the context as a whole or the user’s initial question.

We continue with generating exactly one question-answer pair for each librarian answer, in which the question is the previous utterance of the user. This approach, in some cases, suffers from a loss of context due to singular “Ok.”-fashioned utterances. For instance, the librarian initially response with a greeting message and asks for time, then the user says “Ok.”, which is then used as context for the next answer. Still, this approach lead to most reasonable response suggestion models in our pre-experiments. Using this approach, we generated 14,059 question-answer pairs from the 3,246 chat transcripts.

3.4 Language Detection

In the final data preparation step, we apply language detection. The majority of chat transcripts consisted of German utterances. Still a considerable amount of transcripts also held English utterances. To detect the language of a question-answer pair, we employed Naive Bayes classifier operating on character n-grams. We used the implementation of PyCLD2⁴, which in turn utilizes the Compact Language Detector 2⁵ by Google. Among the 14,059 generated question-answer pairs, 11,511 were recognized as German and 2,795 were recognized as English language. The language detection is, however, not necessarily exclusive: 246 pairs were classified as both German and English.

⁴ <https://github.com/aboSamoor/pycltd2>.

⁵ <https://github.com/CLD2Owners/cld2>.

4 Response Suggestion

In the following, we introduce the employed approaches for response suggestion from three different families: information retrieval in Sect. 4.1, condition-generation in Sect. 4.2, and pure representation learning in Sect. 4.3.

4.1 Retrieval Models

We consider employing retrieval models for response suggestion. After indexing the paired training data, we use the patrons' questions as query to the known questions. We employ two retrieval models: TF-IDF [17], a well-known baseline from Information Retrieval [13], and word centroid distance (WCD) [6, 12] which uses word embeddings [14] to compute a similarity score. As investigated in our prior work [6], we use inverse document frequency (as in TF-IDF) to re-weight term counts for computing the word centroid distance.

Using these building blocks, we evaluate multiple combinations as retrieval models: TF, TF-IDF, WCD, and WCD-IDF. In addition, we evaluate the aforementioned retrieval models with a disjunctive term matching operation (abbreviated with the prefix M) that reduces the candidates to those that have at least one term in common with the query, which is in our case the question. For all retrieval models, we use cosine similarity to retrieve the most similar context to the given question and return the known answer to this context.

To compute the word vector centroids, we use German *fastText* [4] word vectors trained on CommonCrawl and Wikipedia⁶ [7]. For the experiment on English utterances, we use *Word2Vec* word vectors trained on Google News [14].

We further experiment with employing k-nearest neighbors (k-NN) with $k \in \{1, 3, 5, 7\}$ to predict the response to a question. The k nearest question-response pairs are retrieved via cosine similarity. Each neighbor is associated with a specific response. The neighbors then cast a similarity-based vote on the response to return. It is possible that two or more neighbors point to the same (tokenized) response. In this case, the predicted response may differ from the ones of the retrieval-based approach, which is comparable to k-NN using a single nearest neighbor.

4.2 Conditioned-Generation

Generating an answer based on a question can be regarded as a sequence transduction task [16]. Common sequence-to-sequence models [19] use one recurrent neural network to encode the question into a hidden state. Then, another recurrent neural network decodes the hidden state into the target (known) answer. Sequence-to-sequence models can also be applied as conditioned response generation [16]. For all following experiments, we use the Tensorflow [1] sequence-to-sequence implementation⁷ with its default hyperparameters. Those were two

⁶ <https://github.com/facebookresearch/fastText/blob/master/docs/crawl-vectors.md>.

⁷ <https://www.tensorflow.org/versions/r1.1/tutorials/seq2seq>.

hidden layers of 128 hidden units with a 0.2 dropout and a learning rate of 1. The vocabulary for the sequence-to-sequence model was generated on the training data only.

4.3 Representation Learning

We report the results of two approaches for response suggestion from the related work [8]. The authors label the two approaches as *joint* and *dotproduct*, respectively. Both of them rely on producing a score, given word n-grams of the question and the response. The score determines how appropriate the specific response is to the question. In the *joint* approach, the bag-of-gram representations of the question and the response are concatenated and then fed to three hidden layers with ReLU activations and a final layer outputs the score. In the *dotproduct* approach, the questions and responses are separately encoded into vector representations before using cosine similarity for scoring. In this case, we use three hidden layers with Tanh activations as in the original work [8]. The computed score is optimized to be high for matching question-answer pairs and low for negative samples. We chose to use soft-margin loss as objective to rank the correct response higher than the responses of negative samples.

We use unigrams and bigrams as initial representation. We chose a hidden layer size of 100 and apply dropout [18] with a drop probability of 0.2 on each intermediate hidden layer. We further experimented with employing pre-trained word vectors for the initial conversion between words and vectors. This, however, did not lead to an improvement. All variants are trained for 50 epochs with Adam [11] optimizer using an initial learning rate of 0.001. We experimented with using between one and five negative samples.

5 Experiments

In the following, we describe our experiments conducted on both the German and the English subset of generated question-answer pairs. After briefly introducing the general experimental procedure in Sect. 5.1, we provide details on word count statistics in Sect. 5.2, before we describe our evaluation metrics in Sect. 5.3.

5.1 Experimental Procedure

For conducting our experiments, we split the data into 90% training pairs and 10% test pairs. The sequence-to-sequence approach internally separates another 10% from the training data for validation. After the training step, we supply each question of the test set to the respective method and compare its top suggested response against the real response.

Table 1. Word count statistics of questions and answers

Data	Min	Q25	Q50	Q75	Max	Mean	SD
English sources	1	4	9	18	386	14.19	19.20
English targets	1	10	19	34	697	31.34	50.84
German sources	1	4	9	18	386	13.99	16.77
German targets	1	9	17	30	790	24.53	34.14

5.2 Dataset Characteristics

The basic statistics of questions’ and answers’ word count are provided in Table 1 for both the German and the English subset. The median of the word counts is between 4 and 10. Most of utterances are rather short, the median is lower than the mean length of utterances (between 16 and 50). The longest question is 386 words long while the longest answer consists of 697 words in English and 790 words in German. We manually removed one pair, in which a whole email-conversation of more than 2,000 words has been copied into the chat.

5.3 Evaluation Metrics

Since the responses of our generated datasets are not organized along classes, we use evaluation metrics from machine translation. To evaluate the suggested responses, we employ the BLEU (bilingual evaluation understudy) score, which was proposed by Papineni et al. [15]. The BLEU score computes the precision of word n -grams of the hypothesis against one or more references. In our case, we use a single reference, which is the correct response given by the data, for each question-response pair. We reuse the Corpus-BLEU implementation provided by NLTK⁸. The Corpus-BLEU score aggregates word n -gram counts over the whole corpus before performing a single division. A brevity penalty for the length difference between question and reference answer is imposed. We abbreviate the Corpus-BLEU score in our results by BLEU. We use smoothing function 5 from the work of Chen and Cherry [5], which averages the n -gram precision scores over $n - 1, n, n + 1$ before computing the BLEU score. This method attained the highest correlation with human judgements without introducing a new parameter for the metric. We also evaluate the sole word choice and fluency part of BLEU: modified precision [15] (p_n) on word uni-, bi-, and trigrams. The modified precision score refers to the fraction of words in the suggested response that also appear in the actual response, bounded by the number of times a word appears in the actual response. This allows a more detailed inspection of the models, whereas a single corpus wide BLEU score would paint a superficial picture.

⁸ http://www.nltk.org/_modules/nltk/translate/bleu_score.html.

Table 2. BLEU modified precision scores using uni- (p_1), bi- (p_2), and trigrams (p_3) along with Corpus BLEU score of the response suggestion methods on the **German** question-answer pairs.

Model	p_1	p_2	p_3	BLEU
<i>Traditional retrieval</i>				
TF	28.53	18.65	16.70	23.76
TF-IDF	28.73	19.39	17.54	23.57
M-TF	28.05	18.29	16.45	23.56
M-TF-IDF	28.48	19.03	16.88	23.19
<i>Retrieval with KNN</i>				
1-NN	29.27	19.68	17.60	23.49
3-NN	29.47	20.15	18.12	23.35
5-NN	28.47	19.04	17.02	23.08
7-NN	28.04	18.71	16.70	23.08
<i>Retrieval with word vectors</i>				
WCD	26.87	17.65	16.22	23.70
WCD-IDF	27.55	18.25	16.35	24.17
M-WCD	27.45	17.94	16.31	23.51
M-WCD-IDF	28.16	18.67	16.92	24.37
<i>Representation learning</i>				
Dotproduct-n1	11.43	1.18	0.38	8.04
Dotproduct-n3	7.36	0.69	0.20	7.61
Joint-n3	26.06	16.95	15.26	21.19
Joint-n4	27.21	17.25	15.71	17.99
Joint-n5	24.23	16.13	14.77	18.11
<i>Conditioned-generation</i>				
seq2seq	11.51	2.75	0.99	7.42

6 Results

Table 2 shows the results for the experiment on the German question-answer pairs. We report the mean modified precision values p_1 , p_2 , p_3 , and BLEU scores for each of the models described in Sect. 4. KNN with $k = 3$ is the top-performing approach considering p_1 , p_2 , and p_3 . Considering BLEU score, the M-WCD-IDF retrieval model yields the highest score of 24.37. The *joint* representation learning approach is the closest competitor to the retrieval-based methods and attains BLEU scores of up to 21.19.

Table 3 shows the BLEU and modified-precision scores of the methods on the English question-answer pairs. The approaches M-WCD-IDF and M-TF-IDF yield the highest p_1 , p_2 , p_3 scores. The KNN-based approaches consistently yield the highest BLEU scores. With 5 nearest neighbors, the BLEU score peaks

Table 3. BLEU modified precision scores using uni- (p_1), bi- (p_2), and trigrams (p_3) along with Corpus BLEU score of the response suggestion methods on the **English** question-answer pairs.

Model	p_1	p_2	p_3	BLEU
<i>Traditional retrieval</i>				
TF	27.32	17.33	15.34	19.50
TF-IDF	27.89	17.59	15.85	21.16
M-TF	26.47	17.20	15.19	18.84
M-TF-IDF	27.89	17.96	16.26	21.67
<i>Retrieval with KNN</i>				
1-NN	27.40	17.26	15.50	23.09
3-NN	26.64	16.93	15.59	23.71
5-NN	26.92	17.29	15.93	24.00
7-NN	27.02	17.25	15.87	23.79
<i>Retrieval with word vectors</i>				
WCD	28.40	17.70	15.59	20.72
WCD-IDF	28.09	17.35	15.25	20.57
M-WCD	26.76	17.24	15.57	18.55
M-WCD-IDF	28.96	17.77	15.49	19.44
<i>Representation learning</i>				
Dotproduct-n1	13.61	1.55	0.72	8.59
Dotproduct-n3	4.52	0.51	0.04	7.13
Joint-n3	25.34	16.23	14.80	14.07
Joint-n4	25.72	16.73	15.22	16.81
Joint-n5	26.53	16.69	14.83	18.73
<i>Conditioned-generation</i>				
seq2seq	16.93	7.58	4.8	4.79

at 24.00. The representation learning approach *joint* is the strongest competitor for the retrieval-based methods with a modified precision values of 26.53, 16.69, and 14.83, respectively, along with a BLEU score of 18.73.

7 Discussion

The main result is that retrieval models are superior to conditioned-generation approaches in our closed domain case study with limited training data. We hypothesize that this is primarily caused by the lack of training data for the parameter-learning approaches. The *joint* approach becomes competitive to the retrieval methods when tuning the number of negative samples. The *dotproduct* approach did not lead to reasonable results, also not with more negative samples.

A limitation of this work is that we, for now, focussed on comparing the retrieval-based approaches and the representation learning approaches to the basic RNN sequence-to-sequence architecture. Sequence-to-sequence is, however, a general framework, in which many extensions [3, 20, 23, 24] are possible. While our results are a strong indicator that the sequence-to-sequence models lack training data, we cannot exclude that a different configuration of a sequence-to-sequence architecture might yield a higher performance.

Both retrieval models and representation learning approaches yield responses that are also present in the training set. In contrast, the conditioned-generation model generates new sentences and is susceptible to be ungrammatical or semantically questionable. This might be insufficiently addressed by the BLEU score. For this reason, we manually inspected the generated responses. Despite the limited amount of training data, the learned grammar was in most cases appropriate. The generated responses often contained phrases of two to five words that were reminiscent of the training set. As expected, we observe a tendency that the generated responses are rather generic and reflect the most-common responses of the training set. While suggesting such generic responses is in principle not undesirable, the BLEU scores show that the generated response does, in most cases, not reflect what the librarian’s actual response was.

Other works use response normalization [8, 16] for evaluation. The responses of the test set are normalized with respect to the responses of the training set. This would also alleviate the aforementioned issue of *UNK* tokens in conditioned-generation approaches. In our case, however, response normalization would have biased our evaluation. Due to the limited data, more semantically different answers would have been mapped to the same response class. Thus, we preferred the BLEU score for our evaluation.

We considered alternative transformations of chat transcripts into paired training data. One alternative was to solely extract the initial question and answer. This, however, leads to rather uninformative response suggestions such as greeting the user and notifying him/her that his/her question is recognized. Still, in some cases, the initial response did already hold substantial information, such that omitting the first response was also not an option. We also considered aggregating all past utterances within a transcript into a joint context. Whenever an utterance of a librarian is traversed, a pair of context and response is emitted. Both the user’s utterances and past librarian utterances are kept in the context until the end of the transcript. The loss of immediate context, however, harmed the performance of the response suggestion models. As future work, we consider extracting contextualized triples holding the immediate context, the long-term contexts, and the response.

We evaluated the different approaches over two languages of a single real-world dataset. Further studies are necessary to assert whether our results generalize to other datasets. For this dataset, we started from raw data to a training set of question-answer pairs. As future work, we consider taking also the context (the chat history) of each pair into account. We envision that more contextualized predictions may help to mitigate the lack of training data.

8 Conclusion

We have shown that retrieval-based methods yield higher BLEU scores on a response suggestion task than sequence-to-sequence models and representation learning approaches, when the training data is limited. Dedicated representation learning techniques, however, are in some cases competitive to the strong retrieval baselines. More specifically, an architecture trained to learn a joint representation of question and answers using three hidden layers with rectified linear activations did yield competitive scores. Given our results, this approach can be considered the most promising parametric approach for response suggestion in scenarios with limited training data. We presented a text-processing pipeline to extract question-answer pairs from XML data. Such question-answer pairs, regardless of where they come from, can be used to construct a response suggestion model. We make our text-processing pipeline, the response suggestion models, and the evaluation procedure openly available on GitHub⁹. The goal is that other researchers and practitioners may re-use our implementation to verify our results on other datasets or construct question answering systems for a real world applications.¹⁰

Acknowledgements. This research was co-financed by the EU H2020 project MOVING (see footnote 10) under contract no 693092. We thank Nicole Krueger from ZBW for providing the chat transcripts and helpful discussions on requirements and possible applications.

References

1. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., et al.: TensorFlow: large-scale machine learning on heterogeneous distributed systems. CoRR abs/1603.04467 (2016)
2. Al-Rfou, R., Pickett, M., Snaider, J., Sung, Y., Strope, B., Kurzweil, R.: Conversational contextual cues: the case of personalization and history for response ranking. CoRR abs/1606.00372 (2016)
3. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. CoRR abs/1409.0473 (2014)
4. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. TACL **5**, 135–146 (2017)
5. Chen, B., Cherry, C.: A systematic comparison of smoothing techniques for sentence-level BLEU. In: WMT@ACL. The Association for Computer Linguistics (2014)
6. Galke, L., Saleh, A., Scherp, A.: Word embeddings for practical information retrieval. In: GI-Jahrestagung. LNI, vol. P-275. GI (2017)
7. Grave, E., Bojanowski, P., Gupta, P., Joulin, A., Mikolov, T.: Learning word vectors for 157 languages. In: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018) (2018)



⁹ <https://github.com/lgalke/resuggest>.

¹⁰ <http://www.moving-project.eu/>.

8. Henderson, M., et al.: Efficient natural language response suggestion for smart reply. CoRR abs/1705.00652 (2017)
9. Huang, P., He, X., Gao, J., Deng, L., Acero, A., Heck, L.P.: Learning deep structured semantic models for web search using clickthrough data. In: CIKM. ACM (2013)
10. Kannan, A., et al.: Smart reply: Automated response suggestion for email. In: KDD. ACM (2016)
11. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR abs/1412.6980 (2014)
12. Kusner, M.J., Sun, Y., Kolkin, N.I., Weinberger, K.Q.: From word embeddings to document distances. In: ICML. JMLR Workshop and Conference Proceedings, vol. 37. JMLR.org (2015)
13. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
14. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: NIPS (2013)
15. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: 40th Annual meeting of the Association for Computational Linguistics, ACL-2002 (2002)
16. Ritter, A., Cherry, C., Dolan, W.B.: Data-driven response generation in social media. In: EMNLP. ACL (2011)
17. Salton, G., Buckley, C.: Term-weighting approaches in automatic text retrieval. *Inf. Process. Manag.* **24**(5), 513–523 (1988)
18. Srivastava, N., Hinton, G.E., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.* **15**(1), 1929–1958 (2014)
19. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: NIPS (2014)
20. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NIPS (2017)
21. Vinyals, O., Le, Q.V.: A neural conversational model. CoRR abs/1506.05869 (2015)
22. Wu, L., Fisch, A., Chopra, S., Adams, K., Bordes, A., Weston, J.: StarSpace: embed all the things! CoRR abs/1709.03856 (2017)
23. Wu, Y., Wu, W., Yang, D., Xu, C., Li, Z., Zhou, M.: Neural response generation with dynamic vocabularies. CoRR abs/1711.11191 (2017)
24. Xu, Z., et al.: Neural response generation via GAN with an approximate embedding layer. In: EMNLP. Association for Computational Linguistics (2017)



What to Read Next? Challenges and Preliminary Results in Selecting Representative Documents

Tilman Beck¹, Falk Bösch¹(✉) , and Ansgar Scherp² 

¹ Department of Computer Science, Kiel University, 24118 Kiel, Germany
{stu127568,fboe}@informatik.uni-kiel.de

² Computing Science and Mathematics, University of Stirling,
Stirling FK9 4LA, Scotland, UK
ansgar.scherp@stir.ac.uk

Abstract. The vast amount of scientific literature poses a challenge when one is trying to understand a previously unknown topic. Selecting a representative subset of documents that covers most of the desired content can solve this challenge by presenting the user a small subset of documents. We build on existing research on representative subset extraction and apply it in an information retrieval setting. Our document selection process consists of three steps: computation of the document representations, clustering, and selection of documents. We implement and compare two different document representations, two different clustering algorithms, and three different selection methods using a coverage and a redundancy metric. We execute our 36 experiments on two datasets, with 10 sample queries each, from different domains. The results show that there is no clear favorite and that we need to ask the question whether coverage and redundancy are sufficient for evaluating representative subsets.

Keywords: Representative document selection · Document clustering

1 Introduction

As an early-stage researcher or practitioner, delving into a new, previously unknown topic usually starts with issuing broad queries to a search engine. The aim of such an introductory search is to get an overview of the different subtopics, most fundamental works in the field, and the state-of-the-art. Search engines return those documents which best match with the user query and present the results as a ranked list. Such a relevance-based ranking works well for standard information retrieval (IR) tasks, but it is not suited for finding a complete, representative, and comprehensive selection when exploring a new field. Furthermore, the large number of search results makes it very time-consuming for the user to identify the desired documents because similarly ranked documents have usually similar content. Thus, result lists are often highly redundant, especially the top

results (i. e., the first page) that are usually only considered by the user. It is also difficult to estimate at which position of the ranked list one has read about all aspects of a topic [3], i. e., got a representative overview, because the breakpoint differs from one topic to another.

A representative subset can address these challenges. In general, a representative subset is considered as a selection which covers most of the content, contains the least possible amount of redundant information, and is notably smaller in size than the original dataset. Zhang et al. [21] proposed such a method for text documents that uses clustering and a coverage and redundancy based selection to create a representative subset from a set of documents. We have re-implemented the work of Zhang et al. and evaluated it on search results created from queries issued to two datasets of different domains. Based on the results of preliminary experiments, we decided to investigate the following research questions:

- RQ1: What influence does the choice of (a) document representation, (b) clustering algorithm, and (c) selection method have on the coverage and redundancy scores of the representative subset?
- RQ2: Are the evaluation metrics, coverage and redundancy, sufficient to evaluate the representativeness of a document set?

The outline of this paper is: In Sect. 2, we briefly discuss related work followed by an introduction of our approach in Sect. 3. We describe our datasets, metrics, and experiment setup in Sect. 4. Finally, we present our results in Sect. 5 and discuss them before we conclude.

2 Related Work

Zhang et al. [21] propose a selection technique which uses an unsupervised text mining approach to find a representative set of documents from a large corpus. They first cluster the documents using X-Means, an adaption of K-Means which can be used without prior specification of the cluster number k , to identify the different topics in the dataset. From each cluster, they extract the documents which maximize the content coverage and introduce as less redundant content as possible. The size of the result set is determined by the proportional sizes of the clusters. Their framework outperforms a greedy approach, which directly optimizes for coverage, and a top- n method, which selects the best n documents with regards to coverage. The authors conducted a user study with 20 participants that indicated a preference for their selection approach.

The task of generating reading lists [7] is quite similar to selecting representative documents. However, it aims more to propose subjective and expert-based reading lists rather than an objective, representative selection. Jardine and Teufel [8] proposed an adaptation to the PageRank score, called Themed-PageRank (TPR), which has an LDA-inferred topic dimension and a so-called age-tapering component to incorporate the time aspect. They compute the TPR for each document that is returned by an IR system given a specific query. They rank the documents based on TPR and return the top-20 documents as reading

list. Their evaluation using expert-created reading lists showed improvements on previous state-of-the-art models based on PageRank. A recent approach by Zhang et al. [20] generates book reading lists for certain topics. They use data obtained from social media to train vector representations for each topic-book pair using content from social media for relevance, quality, timeliness, and diversity.

We base our experiments on the work by Zhang et al. [21]. However, they compared their selection to methods which optimize for coverage only, but not for redundancy, which limits the comparability. Furthermore, only X-Means was considered for clustering and no information is given about the actual number of documents retrieved from the datasets, which is an important factor when searching for a small representative document set. Thus, we extend in this paper on the work of Zhang et al. and compare different clustering algorithms, selection methods, and document representations while taking inspiration from the literature.

Please note that we decided to deliberately exclude approaches for result list re-ranking since they are not well comparable to a representative subset, due to the unknown breakpoint [3], as discussed in the introduction. Furthermore, we exclude approaches from the related area of text summarization [13] since they work on a different granularity level.

3 Document Selection

Our approach for document selection is based on Zhang et al. [21] and consists of three steps (see Fig. 1). First, we retrieve the documents that match a certain query and compute the representations of the documents in the result set. Second, we cluster the documents into topics. Third, we select documents from the clusters to form a representative subset. Thus, we extend Zhang by an initial retrieval step to address the retrieval setting. Below, we introduce for each of the three steps the different methods that we compare in our experiments.

3.1 Document Representation

We use two different document representations in our experiments, Bag-of-Words and Paragraph Vectors.

Bag-of-Words: The classical Bag-of-Words (BOW) model represents each document as a vector that contains the weighted term counts for every term of the dataset. Whissell et al. [19] have investigated the effect of different feature weighting approaches on the document clustering performance. They concluded that BM25 outperforms other feature weighting approaches and suggested to use BM25 for clustering tasks.

Paragraph Vectors: Paragraph Vectors, which were introduced by Le and Mikolov [9], are dense vector representations of text fragments of arbitrary length, i. e., paragraphs, or documents, in a significantly lower-dimensional space

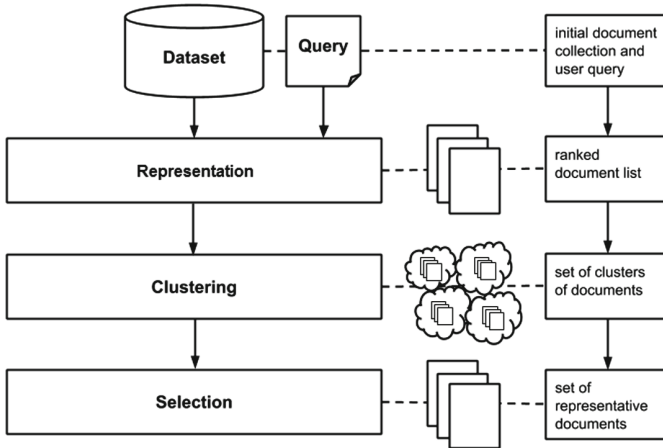


Fig. 1. Our document selection process with its representation, clustering, and selection step in a retrieval setting

than the corpus’ dictionary. They are generated by neural networks that are trained to predict the words surrounding a given word. Those vectors are able to carry semantic information, thus making them often superior to BOW models which ignore word ordering. This process is called doc2vec (D2V).

3.2 Clustering

We use spherical k-Means and Latent Dirichlet Allocation to cluster the documents into topics. Please note that we do not use X-Means, which was used by Zhang et al. [21], since it is too slow, as Zhang et al. stated themselves in a more recent work [22].

Spherical K-Means: K-Means [11] (KM) is a centroid-based clustering algorithm that iteratively assigns each datapoint to the nearest centroid and then recomputes the centroids from the assigned points until it converges. For a larger number of dimensions, when using the Euclidean Distance, the curse of dimensionality leads to uniform distances between data points [1], limiting the applicability. However, the distance metric can be exchanged with cosine similarity without violating the Gaussian distribution assumption underlying K-Means. This spherical K-Means [5] is more suitable for the application on textual data and thus is used in our experiments.

Latent Dirichlet Allocation: Latent Dirichlet Allocation (LDA) [4] is a probabilistic, generative model which identifies hidden topics in a corpus of documents. The basic idea of LDA is that documents are represented as a mixture of topics and each topic is identified by a distribution over words. Given a document corpus, LDA infers a model that is likely to have generated that corpus. LDA takes the term-document count matrix as input and creates a $n \times k$ document-topic

matrix W with n documents and k topics. An entry $W_{i,j}$ describes the probability that topic j is contained in document i . We can use these probabilities to cluster the documents into topic clusters. To adhere to our goal of clustering the documents into topics and then selecting representatives per topic, we decided to keep it simple and cluster the documents by assigning them to the topic for which they have the highest probability.

3.3 Selection

We consider two selection strategies and a baseline. The number of selected documents results from the cluster proportions as proposed by Zhang et al. [21]. This means that we select twice as many documents from a cluster if it is twice as big as another cluster. Thus, the smallest cluster, and the number of documents selected from it, determines the size of the result set.

Selection by Coverage and Redundancy: Zhang et al. [21] proposed a coverage and redundancy based selection (CR). First, from each cluster, the document that is closest to the centroid is selected since it has the highest coverage inside the cluster. Subsequently, the documents with the lowest similarity to the previously selected documents are extracted to minimize redundant content inside the selected set. For all clustering algorithms, except LDA-based clustering, cosine similarity is used to compute the similarity between documents. In the case of LDA, the Jensen-Shannon divergence [10] is used because it is more suitable for probability distributions.

Selection by User Intent: The Intent Aware selection [2] (IA) of documents is based on the relevance of the documents to the query and the probability to satisfy any of the k topics. It maximizes the marginal utility, which is the sum over all topics of the product of the retrieval score and the conditional probability that the so far selected documents failed to represent that topic. The overall goal is to increase the diversity of topics among the selected documents. The original Intent Aware method does not use clustering and selects documents based on their probabilities to satisfy the query and the topics. Thus, theoretically, it can happen that certain clusters are not considered at all. We address this by taking at least one document per cluster (topic).

Random Selection: To evaluate the usefulness of the previously described selection strategies, a random selection strategy (R) is introduced as a baseline. From each cluster, based on their proportions, documents are chosen uniformly at random into the representative subset.

4 Evaluation

4.1 Datasets

We use two datasets of scientific publications for our experiments. We have sampled ten queries for each dataset from corresponding/suitable thesauri for our retrieval setting that return at least 1,000 documents (see Table 1).

ACL Anthology Network: The ACL Anthology Network [15–17] dataset is a collection of research papers of different Association for Computational Linguistics (ACL) venues. We removed all documents that did not have a full-text, leading to a dataset consisting of 22,486 English full-text papers. The query terms were selected from the ACM CCS¹ since the ACL thesaurus is still under construction. On average, the queries return 1,500 documents.

PubMed Open Access: PubMed Central² is a free full-text archive of biomedical and life sciences literature maintained by the United States National Institutes of Health’s National Library of Medicine. From the 4.3 million publications available, about 1.5 million have an open-access license. We were able to acquire the full-text of 646,513 English documents from them. The queries for the PubMed dataset were sampled from the Medical Subject Headings³, a hierarchically-organized medical vocabulary, each yielding on average 1,100 results.

Table 1. The queries and their corresponding number of relevant documents for both datasets. Documents are relevant if they were returned by the IR system.

ACL		PubMed	
Query terms	# rel. docs	Query terms	# rel. docs
Cognitive science	2,066	Dermatologists	1,227
Supervised learning	2,035	Cancellous bone	1,171
Similarity measures	1,639	Meniscus	1,164
Bootstrapping	1,590	Gastroenterologists	1,147
Dynamic programming	1,497	Radiation oncologists	1,084
Maximum entropy modeling	1,452	Endocrinologists	1,075
Natural language generation	1,441	Orthopedic surgeons	1,073
Feature selection	1,317	Surgical wound	1,048
Neural networks	1,135	Nephrologists	1,017
Machine learning approaches	1,069	Tartrate-resistant acid phosphatase	1,002

4.2 Metrics

To ensure comparability, we evaluate our approach using the metrics by Ma et al. [12] that were used by Zhang et al. [21]. Please note that $\text{sim}()$ refers to the cosine similarity, as it was chosen by Zhang et al. [21].

Coverage: Coverage evaluates how much content of a dataset D is covered by a subset S :

$$\text{coverage}(S, D) = \frac{1}{|D|} \sum_{r \in D} (\max_{d \in S} (\text{sim}(d, r))) \quad (1)$$

¹ <http://www.acm.org/about/class/class/2012>.

² <https://www.ncbi.nlm.nih.gov/pmc/>.

³ <https://www.nlm.nih.gov/mesh/>.

In the case that all documents are selected, the coverage reaches its maximal value of 1. The coverage will be close to zero if the selected set of documents only resembles a minimal fraction of the complete set of documents.

Redundancy: The redundant information in a subset S is assessed by:

$$\text{redundancy}(S) = \sum_{d_i \in S} \left(\frac{1 - \frac{1}{|S|} \sum_{d_j \in S} \text{sim}(d_i, d_j)}{|S|} \right) \quad (2)$$

Please note that this computation also considers the size of the subset, i.e., having a subset of three duplicates and a subset of five duplicates would yield different scores. We are also aware that the metric has some shortcomings when used with cosine similarity. However, for the sake of comparability, we use it as it was used by Zhang et al. [21].

4.3 Experiment Setup

We indexed each dataset using the full-text of the documents and used BM25 ($k1 = 1.2$ and $b = 0.75$) as our scoring function for retrieval in Elasticsearch⁴. We retrieved the documents for each query by using exact matching and pre-processed the resulting document set using Porter stemming [14] and stop-word removal using NLTK stop-words before computing the document representation (e.g., BoW or Paragraph Vectors). To address the curse of dimensionality, all terms that appeared in more than 95% of the documents or in less than two documents were removed. We further limited our vocabulary to the remaining 50,000 most popular terms.

We calculated the document representation of the preprocessed documents using the methods described in Sect. 3.1. In case of the BoW model, we used BM25 with the parameters $k1 = 20$ and $b = 1$ based on a study of Whissell et al. [19]. For the paragraph vector model, we used a model that was trained on a dump of all English Wikipedia articles from December 2017 using the gensim library [18].

Under the assumption that the topical diversity is limited, we decided to cluster the documents with $k \in \{5, 10, 25, 50\}$ since the true number of clusters is unknown.

After clustering, representative documents were selected from the clusters using the proposed selection methods. For the calculation of the selected set S for the IA selection in combination with LDA, we follow the procedure described in [6]. In the case of k-Means clustering (for both BOW and D2V), we take a different approach as the necessary probability distributions are not provided by the clustering algorithm. We compute the quality value using the retrieval score, weighted by the cosine distance of a document to its corresponding cluster center. For the calculation of the conditional probability, first, the feature vector for the query is derived from the vocabulary (or pretrained model in case of paragraph

⁴ <https://www.elastic.co/>.

vectors). Then, for each topic, the probability is the cosine distance between the topic and the query. To compute the similarity between two documents, we use the cosine similarity except for LDA-based clustering where the Jensen-Shannon divergence is used.

To allow for a fair comparison between the different document selection strategies, we compute the metrics using the BM25-weighted BoW model and cosine similarity (or distance, respectively) even if the clustering was using different feature vectors (e.g., LDA-based clustering).

In total, we ran 36 experiments, each using a different combination of the 2 document representations, 2 clustering algorithms, 4 different values for k , and 3 different selection methods, on 20 different document sets, which were returned by 10 queries on our two datasets. We repeated each experiment 5 times and averaged over all runs.

5 Results and Discussion

Figures 2 and 3 show the average coverage and redundancy values for the different experiment configurations on the ACL and PubMed datasets, respectively. We analyze the results along the three components of RQ1: (a) document representation, (b) clustering, and (c) selection. To answer RQ1 (a), we look at the results for K-Means using the bag-of-words model (KM-BOW) and K-Means using document embeddings (KM-D2V). On both datasets, for $k = 5$ and $k = 10$, the coverage has no large difference but for the selection methods IA and R with document embeddings there is slightly less redundancy. Starting from $k = 25$, selections based on KM-D2V have a higher coverage and a sharper increase in redundancy. The use of D2V most likely influences the representative subset selection so that for small k , slightly less redundant content is selected. For larger k , clearly, more content is covered while the increase in redundancy is neglectable.

To answer RQ1 (b), we compare the results of all clustering algorithms. Except for LDA, the coverage and redundancy results for the strategies increase steadily with larger k , all achieving their maximum at $k = 50$. For LDA, both scores are close to 1 when increasing to $k = 25$ and above. The differences between the algorithms are more distinct at a larger k .

We compare the coverage and redundancy for the selection methods to answer RQ1 (c). One can see that the CR selection generates lower redundancy scores in combination with KM-BOW clustering but the effect is diminishing with larger k . For the other clustering algorithms, one can observe that CR has equal or higher redundancy scores than the other selection methods. In terms of coverage, the choice of the selection method is less important than the clustering algorithm as the differences between CR, IA, and R are minimal.

Summarizing our results regarding RQ1, on both datasets, the best performing configurations, with respect to coverage, are those that use D2V or LDA. However, the selections based on BOW have the lowest redundancy scores.

Regarding RQ2, we make three observations with respect to coverage and redundancy. First, the scores for both metrics increase consistently for a larger

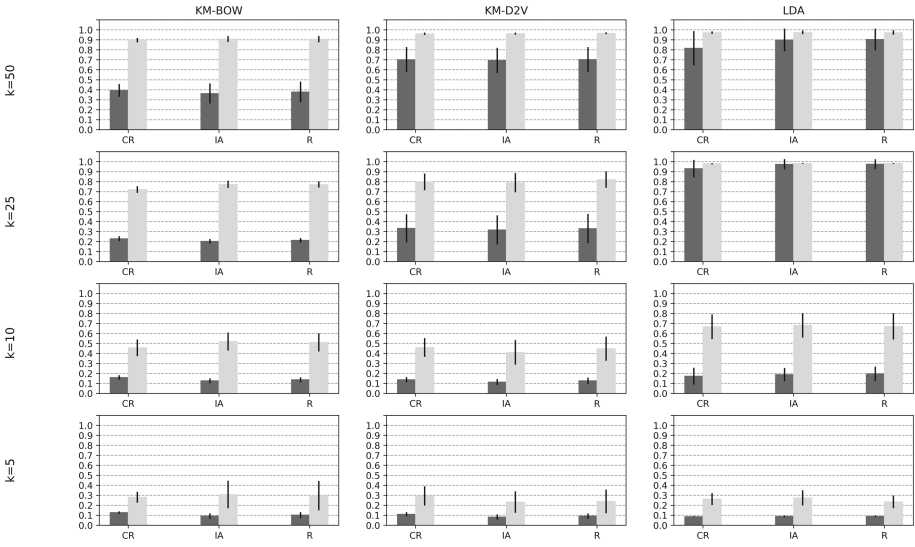


Fig. 2. Coverage (left black bars) and redundancy (right grey bars) averaged over all queries for the different document selection strategies on the ACL dataset using $k \in \{5, 10, 15, 20\}$. The standard deviation is indicated as a black line on top of each bar.

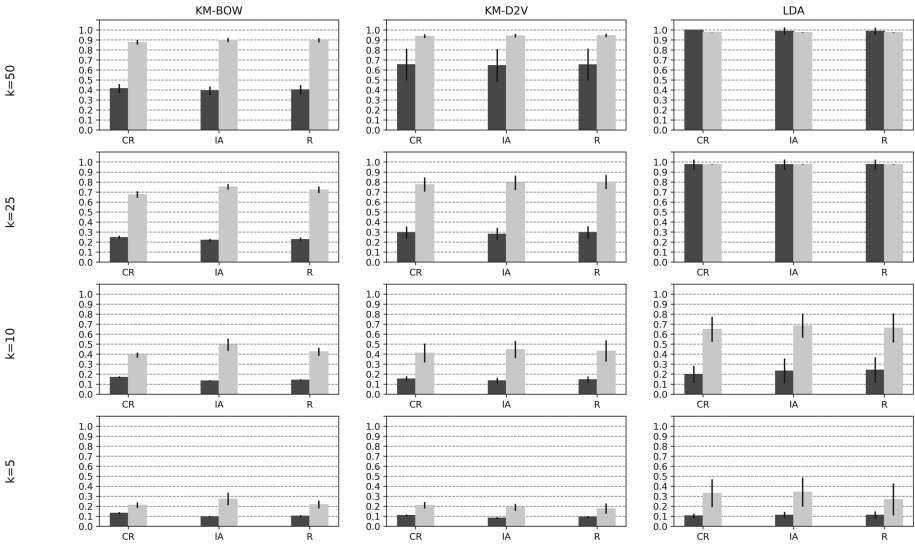


Fig. 3. Coverage (left black bars) and redundancy (right grey bars) averaged over all queries for the different document selection strategies on the PubMed dataset using $k \in \{5, 10, 15, 20\}$. The standard deviation is indicated as a black line on top of each bar.

number of clusters. This correlates directly with the number of documents due to the cluster proportion calculation, which defines the number of documents that are selected. In the case of more heterogeneous cluster sizes, more documents will be selected from each cluster. With larger k , it is more likely that the documents are unevenly distributed among the clusters. One example is the LDA-based selection for $k \in \{25, 50\}$, which contains most documents and has coverage scores close to 1. Thus, coverage and redundancy are inflated by selecting most of the documents rather than being a result of a better selection.

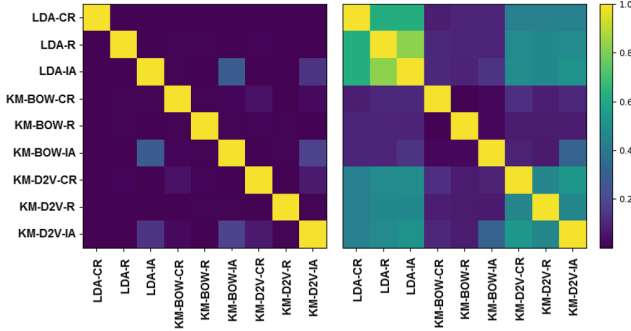


Fig. 4. Average fractions of shared documents in the representative document sets, selected by our nine different document selection strategies on the ACL dataset. On the left for $k = 5$ and on the right for $k = 50$.

Second, from the comparison with a random selection method, we observed an independence of the evaluation metrics from the actual choice of documents. This raises the question whether the selection methods select similar document sets. Therefore, we decided to have a closer look at the subsets. Figure 4 shows the result of the comparison of the subsets for $k = 5$ and $k = 50$ at the example of ACL, i. e., the fraction of mutual documents between the selection strategies. We can see that for small k none of the selections share many similar documents, while for a larger k strategies with the same document representation and clustering algorithm (but different selection strategies) start to select similar documents. However, these document selections are more alike as more documents are selected in general. This becomes obvious, especially for D2V and LDA, since both are more susceptible to imbalanced cluster sizes. This limits the generalizability of coverage and redundancy to evaluate the representativeness of a document subset. Please note, we have omitted the analysis on the PubMed dataset since the results were similar.

Finally, in contrast to the original work of Zhang et al. [21], we observe for each strategy that the redundancy exceeds the coverage scores. We have investigated whether it results from our IR setting and hence a general higher similarity of documents. However, we achieved similar results when using an equal amount of documents randomly sampled from the full dataset. Further

research needs to be conducted to explain the difference between the results on our and Zhangs' datasets.

6 Conclusion

We have proposed a document selection framework in an information retrieval context as an extension of the representative subset selection by Zhang et al. [21]. Our analysis reveals that there is no unique representative document set with regards to the evaluation metrics but instead most strategies achieve comparable results with different document subsets, even our random baseline. This raises the question whether coverage and redundancy are sufficient to evaluate the representativeness of a document set. Furthermore, we identified the size of the result set as problematic. It is often too large for a representative subset due to the selection based on the cluster proportions. Therefore, as future work, we propose to enhance the representativeness metric introduced by Ma et al. [12] with a weighting term which promotes those solutions which select fewer documents for evaluating representative subsets. Finally, we plan to further investigate the influence of different dataset characteristics and preprocessing methods on the overall document selection process.

Acknowledgment. This research was co-financed by the EU H2020 project MOVING (<http://www.moving-project.eu/>) under contract no 693092 and the EU project DigitalChampions_SH.

References




1. Aggarwal, C.C., Hinneburg, A., Keim, D.A.: On the surprising behavior of distance metrics in high dimensional space. In: Van den Bussche, J., Vianu, V. (eds.) ICDT 2001. LNCS, vol. 1973, pp. 420–434. Springer, Heidelberg (2001). https://doi.org/10.1007/3-540-44503-X_27
2. Agrawal, R., Gollapudi, S., Halverson, A., Ieong, S.: Diversifying search results. In: Baeza-Yates, R.A., Boldi, P., Ribeiro-Neto, B.A., Cambazoglu, B.B. (eds.) Proceedings of the Second International Conference on Web Search and Web Data Mining, WSDM 2009, Barcelona, Spain, 9–11 February 2009, pp. 5–14. ACM (2009). <https://doi.org/10.1145/1498759.1498766>
3. Arampatzis, A., Kamps, J., Robertson, S.: Where to stop reading a ranked list?: threshold optimization using truncated score distributions. In: Allan, J., Aslam, J.A., Sanderson, M., Zhai, C., Zobel, J. (eds.) Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, 19–23 July 2009, pp. 524–531. ACM (2009). <https://doi.org/10.1145/1571941.1572031>
4. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent Dirichlet allocation. In: Dietterich, T.G., Becker, S., Ghahramani, Z. (eds.) Advances in Neural Information Processing Systems 14 (Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, Vancouver, British Columbia, Canada, 3–8 December 2001), pp. 601–608. MIT Press (2001). <http://papers.nips.cc/paper/2070-latent-dirichlet-allocation>

5. Endo, Y., Miyamoto, S.: Spherical k -means++ clustering. In: Torra, V., Narukawa, Y. (eds.) MDAI 2015. LNCS (LNAI), vol. 9321, pp. 103–114. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-23240-9_9
6. He, J., Meij, E., de Rijke, M.: Result diversification based on query-specific cluster ranking. *JASIST* **62**(3), 550–571 (2011). <https://doi.org/10.1002/asi.21468>
7. Jardine, J.G.: Automatically generating reading lists. Ph.D. thesis, University of Cambridge, UK (2014). <http://ethos.bl.uk/OrderDetails.do?uin=uk.bl.ethos.648722>
8. Jardine, J.G., Teufel, S.: Topical PageRank: a model of scientific expertise for bibliographic search. In: Bouma, G., Parmentier, Y. (eds.) Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2014, Gothenburg, Sweden, 26–30 April 2014, pp. 501–510. The Association for Computer Linguistics (2014). <http://aclweb.org/anthology/E/E14/E14-1053.pdf>
9. Le, Q.V., Mikolov, T.: Distributed representations of sentences and documents. In: Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21–26 June 2014. JMLR Workshop and Conference Proceedings, vol. 32, pp. 1188–1196. JMLR.org (2014). <http://jmlr.org/proceedings/papers/v32/le14.html>
10. Lin, J.: Divergence measures based on the Shannon entropy. *IEEE Trans. Inf. Theory* **37**(1), 145–151 (1991). <https://doi.org/10.1109/18.61115>
11. Lloyd, S.P.: Least squares quantization in PCM. *IEEE Trans. Inf. Theory* **28**(2), 129–136 (1982). <https://doi.org/10.1109/TIT.1982.1056489>
12. Ma, B., Wei, Q., Chen, G.: A combined measure for representative information retrieval in enterprise information systems. *J. Enterp. Inf. Manag.* **24**(4), 310–321 (2011). <https://doi.org/10.1108/17410391111148567>
13. Naveen, G.K.R., Nedungadi, P.: Query-based multi-document summarization by clustering of documents. In: Proceedings of the 2014 International Conference on Interdisciplinary Advances in Applied Computing, ICONIAAC 2014, pp. 58:1–58:8. ACM, New York (2014). <https://doi.org/10.1145/2660859.2660972>
14. Porter, M.F.: An algorithm for suffix stripping. *Program* **14**(3), 130–137 (1980). <https://doi.org/10.1108/eb046814>
15. Radev, D.R., Joseph, M.T., Gibson, B., Muthukrishnan, P.: A bibliometric and network analysis of the field of computational linguistics. *J. Am. Soc. Inf. Sci. Technol.* (2009)
16. Radev, D.R., Muthukrishnan, P., Qazvinian, V.: The ACL anthology network corpus. In: Proceedings, ACL Workshop on Natural Language Processing and Information Retrieval for Digital Libraries, Singapore (2009)
17. Radev, D.R., Muthukrishnan, P., Qazvinian, V., Abu-Jbara, A.: The ACL anthology network corpus. *Lang. Res. Eval.* **47**, 1–26 (2013). <https://doi.org/10.1007/s10579-012-9211-2>
18. Řehůřek, R., Sojka, P.: Software framework for topic modelling with large corpora. In: Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks, pp. 45–50. ELRA, Valletta, May 2010. <http://is.muni.cz/publication/884893/en>
19. Whissell, J.S., Clarke, C.L.A.: Improving document clustering using Okapi BM25 feature weighting. *Inf. Retr.* **14**(5), 466–487 (2011). <https://doi.org/10.1007/s10791-011-9163-y>

20. Zhang, B., Yin, X., Zhou, F., Jin, J.: Building your own reading list anytime via embedding relevance, quality, timeliness and diversity. In: Kando, N., Sakai, T., Joho, H., Li, H., de Vries, A.P., White, R.W. (eds.) Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, Shinjuku, Tokyo, Japan, 7–11 August 2017, pp. 1109–1112. ACM (2017). <https://doi.org/10.1145/3077136.3080734>
21. Zhang, J., Liu, G., Ren, M.: Finding a representative subset from large-scale documents. *J. Informetr.* **10**(3), 762–775 (2016). <https://doi.org/10.1016/j.joi.2016.05.003>
22. Zhang, J., Ren, M., Xiao, X., Zhang, J.: Providing consumers with a representative subset from online reviews. *Online Inf. Rev.* **41**(6), 877–899 (2017). <https://doi.org/10.1108/OIR-05-2016-0125>



Text-Based Annotation of Scientific Images Using Wikimedia Categories

Frieda Josi , Christian Wartena^(✉) , and Jean Charbonnier 

University of Applied Sciences and Arts Hanover, Expo Plaza 12,
30539 Hanover, Germany
christian.wartena@hs-hannover.de

Abstract. The reuse of scientific raw data is a key demand of Open Science. In the project NOA we foster reuse of scientific images by collecting and uploading them to Wikimedia Commons. In this paper we present a text-based annotation method that proposes Wikipedia categories for open access images. The assigned categories can be used for image retrieval or to upload images to Wikimedia Commons. The annotation basically consists of two phases: extracting salient keywords and mapping these keywords to categories. The results are evaluated on a small record of open access images that were manually annotated.

Keywords: Scientific image search · Text annotation
Wikipedia categories

1 Introduction

In order to increase the reuse of scientific images from open access journals, we collect scientific images, make them available in a search engine and upload high quality images to Wikimedia Commons. A beta version of the NOA scientific image search, using the categories extracted by the proposed method is available under: <http://noa.wp.hs-hannover.de> [1].

Metadata such as author and disciplines can be adopted from the papers the images are taken from. However, publishers do not provide no specific metadata for the individual images. In this paper we present a method for extracting detailed categories for each image from its caption and from text fragments referring to the image.

After discussing related work, we will present the data we have used (Sect. 3) and a method that is based on extracting keywords from the captions and related text and assigning categories on the base of the keywords, for which categories are known (Sect. 4). In Sects. 5 and 6 we present an evaluation carried out on the basis of 100 images from open access journals that were uploaded to Wikimedia Commons and annotated manually.

2 Related Work

Relevant work for using Wikipedia titles and categories focuses on linking to the Wikipedia articles [2] or using article titles for indexing images [3]. The categories of Wikipedia articles were used much less frequently. An example of the use of the Wikipedia category system for annotation is the work by Wartena and Brussee [4].

The extraction of key terms is a well studied area with numerous publications. Popular algorithms are those described by Frank et al. [5], Turney [6] and by Mihalcea and Tarau [7]. Leong et al. [8] explicitly use keyword extraction to describe images. The method described by Frank et al. [5] and Turney [6] uses various features to determine the suitability of a word as keyword. The most important feature still is the inverse document frequency (idf) that was already proposed by Salton [9]. Besides idf we use the distributional similarity of a keyword with the entire text. This method was proposed by [10].

The matching of the extracted terms to the Wikipedia article title is described in Mihalcea and Csomai [2]. Classification of text based on the classification of key terms found in the text was e.g. done by Wartena and Sommer [11].

3 Data Records and Wikipedia Categories

For the development of the annotation method, 397 data records were used. Each data record contains the caption and the sentences referring to the image. These images have been published in open access journals by Copernicus, Hindawi, Frontiers and Springer Open. We use the XML markup provided by the publishers to identify references to each image. We use the whole sentence containing the image reference as a context for the image. The image captions used have an average length of 308 words and 1881 characters. Table 1 shows an overview of the number of words in captions and the complete data record, include sentences referring to the image¹.

Table 1. Number of words in caption and referring sentences in the development, the evaluation data record and in the entire database (application).

Data record	Text	Size	Average	Min	Max
Develop	Caption	397	54	2	503
	Caption+ref. sent		308	13	2274
Evaluation	Caption	100	46	3	404
	Caption+ref. sent		326	10	2938
Application	Caption	2,9M	81	0	5268
	Caption+ref. sent		405	0	43817

¹ For an example of an extremely long capture see Fig. 5 in <http://dx.doi.org/10.1002/ece3.2579>. Also some parsing errors resulted in long captions.

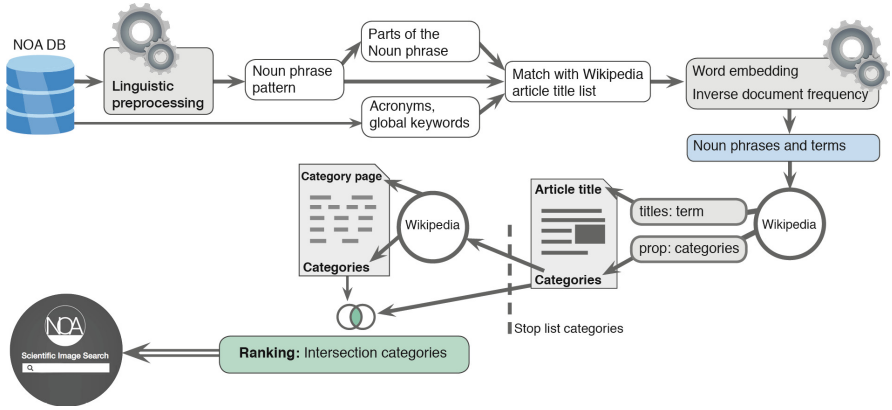


Fig. 1. Text-based annotation model with Wikipedia categories

The whole corpus of the NOA image search engine contains 870,840 articles with images. In total, there are 2.9M scientific images from the subjects medicine, science, health sciences, biology, technology, chemistry and more. We used the whole corpus to compute the idf-values of all terms.

For evaluation we collected 100 images that were not part of the development set and that seemed to be more or less interesting for the Wikimedia community. i.e. we excluded charts, microscopy images, etc.

For the categorization of the images we used the categories of Wikipedia. The category system in the Wikimedia Foundation projects consists of categories created by volunteers as well as categories from existing norm data. In this way, the norm data of LCCN, and VIAF can be included in the articles of Wikipedia [12]. The article pages in Wikipedia are organized using categories. The categories include theme categories, object categories, structural categories and metacategories [13]. A page can be assigned to several categories. It has to be noted, that Wikipedia categories can be used in Wikimedia Commons, but in principle Wikimedia Commons and the Wikipedia of each language has its own independent category system.

4 Annotating Scientific Images

Our text-based annotation method consists of two phases: extracting Wikipedia terms and assigning categories². The process flow is shown schematically in more detail in Fig. 1 and described in the following sections.

² Our source code will be released together with all developed source codes of the NOA project.

4.1 Term and Noun Phrases Extraction

The linguistic preprocessing (tokenization and part of speech tagging and lemmatization) is carried out with the open source Natural Language Toolkit (NLTK) [14] using the Wordnet lemmatizer for lemmatization.

In order to find words and phrases that are used as the title of a Wikipedia article we search noun phrases according to a simple regular expression over POS tags. Using the syntax for chunking grammars described in [14] we define a noun phrase as:

$$\text{NP} : (<\text{CD}>)?(<\text{JJ}>)* <\text{N}(\text{N}|\text{P}).*> + \quad (1)$$

using POS tags from the Penn Treebank Tagset [15]. Thus CD stands for a cardinal number, JJ for adjective and all tags starting with NN or NP for various types of common nouns and proper nouns.

Next we lookup each noun phrase found in Wikipedia (either using the Wikipedia API in the development phase or the SQL-Dump in the application to a larger data set). If the phrase is not found we split the phrase into the first word and the remaining tail. If the first word is a noun, it is looked up in Wikipedia. The tail also is looked up and recursively split until the phrase is found in Wikipedia or no words remain. Thus it is ensured, that only the longer (and more specific) phrase is used if it is a title in Wikipedia, but that the smaller phrases are still used if the longer phrase is not found. E.g., if we find *Greenhouse gas* we don't use the term *gas*, since the whole phrase is found in Wikipedia.

In order to match phrases to Wikipedia titles, we exclude words from a common list of stopwords and we pluralize words if the singular form was not found. All names of albums, magazines, musical groups and films are excluded from matching.

We extract key phrases in this way from the caption, but also from each sentence with a reference to the image. In addition we take all global keywords from the paper (provided by the authors) if they are found in Wikipedia and all expanded acronyms (see [16] for details on acronym resolution in NOA), if they could be expanded automatically and if the complete expansion was found in Wikipedia.

4.2 Term Ranking

After we have found phrases in the text we have to rank them and select the most promising ones. In absence of suitable annotated data that can be used for training, we use only two features: idf and the similarity of the word embedding of the key phrase with the word embedding of the caption.

The inverse document frequency is calculated with:

$$\text{idf}_i = \log \frac{\text{Number of data records in the corpus}}{\text{Number of data records containing NP}} \quad (2)$$

Table 2. Example of Wikipedia terms found for an image, with source, idf-value and distance of context vector to the caption for each term. Source: r = Referring Context, c = Caption.

Wikipedia Terms	source	idf	cos	Wikipedia Terms	source	idf	cos
axillary fascia	r	20.0	0.72	inch	r	10.9	0.33
griffith university	r	18.1	0.20	upper limb	c	10.8	0.65
brachial fascia	c	17.5	0.77	humerus	r	10.4	0.62
quartus	r	15.7	0.35	continuation	r	10.2	0.24
medical literature	r	14.4	0.26	fascia	c	10.0	0.75
common name	r	13.9	0.26	nomenclature	r	9.4	0.15
deep fascia	r	13.9	0.75	depiction	r	9.4	0.23
epicondyle	r	12.7	0.76	rib	r	9.3	0.59
joint capsule	r	12.4	0.58	informed consent	r	9.3	0.59
queensland	r	12.2	0.16	wood	r	9.2	0.24
cadaver	r	11.4	0.40	septum	r	9.1	0.56
axilla	r	11.3	0.56	thorax	c	9.1	0.58
biceps	r	11.1	0.69
tubercle	r	10.9	0.57	number	r	2.3	0.19

Image

In many cases idf alone is not sufficient. In the caption or in the referring context highly specific words can occur that are not related to the image at all. Also the global keywords from the paper might be very good, but sometimes too general for the picture we want to describe. Thus we want to know, how well each term fits to the captions as a whole.

We computed word embeddings for all words in the corpus that occur at least 5 times. We trained an word2vec model using our corpus with a window size of 5 using the CBOW model, an embedding size of 300 and a minimum word occurrence threshold of 5. We removed all tokens from our data that are in the NLTK stopword list or that have less than two characters.

Now, for each word we have an abstract vector of 300 dimensions. We represent both, a key phrase and the caption, by the average vector of all words (excluding stop words) that they consist of. Now the cosine between the vector of the key phrase and the vector of the abstract is a measure for the degree to which the key phrase is representative for the caption.

Table 2 shows the extracted noun phrases and terms for the first image from the article with the DOI: [10.1155/2016/5402081](https://doi.org/10.1155/2016/5402081). Also the idf and the cosine

between the vectors for the phrase and the caption are given. Here we see, that e.g. *Griffith University* has a very high idf value, but is not very representative for the caption. Here the cosine between the term vector and the caption vector is a much better indicator for relevance.

We will evaluate three variants, as shown in Table 3. In the first variant we use the top five phrases according to idf, in the second variant we use five phrases with the highest cosine similarity and finally we combine both criteria, by using five phrases with the highest cosine similarity taken from the 15 most specific phrases.

The five keywords for our example selected by the third variant are set in bold face in Table 2.

Table 3. Variants of the evaluation

Variant 1: Idf
Variant 2: Cosine
Variant 3: Idf + Cosine

4.3 Category Filtering

We want to assign categories to the images, not keywords (or article titles). The categories assigned to articles corresponding to the best keywords for each image are candidates, but we use also the upper categories of each category as a candidate.

Before selecting the most relevant categories, we remove a large number of categories that are not interesting for our purpose such as *Category:Systems* or categories with meta information for Wikipedia internal purposes, like categories for articles missing references etc. To do so, we filtered out all categories that are classified as hidden category in Wikipedia and all categories that are classified as container category. Furthermore we use a list of regular expression for category names that are filtered out, e.g. all categories that contain the word *Wikipedia* or *stub* or *disambiguation*. Finally, a stop list of further categories is used, containing categories like *Category:Nothing*, *Category:Self* or also meta information like e.g. *Category:ISO_basic_Latin_letters*.

4.4 Category Ranking

We assume that the categories of the articles themselves are more likely to be appropriate than their upper categories and we assume that a category that is the category of two keywords is better than a category that comes only with one keyword. To formalize this intuition, we define the number of keywords associated with a category c at different levels. We say a keyword k is associated at level 0 with c if c is a category of (the Wikipedia article with title) k and c is identical with k . E.g. the article *Fascia* has the category *Fascia*, so the category

Table 4. Ranking of categories for the example in Table 2

Category	Value	Category	Value
Fascia	3.0	Limbs (anatomy)	0.4
Muscular system	1.6	Muscles by action and location	0.4
Musculoskeletal system	1.6	Joints	0.4
Soft tissue	1.2	Elbow	0.4
Connective tissue	1.2	Forearm	0.4
Tissues (biology)	1.2	Muscles by location	0.4
Elbow flexors	1.0	Flexors	0.4
Forearm supinators	1.0	Upper limb anatomy	0.4
Muscles of the upper limb	1.0	Muscles by action	0.4
Shoulder flexors	1.0	Shoulder	0.4
Skeletal system	1.0	Organ systems	0.4
Medical Subject Headings	0.8	Dance science	0.4

Fascia is associated at level 0 with the keyword *fascia*. We say c is associated at level 1 with k , if c is a category of k but not identical to k ; c is associated at level l with k , if it has a subcategory that is associated with k at level $l - 1$ (and c is not associated with k at level $l - 1$). We denote the number of keywords associated with c at level l as $r_l(c)$. Finally, we define the weight $w(c)$ as:

$$w(c) = \sum_{l=0}^n w_n \cdot r_l(c) \quad (3)$$

Where $n \in \mathbb{N}$ and w_n are the level weights. In absence of training data to determine optimal weights, in the following we let $n = 2$ and $w_0 = 1.2$, $w_1 = 1.0$ and $w_2 = 0.4$. For our example this ranking gives the weights shown in Table 4.

It might seem obvious to take the number of relations between the categories as a feature as well, as was done e.g. by [17]. Our experiments indicated however, that this results in a massive preference for categories from areas that are worked out very well. In most cases these are unspecific general areas and high weights for categories with many connections to other found categories therefore suppress specific and precise categories. On the other hand, as we will see below, categories introduced by several keywords usually are quite adequate.

5 Evaluation

The images for the evaluation were uploaded to Wikimedia Commons and manually annotated with categories.³ The selection criteria for the upload of the

³ The images are available on Wikimedia Commons at the following link: <https://commons.wikimedia.org/w/index.php?title=Special:ListFiles/Sohmen&ilshowall=1>.

Table 5. Examples of semantically related categories used for the semantic evaluation

Commons category	Wikipedia category
Molecular biology	Molecular modeling
Temperature comparisons	Thermodynamics
Cochlear implants	Hearing
Robotics	Robots
Infectious disease control	Infectious diseases

images to Wikimedia were a higher probability of reuse, images from current papers, graphics and photos but no schematics and only images with the copyright license cc-by-2.5, cc-by-3.0 or cc-by-4.0. The categories were assigned by project members and also by other Wikipedia users. The image, the caption and the title of the paper were used to get information about the image and to select suitable categories. Only existing Wikimedia Commons categories were used. In total the images have received 264 categories.

Since the gold standard now is annotated with categories from Wikimedia Commons, while our method predicts categories from Wikipedia, the evaluation was not automated but done manually in order to allow for slight differences in the names of the categories like *soil* (Commons) and *soils* (Wikipedia) or *heart* and *heart (organ)*. The scope of literal consistency includes the singular and plural form of a category [18] and addition of scope notes.

Even when we allow for these small differences, we are too strict: if the gold standard has the category *robotics* and the algorithm proposes *robots*, this is of course not completely wrong. Thus, in addition on the *literal evaluation*, we also did a *semantic evaluation*, in which such broader semantically equivalent and related categories also were counted as matches. Further examples of pairs that were counted as equivalent are given in Table 5. This type of evaluation is similar to the semantic evaluation introduced in [17]. Of course, the results from this evaluation are subjective to a certain degree, but will nevertheless able make a division between useful and completely wrong categories instead of only considering literal identity.

For evaluation we assign always the five categories with the highest rank according to the used ranking variant. As measures for the quality of the results we use precision, recall and the harmonic mean of precision and recall (F1).

6 Results

The results of the evaluation are shown in the Table 6. Interestingly, the results for the semantic evaluation are very similar for all methods, while the variant using only idf is clearly inferior to the other variants for the literal evaluation.

Table 6. Evaluation results

Method	Literal			Semantic		
	Prec.	Rec.	F1	Prec.	Rec.	F1
Variante 1	0.036	0.015	0.021	0.42	0.36	0.39
Variante 2	0.054	0.059	0.057	0.40	0.40	0.40
Variante 3	0.053	0.058	0.055	0.42	0.40	0.41

7 Conclusion and Future Work

The overall result of precision and recall both around 0.4 does not seem to be very high, but is comparable to other systems using such a high number of possible categories. In fact we should compare the results rather to keyword extraction systems than to classification systems. We have applied the proposed method (variant 3) to all images in our database. Here, in total 66,873 different categories were used for 2,889,463 images (each receiving 5 categories). As usual a few categories (e.g. *Proteins* and *Gene Expression*, both over 50,000 times) were used extremely often, while most others were assigned only a few times.

Overall we can conclude that the proposed method gives useful results that, however still can and should be improved.

We see that using word embeddings is much more useful than using idf. It turns out that the cosine similarity between the aggregated semantic vector from key phrase words and caption words is a very effective method to filter out phrases that are found in Wikipedia but that are completely unrelated to the main topic of the caption text. Since many captions are very short or do not contain any words that are found in Wikipedia, we need to include sentences referring to the image as well to get enough candidates. This might, however also be a source that introduces less relevant words. The word embeddings help to filter these words out while keeping the useful ones.

One of the main problems for developing a method to assign good Wikipedia categories of images based on their captions is the absence of larger amounts of data for training and evaluation. Thus e.g. we could not learn optimal values for the weight constants in formula 3. Another problem is the quality of the manual assigned categories. People without much domain knowledge tend to assign categories that are mentioned literally in the caption of the image. Thus any method that does the same will be preferred. For future work we hope that more and more training data will become available if we upload images to Wikimedia Commons and the images get used on Wikipedia.

A further improvement can be expected when we use a better ways to find a representation of phrases and captions than just averaging the word vectors, as e.g. was done by Schlötterer et al. [19].

Another direction that we want to explore is to find a more principled and data driven way to distinguish between categories that are useful to describe images (independent of any actual image) and categories that are not.

Acknowledgment. The presented work was developed within the NOA Project - Automatic Harvesting, Indexing and Provision of Open Access Figures from the Fields of Engineering and Technology Using the Infrastructure of Wikimedia Commons and Wikidata - funded by the DFG under grant number 315976924. NOA is a cooperative project of the Hochschule Hannover and the Technische Informationsbibliothek Hannover. We would like to thank the NOA project team.

References

1. Charbonnier, J., Sohmen, L., Rothman, J., Rohden, B., Wartena, C.: NOA: a search engine for reusable scientific images beyond the life sciences. In: Pasi, G., Piwowarski, B., Azzopardi, L., Hanbury, A. (eds.) ECIR 2018. LNCS, vol. 10772, pp. 797–800. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-76941-7_78
2. Mihalcea, R., Csomai, A.: Wikify linking documents to encyclopedic knowledge. In: Proceedings of the Sixteenth ACM Conference on Conference on Information and Knowledge Management, CIKM 2007, pp. 233–242. ACM, New York (2007). <https://doi.org/10.1145/1321440.1321475>
3. Medelyan, O., Witten, I.H., Milne, D.N.: Topic indexing with Wikipedia. AAAI Technical report WS-08-15, pp. 19–24 (2008). <http://researchcommons.waikato.ac.nz/handle/10289/1776>
4. Wartena, C., Brussee, R.: Instanced-based mapping between thesauri and folksonomies. In: Sheth, A., et al. (eds.) ISWC 2008. LNCS, vol. 5318, pp. 356–370. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-88564-1_23
5. Frank, E., Paynter, G.W., Witten, I.H., Gutwin, C., Nevill-Manning, C.G.: Domain-specific keyphrase extraction. In: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence, IJCAI 1999, pp. 668–673. Morgan Kaufmann Publishers Inc., San Francisco (1999). <http://dl.acm.org/citation.cfm?id=646307.687591>
6. Turney, P.D.: Learning algorithms for keyphrase extraction. *Inf. Retr.* **2**(4), 303–336 (2000). <https://doi.org/10.1023/A:1009976227802>
7. Mihalcea, R., Tarau, P.: Textrank: bringing order into text. In: Proceedings of 9th Conference on Empirical Methods in Natural Language Processing (EMNLP 2004) (2004). <http://ci.nii.ac.jp/naid/20001460576/>
8. Leong, C.W., Mihalcea, R., Hassan, S.: Text mining for automatic image tagging. In: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, COLING 2010, pp. 647–655. Association for Computational Linguistics, Stroudsburg (2010). <http://dl.acm.org/citation.cfm?id=1944566.1944640>
9. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. *Commun. ACM.* **18**(11), 613–620 (1975). <https://doi.org/10.1145/361219.361220>
10. Wartena, C., Brussee, R., Slakhorst, W.: Keyword extraction using word co-occurrence. In: TIR 2010–7th International Workshop on Text-Based Information Retrieval, in Conjunction with DEXA 2010, pp. 54–58, October 2010
11. Wartena, C., Sommer, M.: Automatic classification of scientific records using the German Subject Heading Authority File (SWD), October 2012. <https://serwiss.bib.hs-hannover.de/frontdoor/index/index/docId/328>
12. Voss, J., et al.: Normdaten in Wikidata, May 2014. <https://serwiss.bib.hs-hannover.de/frontdoor/index/index/docId/438>

13. Wikimedia Foundation: Wikipedia: Categorization, page Version ID: 821464874, January 2018. <https://en.wikipedia.org/w/index.php?title=Wikipedia:Categorization&oldid=821464874>
14. Bird, S., Klein, E., Loper, E.: *Natural Language Processing with Python*, 1st edn. O'Reilly and Associates, Beijing (2009)
15. English Penn Treebank tagset with modifications—Sketch Engine. <https://www.sketchengine.eu/english-treetagger-pipeline-2/>
16. Charbonnier, J., Wartena, C.: Using word embeddings for unsupervised acronym disambiguation. In: *Proceedings of the 27th International Conference on Computational Linguistics*. Association for Computational Linguistics, Santa Fe (2018, to appear)
17. Gazendam, L., Wartena, C., Malais, V., Schreiber, G., de Jong, A., Brugman, H.: Automatic annotation suggestions for audiovisual archives: evaluation aspects. *Interdiscipl. Sci. Rev.* **34**(2–3), 172–188 (2009). <https://doi.org/10.1179/174327909X441090>
18. Iivonen, M., Consistency in the selection of search concepts and search terms. *Inf. Process. Manag.* **31**(2), 173–190 (1995). <http://linkinghub.elsevier.com/retrieve/pii/030645739580034Q>
19. Schlötterer, J., Seifert, C., Granitzer, M.: Supporting web surfers in finding related material in digital library repositories. In: Fuhr, N., Kovács, L., Risse, T., Nejdil, W. (eds.) *TPDL 2016*. LNCS, vol. 9819, pp. 434–437. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-43997-6_38



Detecting Link and Landing Page Misalignment in Marketing Emails

Nedim Lipka^(✉), Tak Yeon Lee, and Eunyee Koh

Adobe Research, San Jose, CA, USA
{lipka,talee,eunyeey}@adobe.com

Abstract. Links and their landing pages in the World Wide Web are oftentimes flawed or irrelevant. We created a data set of 4266 links within 160 marketing emails whose relevance with landing pages have been evaluated by crowd workers. We present a study of common misalignments and propose methods for detecting these misalignments. An F-score of 0.63 can be achieved by a neural network for cases where the misaligned label requires the majority out of 5 crowd worker votes.

1 Introduction

The dynamic and decentralized nature of the Internet enables the immediate and flexible delivery of information. At the same time, the World Wide Web is littered with deadwood sites abandoned and woefully out of date, cf. [1]. Our prior work [2] identifies eight types of common issues that people perceive from links within marketing emails. People who click misaligned links would get disappointed by error messages or confused by irrelevant content on landing pages. Unfortunately, the detection of semantically misaligned landing pages yet relies on inefficient, inaccurate, and belated manual testing by the Website owner (i.e. marketers). This paper addresses the issue by proposing a relevance metric that automatically estimates whether a pair of link and landing page is misaligned.

To the best of our knowledge, there exists no labeled data set for training or validating a relevance metric for misalignment. We conducted an online experiment using Amazon Mechanical Turk (MTurk) that collects five labels for each of 4266 links. Results suggest that 58.1% of the links in promotional emails are perceived to be misaligned by at least one out of five people. 4.8% of the links are perceived to be misaligned by all five people.

We provide an exploratory study where we compare unsupervised baseline models and deep neural networks for predicting misalignment. The unsupervised approaches achieve an F-score of 0.74 and 0.48 for the settings where at least one and, respectively, where at least three persons identify a misalignment; the neural networks achieve 0.75 and 0.63.

The main contributions of this paper are summarized as follows:

- A study with descriptive statistics about common misalignments.
- An experimental study that compares deep neural networks and unsupervised baseline models for misalignment detection.

2 Related Work

2.1 Broken Links on the Internet

Recent studies observed that links on the Internet exhibit rapid death and decay of relevance. Fetterly et al. [3] observed that pages disappear at a rate of 0.25–0.5% per week. Hennessey and Ge [4] found that the median lifespan of hyperlinks used in scientific literature was 9.3 years. Zittrain et al. [5] have determined that approximately 50% of the URLs in U.S. Supreme Court opinions no longer point to the original information. There are many tools that detect dead links. W3C Link Checker [6] automatically tests reachability of links in a HTML document. Email marketing platforms such as Adobe Marketing Cloud, Litmus, and DotMailer [7–9] allow users to check if URLs within an email are reachable. Aside from obviously broken links returning HTTP protocol errors, a significant portion of links return substitute pages containing irrelevant information in comparison to the original page, which is called ‘soft-404s’ [10]. To our knowledge, there is no efficient method for detecting soft-404s—predicting whether people will consider a landing page current and relevant given a link.

2.2 Machine Intelligence for Online Marketing

Researchers applied a wide range of machine learning techniques for improving performance of online ads. For instance, context matching is to find ads that are more relevant to recipients’ current interest and the pages where it is shown. For instance, Chakrabarti et al. [11] proposed a logistic regression model to estimate whether an ad is relevant with the page it is shown on. Another line of research tries to predict click-through rate of ads using clustering, keyword matching, and classification in the context of sponsored search [12]. Becker et al. [13] analyzed the impact of a landing page on the user’s conversion behavior for different advertisement categories in the context of sponsored search. Rosales et al. [14] proposed a model to estimate a conversion rate after users click promotion links. While the aforementioned work on online advertising focuses on the relevancy of ad itself, but not semantic relevance of links and landing pages.

2.3 Document Similarity for Testing Link Relevance

While there is no off-the-shelf metric for assessing the relevance within a link and its landing page, an alternative way is to use document similarity measures, which have been used for various applications such as clustering similar documents, detecting plagiarism, or creating anchor text from a target document,

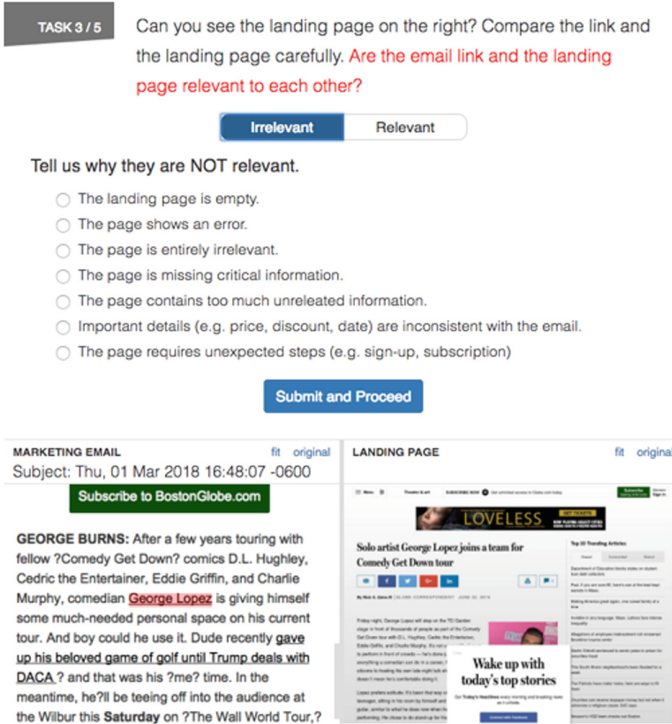


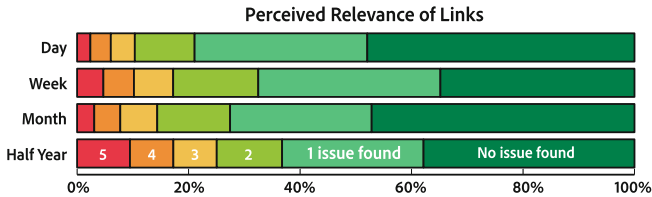
Fig. 1. An Amazon Mechanical Turk task for collecting link relevance assessments. Crowd workers cross-check a link (highlighted in red) in a marketing email (bottom left) and its landing page (bottom right). They assess whether the landing page is misaligned and choose the most accurate type of misalignment. (Color figure online)

cf. [15–17]. For instance, if a link and its landing page are very similar, users may perceive them relevant. However, it is not as straightforward as comparing two text documents since the user’s expectation of landing page content would not only rely on the link text but also on images, text around the link, and even the entire email. Gong et al. [18] proposed using image as well as text information to improve information extraction results. Similarly, we extracted themes and written text from images within an email using OCR and image recognition algorithms.

Semantic similarity can be learned by neural networks. One common approach is to use a Siamese architecture [19], which is based on two identical neural networks—with shared weights—that each take two input vectors representing documents, images, or other media. The output of this architecture is the score of a similarity function that is applied to the previously computed activation tensors. The training is based on pairs that are either relevant (positive examples) or irrelevant (negative examples). Siamese architectures are commonly used for one- or zero-shot learning applications [20–22].

Table 1. Groups in our data set based on the email age relative to the start date of the labeling task. Some groups have less than 40 emails because some image content failed to load. The lower graphic depicts the distribution of agreement among crowd sourcers; e.g., ‘No issue found’ means that all collected votes indicate the landing page to be relevant.

<i>Email Group</i>	<i># emails</i>	<i># links per email</i>		<i># issues per link</i>	
		<i>Avg.</i>	<i>Std.</i>	<i>Avg.</i>	<i>Std.</i>
<i>day</i> (0-3 days old)	40	27.3	15.5	0.18	0.24
<i>week</i> (7-10 days old)	40	29.2	20.8	0.26	0.28
<i>month</i> (30-33 days old)	38	28.2	20.2	0.21	0.27
<i>half year</i> (180-183 days old)	35	26.8	26.6	0.30	0.33
Overall	153	27.9	21.0	0.24	0.28



Another general approach is to train or reuse a pre-trained paragraph or word embedding [23, 24] in combination with a similarity function such as the cosine similarity or the Word Mover’s Distance [25].

3 Data Collection

This section describes how we created a labeled data set with 4266 links from 160 emails, which were chosen from the author’s Google Mail account. In order to increase the diversity of link decay, we picked emails from four groups as summarized in Table 1. First, the *day* group contains emails received within three days prior to the experiment. Emails in the *week* group were 7–10 days old, etc. The emails in each group stem from unique companies. Overall, each email contains on average 27.9 links ($MIN = 3$, $MAX = 165$, $STD = 21.0$). We then extracted information required for data collection such as email screen-shots, links and images. Second, we retrieved landing pages of the links using PhantomJS, took their screen-shots, and extracted textual information.

3.1 Method

We created a crowdsourcing task for labeling data and posted it on MTurk. Each session begins with demographic questions about age and gender and a tutorial that is introducing the experimental user interface that is shown in Fig. 1. The UI displays an email with a link and its landing page side-by-side. The participants are asked to evaluate the relevance between link and page. When they vote for

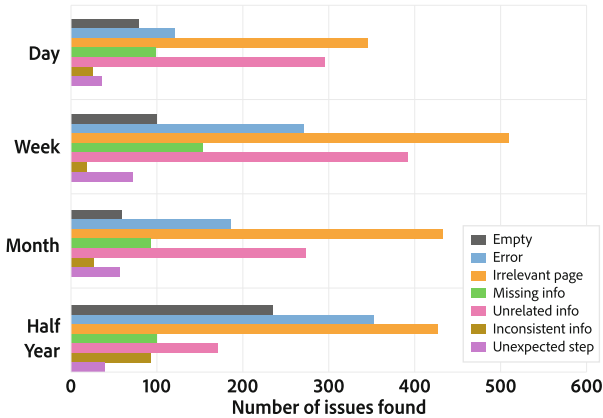


Fig. 2. Irrelevant page is the most frequent type of misalignment for any email age group. Older emails in the group *half year* received more reports of **empty**, **error**, and **inconsistent info**. However, the number of **unrelated info** is significantly larger for the group *week*.

‘misaligned’, they are also asked to select the type of misalignment. The proposed types are based on the results of our previous study [2] that identified groups of misalignments.

Each participant evaluated five links that were randomly drawn from one email. For the ground-truth relevance score of each link, we use the number of votes that are indicating a misalignment. For instance, a perfectly relevant link would have no misalignment at all.

3.2 Human Assessment

A total of 4266 MTurk crowd workers participated in our experiment. Their average age was 31.2 ($MED = 28$, $STD = 9.4$) with 57.1% of them being male and 42.9% female.

Decay of Link Relevance. Prior work suggests that link relevance would decay over time. It turns out to be only partially the case for our data as the decay is not monotonic. The group *day* has the smallest number of links with five votes for ‘misaligned’ (2.4%) and the largest number of links with no issue (47.9%). In contrast, the group *half year* has the largest number of links with five identified issues (9.5%). However, the group *month* has less identified issues than *week*.

Mann-Whitney U posthoc comparisons after a Bonferroni correction show that the group *day* has significantly more relevant links than the groups *week* and *half year*, $p < 0.001$. This also applies when comparing *month* and *half year*. Overall, the Kruskal-Wallis H test shows there is a significant statistical difference between all groups, $H = 80.1$, $p < 0.001$.

Types of Misalignments. The most common type of misalignment, cf. Fig. 2, is **irrelevant page**, reported 1 712 times, followed by **unrelated info** (1 131), **error** (929), **empty** (472), **missing info** (443), **unexpected step** (203), and **inconsistent info** (162).

There is a significant difference between the misalignment distributions across our email age groups, $\chi^2 = 376.2$, $p < 0.001$. For instance, the *half year* group suffers from the misalignments **empty**, **error**, and **inconsistent info** more frequently compared to emails in the other groups, $p < 0.001$. However, misalignment **unrelated info** was reported more frequently for emails in the group *week* group, $p < 0.001$.

4 Detecting Misalignment

Given a link and its landing page, our task is to predict if visitors perceive the landing page misaligned. We approach this problem using unsupervised, i.e., cosine similarity in combination with vector space models as well as deep neural network models.

For our study, we define three settings with different voting thresholds that determine our ground-truth. The number of votes for the target class label ‘misaligned’ is defined by the number of people that indicated an issue with the landing page in the MTurk task. For a minimum risk setting we consider one or more votes sufficient for the target class, for majority voting we require at least 3 votes, and for complete agreement we require 5 votes. Increasing the number of votes required for the target class decreases the number of target instances, which changes the class balance.

Figure 3 shows the precision, recall, and F-scores of two unsupervised baseline models applied to the three classification settings. Both baseline models compute the cosine similarity between a landing page and a concatenation of the link text, its corresponding email body, and the subject line. The similarity scores are used to make binary class predictions by employing a threshold (cut-off) t : if the similarity is below t the landing page is considered misaligned. The first model is based on a TFIDF weighted vector space model while the second one uses the Universal Sentence Encoder from Tensorflow in order to compute a 512-dimensional text embedding as described in [23]. The y -axis shows the performance of a classifier with a cut-off at t (indicated by the x -axis). If $t = 1$, the precision score reflects the class balance. The recall curve provides information about the distribution of similarity scores in the target class. Analogous, the precision curve shows the distribution of similarities in the non-target class.

We study variations of two feedforward deep learning architectures for classification. All of these models combine features based on the landing page content with different sets of features representing the link or its email context. Out of experiments with TFIDF, term frequencies, and term occurrences for vectorizing text, we report results based on binary term occurrence vectors, which lead to the best effectiveness on the test sets. In order to represent a link, we combine the anchor text of the link with extracted OCR information and image tags

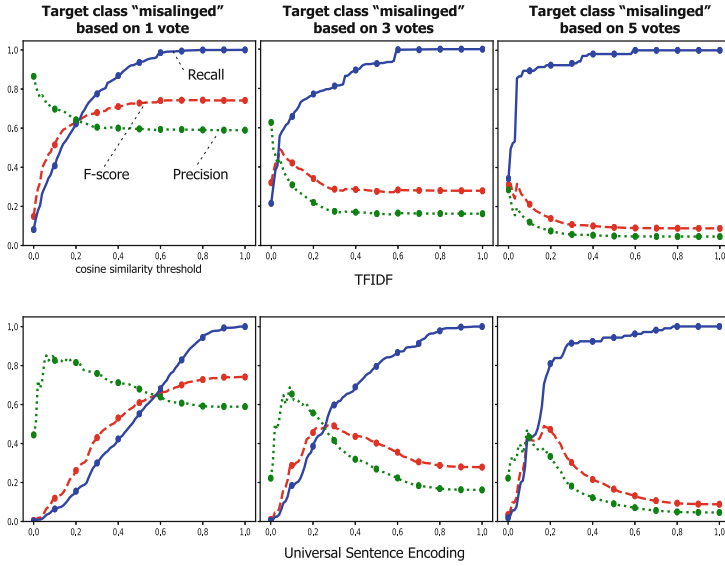


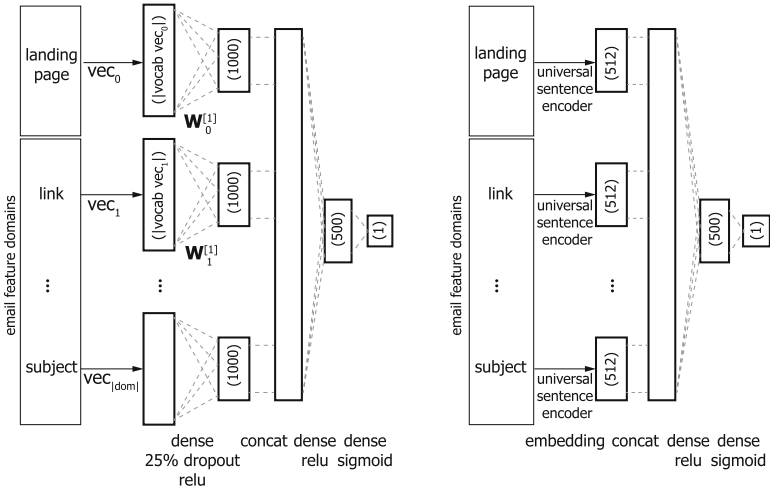
Fig. 3. Baseline precision, recall, and F-scores for two unsupervised models and different problem parameters.

in case an image has been available. We are using the Google Tesseract OCR engine [26] and the Inception V3 model [27] for these tasks. Furthermore, the subject line and the body of the email that includes the corresponding link can be input to the proposed models. We report the combinations of inputs that lead to the highest F-score.

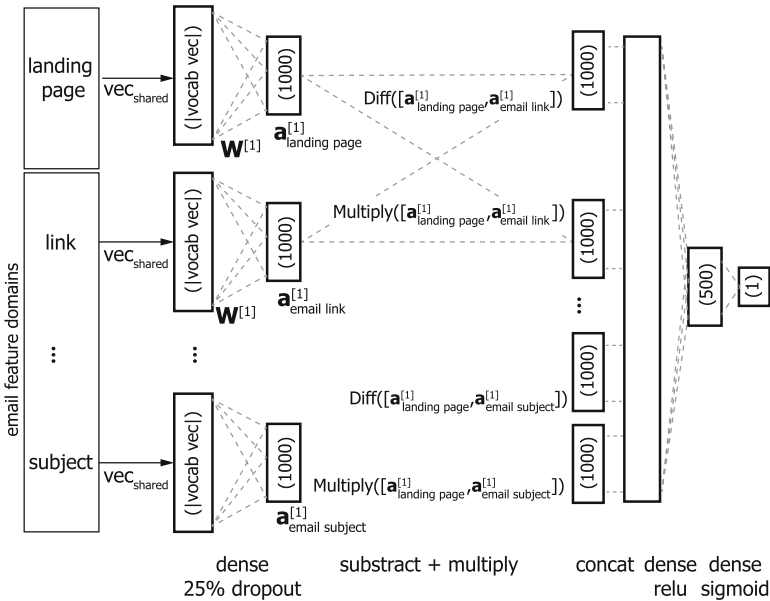
Figure 4a shows variations of a simple deep learning architecture. Each input in variation Fig. 4a (left) has a lower dimensional distributed representation with rectifiers as activation functions and dropout. As illustrated, each input uses an individually fitted, lower cased vocabulary in order to map text to a numeric vector space. It follows a dense layer with rectified linear units before the output layer. Variation Fig. 4a (right) differs in terms of the input embedding which is inferred by employing the pre-trained Unified Sentence Encoder [23].

Figure 4b aims to model the semantic difference between a link and its landing page by computing element-wise differences and multiplications. Therefore, the vocabulary as well as the embedding weights of the first fully connected layer are shared across all inputs. This is similar to a Siamese architecture, cf. [19]. It follows a lower dimensional dense layer with rectified linear units and the output layer.

Table 2 also includes a variation of Fig. 4b, named Fig. 4b (dot), that is closer to the original Siamese architecture as it uses the dot products between landing page and email features instead of element-wise differences and multiplications.



(a)



(b)

Fig. 4. (a) Two variations of a simple feedforward deep learning architecture. Inputs are landing page and various link features, whereby (a) – left, uses term occurrences and individual vocabularies per feature domain, and, (a) – right, a pre-trained text embedding. (b) A deep learning architecture that uses shared embedding weights and models the differences between email and landing page features using subtraction and multiplication layers.

5 Evaluation

Table 2 summarizes precision, recall, and F-scores for the proposed architectures for various settings. A settings that requires a large number of votes for the target class can end up with too few or even no target examples, which results into an impractical scenario (zero values in the table). We report the combination of link inputs that results in the highest F-score for each architecture. The experiments use an early stopping criteria based on the validation loss as generalization strategy. For model fitting we employ binary cross entropy as loss function, balanced class weights, and the Adam optimization algorithm [28].

In the minimum risk setting that requires at least one vote for the target class, the neural network architectures described in Fig. 4a achieve the best F-scores in the majority of the cases. In comparison to our unsupervised baselines, the neural networks are not significantly better for the task ‘all’, which is detecting misalignment without specifying the type.

6 Conclusion

We study the problem of detecting misaligned landing pages given a link in a marketing email. Based on our data set, we find that misalignment is a frequent problem. As digital marketing is moving to more personalized messaging, automatized solutions that capture ‘soft-404s’ become increasingly desirable. The concept of ‘relevance’ in this context is hard to grasp, which can be seen in the distribution of agreement among crowd workers, cf. Table 1. In this paper, we show a first report of effectivenesses of various deep learning models that are suitable for this problem.

References

1. Jesdanun, A.: Internet Littered with Dead Web Sites. The Associated Press (2003)
2. Lee, T.Y., Koh, E.: Identifying types of misalignments between promotion emails and landing pages. In: CHI 2018 Extended Abstracts on Human Factors in Computing Systems (2018)
3. Fetterly, D., Manasse, M., Najork, M., Wiener, J.: A large-scale study of the evolution of web pages. In: Proceedings of the 12th International Conference on World Wide Web (2003)
4. Hennessey, J., Ge, S.X.: A cross disciplinary study of link decay and the effectiveness of mitigation techniques. BMC Bioinform. **14**, S5 (2013)
5. Zittrain, J., Albert, K., Lessig, L.: Perma: scoping and addressing the problem of link and reference rot in legal citations. Legal Inf. Manag. (2014)
6. <https://validator.w3.org/checklink>
7. <https://www.adobe.com/marketing-cloud.html>
8. <https://litmus.com>
9. <https://www.dotmailer.com>
10. Bar-yossef, Z., Kumar, R., Broder, A.Z., Tomkins, A.: Sic Transit Gloria Telae: towards an understanding of the web’s decay. In: Proceedings of the 13th Conference on World Wide Web (2004)

11. Chakrabarti, D., Agarwal, D., Josifovski, V.: Contextual advertising by combining relevance with click feedback. In: Proceedings of the 17th International Conference on World Wide Web (2008)
12. Regelson, M., Fain, D.C.: Predicting click-through rate using keyword clusters. In: Proceedings of EC 2006 (2006)
13. Becker, H., Broder, A., Gabrilovich, E., Josifovski, V., Pang, B.: What happens after an ad click?: quantifying the impact of landing pages in web advertising. In: Proceedings of the 18th ACM Conference on Information and Knowledge Management (2009)
14. Rosales, R., Cheng, H., Manavoglu, E.: Post-click conversion modeling and analysis for non-guaranteed delivery display advertising. In: Proceedings of the 5th ACM International Conference on Web Search and Data Mining (2012)
15. Huang, A.: Similarity measures for text document clustering. In: Proceedings of the Sixth New Zealand CS Research Student Conference (2008)
16. Lukashenko, R., Graudina, V., Grundspenkis, J.: Computer-based plagiarism detection methods and tools: an overview. In: Proceedings of the International Conference on Computer Systems and Technologies (2007)
17. Reberšek, P., Verlic, M.: Application of localized similarity for web documents. In: Proceedings of the Conference on Empirical Methods in Natural Language Processing (2013)
18. Gong, D., Wang, D.Z., Peng, Y.: Multimodal learning for web information extraction. In: Proceedings of the ACM on Multimedia Conference (2017)
19. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (2005)
20. Koch, G., Zemel, R., Salakhutdinov, R.: Siamese neural networks for one-shot image recognition. In: Proceedings of the 32nd International Conference on Machine Learning (2015)
21. Mueller, J., Thyagarajan, A.: Siamese recurrent architectures for learning sentence similarity. In: Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (2016)
22. Qiao, R., Liu, L., Shen, C., van den Hengel, A.: Less is more: zero-shot learning from online textual documents with noise suppression. CoRR (2016)
23. Cer, D., et al.: Universal sentence encoder. arXiv e-prints (2018)
24. Pennington, J., Socher, R., Manning, C.D.: GloVe: global vectors for word representation. In: Empirical Methods in Natural Language Processing (2014)
25. Kusner, M.J., Sun, Y., Kolkin, N.I., Weinberger, K.Q.: From word embeddings to document distances. In: Proceedings of the 32nd International Conference on Machine Learning (2015)
26. Smith, R.: An overview of the Tesseract OCR engine. In: Proceedings of Ninth International Conference on Document Analysis and Recognition (2007)
27. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.: Rethinking the inception architecture for computer vision. CoRR (2015)
28. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. CoRR (2014)



Toward Validation of Textual Information Retrieval Techniques for Software Weaknesses

Jukka Ruohonen^(✉)  and Ville Leppänen

Department of Future Technologies, University of Turku, Turku, Finland
{[juanruo](mailto:juanruo@utu.fi), [ville.leppanen](mailto:ville.leppanen@utu.fi)}@utu.fi

Abstract. This paper presents a preliminary validation of common textual information retrieval techniques for mapping unstructured software vulnerability information to distinct software weaknesses. The validation is carried out with a dataset compiled from four software repositories tracked in the Snyk vulnerability database. According to the results, the information retrieval techniques used perform unsatisfactorily compared to regular expression searches. Although the results vary from a repository to another, the preliminary validation presented indicates that explicit referencing of vulnerability and weakness identifiers is preferable for concrete vulnerability tracking. Such referencing allows the use of keyword-based searches, which currently seem to yield more consistent results compared to information retrieval techniques. Further validation work is required for improving the precision of the techniques, however.

Keywords: Text mining · Software vulnerability · Snyk · LSA
CVE · CWE · NVD

1 Introduction

Software weaknesses—as cataloged in the so-called Common Weakness Enumeration (CWE) framework—are abstractions for security-related mistakes made in software development. Such weaknesses may lead to concrete software vulnerabilities. The CWE framework covers numerous different weaknesses, ranging from software design flaws to inadequate input validation, insecure maintaining of time and state, and lack of encapsulation [34]. The richness of the framework has ensured its usefulness for both research and practice. To name a few of the application domains, the CWE framework has been used for security (compliance) assessments [8, 9], risk analysis [1, 6], quantitative trend analysis [22], data mining [13], static source code analysis [25], dissemination of fuzzing results [15], and last but not least, education and security awareness [16]. Also text mining applications have been common, although there are still gaps in the literature.

Many of the text mining applications have relied on the Open Web Application Security Project (OWASP), which limits the generalizability of the applica-

tions [21,27]. Another gap relates to the common focus on CWE-based ontologies. Although such ontologies are useful for understanding and clustering weaknesses [2,10,36], these have a limited appeal for vulnerability tracking. Here, the term vulnerability tracking refers to the concrete, largely manual software engineering work required for archiving and documenting software vulnerabilities. If this work does not explicitly cover weaknesses, an application of an ontology-based technique must first solve the problem of extracting the CWEs from the typically more or less unstructured textual vulnerability data. Limited attention has also been given for the validity of the text mining applications.

To contribute toward sealing some of these gaps, this paper tentatively examines the validity of common textual information retrieval techniques for extracting CWEs from vulnerability databases. The extraction itself has practical value because many vulnerability databases do not catalog weaknesses, partially due to the complexity of the CWE framework and the manual work required [35]. By superseding the manual assignment of security bug reports to CWEs [8], automatic extraction can also facilitate empirical security research. It is important to further emphasize that the task differs from conventional information retrieval systems that can return multiple documents for a single query. In contrast, this paper adopts a much stricter constraint: each unique vulnerability should map to a single unique CWE-identified weakness. As elaborated in the opening Sect. 2, this constraint can be also used for comparing common textual information retrieval techniques against simple regular expression searches. The comparative results are presented in Sect. 3 and discussed in Sect. 4.

2 Materials

The following will outline the data sources, the subset of data used for the validation, the pre-processing routines, and the weights used for computation.

2.1 Data Sources

The dataset is compiled from three distinct but related sources. The first source is the conventional National Vulnerability Database (NVD) maintained by the National Institute of Standards and Technology (NIST) in cooperation with the non-profit MITRE corporation. This database provides one-to-one mappings between abstract weaknesses identified with CWEs and concrete vulnerabilities identified with Common Vulnerabilities and Exposures (CVEs). These mappings are based on expert opinion; during the archival of vulnerabilities to the database, NVD's maintainers derive the weaknesses from the concrete vulnerabilities archived. At the time of retrieving the database's content [23], there were 102 unique CWEs in the database once rejected CVEs were excluded. (These invalid cases are marked with the string REJECTED in the summary field of a CVE.) These CWEs were used for assembling the estimation subset soon discussed.

The second source is the CWE database maintained by MITRE in cooperation with volunteers and governmental sponsors. In total, there were 730 documented weaknesses in the database at the time of retrieval [18]. These weaknesses have been used to construct different ontologies [10, 27, 36], including the famous “seven pernicious kingdoms” of security-related programming mistakes [34]. Reflecting such ontologies, MITRE provides also many predefined views to subsets of the weaknesses archived. In the information retrieval context the relevant view is the one pointing to the CWEs used by NVD. Due to recent changes made [19], however, the predefined NVD-specific view is not suitable. Therefore, data is used from the CWE database only for the 102 weaknesses that are present also in NVD indirectly via CVE mappings. Although the present context is weaknesses, the idea here is similar to the enforcement of “one-to-one vulnerabilities” between vulnerability databases [35]. In contrast to some previous studies [10], all textual information is used for constructing the corpora. This information contains also meta-data strings, but the results reported did not differ much from those obtained by including only fields specific to natural language. The pre-processing described later on also filters out much of the meta-data.

The third and final source is the so-called Snyk database used for tracking security issues in open source software packages particularly in the web development context [32]. In contrast to the primary package managers used in Linux distributions, Snyk targets the secondary package managers and their repositories that are specific to programming languages. Although the Snyk database has seldom been used for research purposes, the underlying repositories have been studied extensively (see [20, 29, 33], for instance). The following four repositories are included in the dataset assembled: *Maven* (Java), *pip* (Python), *npm* (JavaScript), and *RubyGems* (Ruby). In addition to the web development context, these repositories were selected due to sufficient amounts of vulnerabilities reported for the packages within the repositories. Therefore, the selection used allows to check whether the results are specific only to some repositories.

As is typical in vulnerability tracking [4, 21, 30], the Snyk database contains first-order and (online) second-order relations. As illustrated in Fig. 1, these relations can be either direct or indirect with respect to CWE and CVE identifiers. For instance, the following vulnerability report (pymongo/40183) in the Snyk database represents a second-order indirect relation because the CWE in question can be mapped from the CVE visible in the link pointing to NVD:

```
## Overview
```

```
['pymongo'](https://pypi.python.org/pypi/pymongo) is a Python driver for MongoDB.
```

```
'bson/_cbsonmodule.c' in the mongo-python-driver (aka. pymongo) before 2.5.2, as used in MongoDB, allows context-dependent attackers to cause a denial of service (NULL pointer dereference and crash) via vectors related to decoding of an "invalid DBRef."
```

References

- [NVD] (<https://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2013-2132>)
- [Github Commit] (<https://github.com/mongodb/mongo-python-driver/commit/a060c15ef87e0f0e72974c7c0e57fe811bbd06a2>)

The first-order relations refer to the primary textual content stored to the database. As can be seen from the above excerpt, Snyk’s maintainers archive each vulnerability with a brief textual description, which can be potentially mapped to a CWE with different text mining techniques. When constructing the corpora, all textual information is used except comment fields (lines starting with a #) and links to further online material (lines starting with a dash). The additional online material constitute the second-order relations. Most vulnerabilities archived to Snyk are accompanied with one or more links pointing to security advisories, blogs, mailing lists, bug trackers, version control systems, hosting services such as GitHub, and other vulnerability databases, including NVD in particular. The content behind each of these links was downloaded, and for each successful download, all textual information was further added to the corpora sans the hypertext markup language elements. Given the terseness of the primary (first-order) textual information in the Snyk database, the additional (second-order) online material is beneficial for enlarging the corpora observed.

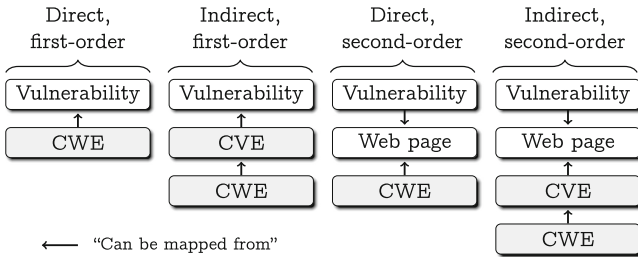


Fig. 1. An example of four abstract relations for software weaknesses

Thus, to summarize, the dataset contains textual information about software weaknesses from the CWE database, and first-order textual information about vulnerabilities from the Snyk database, as well as second-order textual information from online sources referenced in the Snyk database. The goal is to validate how well the vulnerabilities can be mapped to the CWEs. For this validation task, empirical estimation is carried out in a specific subset of the dataset.

2.2 Estimation Subset

A basic difficulty in classifying text documents is the typical absence of a ground truth against which text mining results can be compared. A typical way to tackle the issue is to compare text mining results against labels made by human experts [3]. The same applies in the vulnerability context [4, 27]. However, the

approach adopted takes a different path: the text mining results are compared against simple but relatively robust regular expression searches. For each vulnerability, the following two simple regular expressions were used to search for direct CWE mappings and indirect “CWE-from-CVE-in-NVD” mappings:

$$(? :CWE) [-] [0-9] \{1, \} \quad \text{and} \quad (? :CVE|CAN) [-] [0-9] \{4\} - [0-9] \{4, \}. \quad (1)$$

Only if a vulnerability matched only once from either one of the expressions, it was qualified to the estimation subset alongside the corresponding CWE, provided that the CWE was present among the 102 weaknesses in NVD. Thus, each vulnerability in the estimation subset can be mapped with regular expression searches to a single unique weakness. It can be further noted that techniques such as majority-voting are unsuitable. The reason is simple. Many of the second-order relations point to web pages that catalog all CVEs assigned for a given package. By per-vulnerability voting based on the frequency of CVEs (or CWEs) mentioned in such pages, the uniqueness condition would be lost. Such voting would presumably also lead to haphazard mappings. Given these remarks,

$$n_1 = 82 \text{ weaknesses and } n_2 = 585 \text{ vulnerabilities} \quad (2)$$

were qualified to the estimation subset. For comparing the regular expression searches against information retrieval techniques, a common metric can be used:

$$\text{Precision} = \frac{(\# \text{ same CWE})}{(\# \text{ same CWE}) + (\# \text{ different CWE})}, \quad (3)$$

where the numerator denotes the frequency of vulnerabilities that were assigned to the same CWEs by both the regular expression and information retrieval techniques. The second term in the denominator refers to the frequency of vulnerabilities assigned to different CWE identifiers by the two techniques. Due to the described operationalization of the estimation subset, additional metrics (such as recall and accuracy) are not meaningful—it cannot be deduced whether the remaining vulnerabilities in the Snyk database are relevant or irrelevant with respect to CWEs. That said, it would be possible to use more metrics by splitting the estimation subset further into training and test sets [22], but, as will be seen, the estimation subset and the precision metric are alone sufficient for summarizing the paper’s empirical validation results.

An important further remark should be made. Although the two distinct terms in (3) connote with “true positives” and “false positives”, these terms do not convey their usual meanings in the present context: it is impossible to say with certainty which one of the two techniques is (in)valid. On one hand, the whole vulnerability tracking infrastructure is based on the unique CVE identifiers that are easy to search based on keywords [13, 31]. Therefore: if a vulnerability archived to Snyk can be mapped with a regular expression search to a unique CWE either directly or indirectly via a CVE identifier, there is a fairly good chance that the vulnerability truly reflects the given software weakness. This argument is reinforced by the sample: many of the vulnerabilities archived

to Snyk are accompanied with CVE-specific links to NVD’s website, which, in turn, contains also the CWEs assigned for the CVEs in question. On the other hand, some vulnerabilities archived to the Snyk database contain explicit notes that these particular vulnerabilities differ from those archived to NVD with some particular CVE identifiers. Such remarks cause some false positives for the regular expression searches but not for the information retrieval techniques.

2.3 Pre-processing

The pre-processing routine is fairly typical. To begin with, all textual data was *transformed* by removing hypertext markup language elements and then lower-casing all letters. After the transformation, the data was *tokenized* according to white space, punctuation characters, and other elements separating word boundaries. The resulting tokens were subsequently *trimmed* by excluding tokens containing non-alphabetic characters, tokens that are common stop words, as well as tokens with length less than three or more than twenty characters. The remaining tokens were finally *stemmed* with the Porter’s [28] classical algorithm. The NLTK library [24] was used for the tokenization, stop words, and stemming.

The stemmed tokens were used for three types of “bag-of-words” matrices:

$$\mathbf{W}_{(k)}, \quad \text{where } k \in \{1, 2, 3\}. \quad (4)$$

An element $w_{(k)ij}$ in a $\mathbf{W}_{(k)}$ denotes the *weight* given for the j :th tokenization-based k -gram (i.e., a unigram, a bigram, or a trigram) in the i :th *document* (i.e., either a given CWE or a given vulnerability). The number of rows remains constant in each matrix: $i = 1, \dots, n_1 + n_2$, where n_1 and n_2 are defined in (2). Given that the use of bigrams and trigrams substantially enlarges the amount of data, the number columns varies: $j = 1, \dots, m_k$, where m_k denotes the number of unique k -grams observed. It should be also noted that before assigning the actual weights, the underlying abstract data structures were pruned by excluding those k -grams that were present in a given corpus only twice or less.

2.4 Weights

Five different weights are used for the empirical validation. The *term* frequency (that is, in the present context, k -gram frequency) provides the starting point:

$$\text{TF} : w_{(k)ij} = f_{(k)ij}, \quad w_{(k)ij} \in \mathbf{W}_{(k)}, \quad (5)$$

where $f_{(k)ij}$ denotes the number of occurrences of the j :th k -gram in the document i . In addition, two simple TF-derivatives are empirically explored:

$$\text{TF-LOG} : w_{(k)ij} = \log(f_{(k)ij} + 1) \quad \text{and} \quad (6)$$

$$\text{TF-BOOLEAN} : w_{(k)ij} = \begin{cases} 0 & \text{if } f_{(k)ij} = 0, \\ 1 & \text{if } f_{(k)ij} > 0. \end{cases} \quad (7)$$

Although there is no particular prior reason to expect that the results would differ with respect to these three simple weighting schemes, the usual rationale for TF-LOG, for instance, is based on the reasoning that the importance of a term is unlikely to grow linearly as implied by the TF weights [26]. The same rationale works also behind the current *de facto* weighting scheme, the so-called term-frequency-inverse-document-frequency (TF-IDF). Despite of the scheme’s complex name, the actual weighting is simple:

$$\begin{aligned} \text{TF-IDF} : w_{(k)ij} &= f_{(k)ij} \times \omega_{(k)j}, \quad \text{where} \\ \omega_{(k)j} &= \log([n_1 + n_2 + 1] / [\tilde{n}_{(k)j} + 1]) + 1 \end{aligned} \tag{8}$$

and $\tilde{n}_{(k)j}$ denotes the number of documents containing the j :th term (that is, k -gram). The IDF term, $\omega_{(k)j}$, used in (8) is one of the many smoothed variants of the standard IDF formula. With smoothing or no smoothing, the rationale behind the IDF term is to penalize the occurrence of common terms. Even though the TF-IDF weights perform well in many applied problems, the same rationale has been used to define also many other more or less analogous weights [14, 26]. To empirically explore one of these IDF-based derivatives, the so-called (pivoted) document length normalization (DLM) is as a good choice as any. It is given by

$$\text{DLM-IDF} : w_{(k)ij} = \left[\frac{1 + \log(f_{(k)ij} + 1)}{(1 - \beta) + \beta(L_{(k)i} / \bar{L}_{(k)})} \right] \omega_{(k)j}, \tag{9}$$

where β is fixed to a scalar 0.2, $L_{(k)i}$ denotes the length of the i :th document (defined as the sum of all term frequencies), and $\bar{L}_{(k)}$ refers to the average document length in a whole corpus [12]. The five different weights enumerated are used for disseminating the subsequently discussed empirical validation results.

3 Results

The five different weights from (5) to (9) were assigned to each of the matrix types noted in (4). This assignment resulted in fifteen weight matrices that establish the basis for the empirical results. For computing the similarities between the CWEs and vulnerabilities observed, the standard cosine similarity is used for each of the matrices. If $\tilde{\mathbf{C}}$ denotes a document-by-document matrix of cosine similarities, a $n_2 \times n_1$ vulnerability-by-weakness matrix \mathbf{C} can be obtained by deleting n_1 rows from $\tilde{\mathbf{C}}$ that refer to CWEs, and carrying out an analogous operation for the columns. Then, each vulnerability (row) is mapped to the corresponding maximum row value in \mathbf{C} . If ties are present (meaning that a vulnerability has the same maximum cosine similarity with two or more CWEs), the vulnerability is mapped to the CWE picked by the regular expression searches, provided that any of the maximum values map to this particular CWE identifier.

The so-called latent semantic analysis (LSA) was also briefly examined as an additional validation check. Although the usefulness for applied problems has been debated [7], LSA builds on the rationale that a rank-reduced similarity matrix based on linear combinations may better reveal the underlying latent

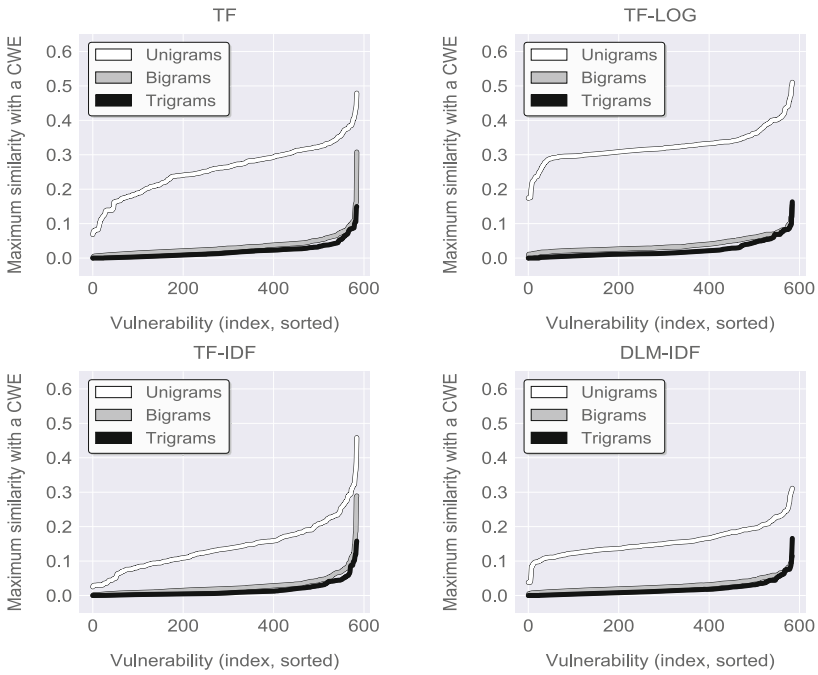


Fig. 2. Maximum cosine similarities according to four weights

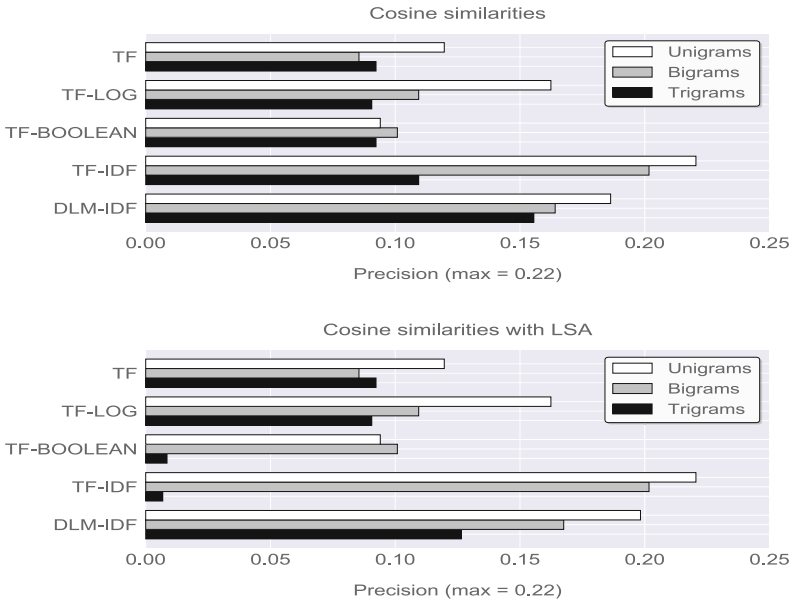


Fig. 3. Precision with five weights

structure of a corpus. Computation is simple but expensive: LSA is a straightforward application of the fundamental singular value decomposition, $\tilde{\mathbf{C}} = \mathbf{U}\mathbf{S}\mathbf{V}^T$. Without delving into the linear algebra details, the actual reduction is also simple: a predefined number of the descending singular values in the diagonal matrix \mathbf{S} is restricted to zero, after which $\tilde{\mathbf{C}}$ is reconstructed by using the manipulated diagonal matrix [3, 11]. A simple heuristic was used for the manipulation: all singular values less than or equal to one were set to zero. For each of the fifteen matrices, roughly about a quarter of the singular values satisfied this heuristic.

Given these computational remarks, the results indicate only modest similarities between the CWEs and the vulnerabilities archived to Snyk. To illustrate this observation, Fig. 2 displays the maximum cosine similarities with four weights. Unigrams perform much better than bigrams or trigrams regardless of the weights used. The likely explanation relates to the numbers in Table 1, which shows the number of columns in $\mathbf{W}_{(k)}$ and the average document lengths used for the DLM-IDF weights in (9). That is: even after the pruning of the weight matrices, many of the bigrams and trigrams appear only in few documents. The TF and TF-LOG weights also attain higher maximum similarities than the two IDF-based weights shown in the two plots on the bottom row in Fig. 2. On average, however, the clear majority of the cosine similarities are well below 0.5 for all weights used. These low similarity values translate into poor precision. As illustrated in Fig. 3, the maximum precision is as low as 0.22. The TF-IDF weights perform the best in this regard. LSA does not improve the precision.

Table 1. Descriptive statistics

	Unigrams	Bigrams	Trigrams
Unique k -grams (m_k)	8435	32166	31745
Average document length ($\bar{L}_{(k)}$)	1095	935	839
• Average CWE length	424	357	252
• Average vulnerability length	1175	1016	921

Table 2. Average per-repository precision (TF-IDF)

	<i>Maven</i>	<i>pip</i>	<i>npm</i>	<i>RubyGems</i>
Unigrams	0.17	0.34	0.55	0.25
Bigrams	0.16	0.31	0.09	0.62
Trigrams	0.10	0.12	<0.01	0.50

All in all, the validity of common textual information retrieval techniques seems questionable for software weaknesses. That said, the results diverge considerably between the four repositories (see Table 2). While the results are consistently poor particularly for *Maven* and to a lesser extent *pip*, an average precision

of 0.55 is obtained for *npm*. Interestingly, the use of bigrams raises the average precision to 0.62 for *RubyGems*. Thus, it can be also concluded that the poor precision cannot be generalized to all repositories or programming languages.

4 Discussion

This paper presented a preliminary validation of common textual information retrieval techniques for mapping vulnerabilities to weaknesses. When compared to basic regular expression searches, the techniques seem to perform poorly according to the results based on four repositories tracked in the Snyk vulnerability database. For searching specific vulnerabilities or weaknesses from software repositories, simple keyword searches based on CVE and CWE identifiers seem more robust. These commonly used [4, 13, 31] domain-specific searches could be augmented by the information retrieval techniques [5], however. In other words: it might be possible to prefer the regular expression searches as a primary retrieval technique and use the information retrieval techniques as a secondary method for retrieving additional content not captured by the keyword-based searches. It is also important to stress that the results vary across repositories. This observation hints that the choice over particular security-related corpora has likely a strong effect upon the vulnerability-CWE mappings.

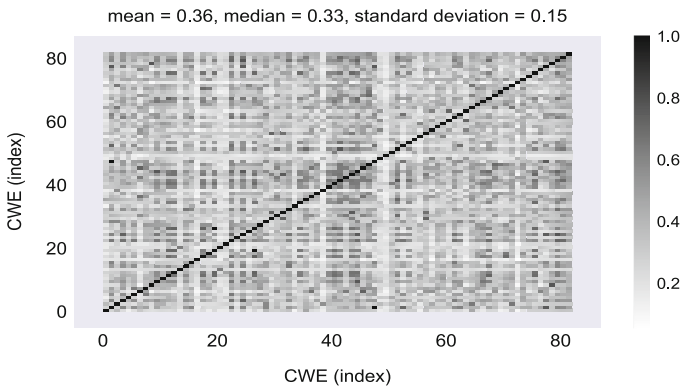


Fig. 4. Similarities between weaknesses (TF-IDF, unigrams)

Three points can be noted about limitations and directions for further research. First, as CWE is a hierarchical classification system, it might be argued that larger ontology-based weakness groups should be used. However, all of the individual CWEs observed in this paper have been tracked in NVD, which undermines the rationale for using such CWE groups in the present context. The construct validity is also undermined by the illustration in Fig. 4, which shows that the CWEs observed are not very similar with respect to each other.

Second, many computational checks could be done to further validate the vulnerability-CWE mappings. For instance, the cosine similarity used could be verified against other commonly used similarity metrics. (It can be noted that the so-called Jaccard similarity performs even worse, however.) Further examples include the calibration of the LSA's reduction step, the use of a more aggressive stemming algorithm, and the examination of part-of-speech tagging.

Third, it seems reasonable to try to either enlarge or enrich the datasets used for validation. The textual information stored to Snyk, CWE, OWASP, and related databases is terse in terms of natural language and prose, but there are technical terms that should be specifically weighted. If the small excerpt shown in Subsect. 2.1 is taken as an example, the trigrams **denial of service** and **NULL pointer dereference** should attain higher weights than any of the other k -grams. Such domain-specific weights entail the construction of a reference corpus. Given the generally poor precision reported and the variance across repositories, it may also be that neither CWE nor OWASP are ideal for a reconstruction of a reference corpus. Instead, language-specific guides for secure programming [17] may be more suitable, among other potential sources related to software weaknesses and vulnerabilities. As an alternative to enrichment, big data analysis is also plausible. Due to the central role of CVE identifiers (cf. Fig. 1), web crawling could be used to gather a truly massive dataset for text mining. Recent work [4] shows also some promise for web crawling approaches. But the larger the datasets, the coarser the mappings, and the bigger the validity concerns.

References

1. Alsaleh, M.N., Al-Shaer, E., Husari, G.: ROI-driven cyber risk mitigation using host compliance and network configuration. *J. Netw. Syst. Manag.* **25**(4), 759–783 (2017)
2. Bojanova, I., Black, P.E., Yesha, Y., Wu, Y.: The bugs framework (BF): a structured approach to express bugs. In: *Proceedings of the IEEE International Conference on Software Quality, Reliability and Security (QRS 2016)*, Vienna, pp. 175–182. IEEE (2016)
3. dos Santos, J.C.A., Favero, E.L.: Practical use of a latent semantic analysis (LSA) model for automatic evaluation of written answers. *J. Braz. Comput. Soc.* **21**(1), 1–21 (2015)
4. Du, D.: Refining traceability links between vulnerability and software component in a vulnerability knowledge graph. In: Mikkonen, T., Klamma, R., Hernández, J. (eds.) *ICWE 2018*. LNCS, vol. 10845, pp. 33–49. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-91662-0_3
5. Fautsch, C., Savoy, J.: Adapting the TF IDF vector-space model to domain specific information retrieval. In: *Proceedings of the 2010 ACM Symposium on Applied Computing (SAC 2010)*, Sierre, pp. 1708–1712. ACM (2010)
6. Franqueira, V.N.L., Tun, T.T., Yu, Y., Wieringa, R., Nuseibeh, B.: Risk and argument: a risk-based argumentation method for practical security. In: *Proceedings of the IEEE 19th International Requirements Engineering Conference (RE 2011)*, Trento, pp. 239–248. IEEE (2011)

7. Gamallo, P., Bordag, S.: Is singular value decomposition useful for word similarity extraction? *Lang. Resour. Eval.* **45**(2), 95–119 (2011)
8. Goseva-Popstojanova, K., Tyo, J.: Experience report: security vulnerability profiles of mission critical software: empirical analysis of security related bug reports. In: *Proceedings of the IEEE 28th International Symposium on Software Reliability Engineering (ISSRE 2017)*, Toulouse, pp. 152–163. IEEE (2017)
9. Hale, M.L., Gamble, R.F.: Semantic hierarchies for extracting, modeling, and connecting compliance requirements in information security control standards. *Requir. Eng.* 1–38 (2018). Published online in December 2017
10. Han, Z., Li, X., Liu, H., Xing, Z., Feng, Z.: DeepWeak: reasoning common software weaknesses via knowledge graph embedding. In: *Proceedings of the IEEE 25th International Conference on Software Analysis, Evolution and Reengineering (SANER 2018)*, Campobasso, pp. 456–466. IEEE (2018)
11. Hussain, S.F., Suryani, A.: On retrieving intelligently plagiarized documents using semantic similarity. *Eng. Appl. Artif. Intell.* **45**, 246–258 (2015)
12. Ibrahim, O.A.S., Landa-Silva, D.: Term frequency with average term occurrences for textual information retrieval. *Soft. Comput.* **20**(8), 3045–3061 (2016)
13. Jimenez, M., Papadakis, M., Traon, Y.L.: An empirical analysis of vulnerabilities in OpenSSL and the Linux Kernel. In: *Proceedings of the 23rd Asia-Pacific Software Engineering Conference (APSEC 2016)*, Hamilton, pp. 105–112. IEEE (2016)
14. Jin, R., Chai, J.Y., Si, L.: Learn to weight terms in information retrieval using category information. In: *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005)*, Bonn, pp. 353–360. ACM (2005)
15. Kang, J., Park, J.H.: A secure-coding and vulnerability check system based on smart-fuzzing and exploit. *Neurocomputing* **256**, 23–34 (2017)
16. Martin, R.A., Barnum, S.: Common weaknesses enumeration (CWE) status update. *ACM SIGAda Ada Lett. Arch.* **XXVII**(1), 88–91 (2008)
17. McManus, J.: SEI CERT Oracle Coding Standard for Java, Carnegie Mellon University, Software Engineering Institute (SEI) (2018). <https://wiki.sei.cmu.edu/confluence/display/java/SEI+CERT+Oracle+Coding+Standard+for+Java>. Accessed May 2018
18. MITRE: Common Weaknesses Enumeration, CWE List Version 3.1, CWE Comprehensive View (2018). <http://cwe.mitre.org/data/csv/2000.csv.zip>. Accessed April 2018
19. MITRE: CWE VIEW: Weaknesses Originally Used by NVD from 2008 to 2016 (2018). <http://cwe.mitre.org/data/definitions/635.html>. Accessed January 2018
20. Mitropoulos, D., Karakoidas, V., Louridas, P., Gousios, G., Spinellis, D.: The bug catalog of the maven ecosystem. In: *Proceedings of the 11th Working Conference on Mining Software Repositories (MSR 2014)*, Hyderabad, pp. 372–375. ACM (2014)
21. Muñoz, F.R., Villalba, L.J.G.: An algorithm to find relationships between web vulnerabilities. *J. Supercomput.* **74**(3), 1061–1089 (2018)
22. Murtaza, S., Khreich, W., Hamou-Lhadj, A., Bener, A.B.: Mining trends and patterns of software vulnerabilities. *J. Syst. Softw.* **117**, 218–228 (2016)
23. NIST: NVD Data Feeds, National Institute of Standards and Technology (NIST) (2018). <https://nvd.nist.gov/vuln/data-feeds>. Accessed April 2018
24. The Natural Language Toolkit (NLTK): NLTK 3.2.5 Documentation (2017). <http://www.nltk.org>. Accessed April 2018
25. Oyetoyan, T.D., Milosheska, B., Grini, M., Soares Cruzes, D.: Myths and facts about static application security testing tools: an action research at Telenor digital. In: Garbajosa, J., Wang, X., Aguiar, A. (eds.) *XP 2018*. LNBIP, vol. 314, pp. 86–103. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-91602-6_6

26. Paik, J.H.: A novel TF-IDF weighting scheme for effective ranking. In: Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2013), Dublin, pp. 343–352. ACM (2013)
27. Peclat, R.N., Ramos, G.N.: Semantic analysis for identifying security concerns in software procurement edicts. *New Gener. Comput.* **36**(1), 21–40 (2018)
28. Porter, M.F.: An algorithm for suffix stripping. *Program* **14**(3), 130–137 (1980)
29. Raemaekers, S., van Deursen, A., Visser, J.: Semantic versioning and impact of breaking changes in the maven repository. *J. Syst. Softw.* **129**, 140–158 (2017)
30. Ruohonen, J.: Classifying web exploits with topic modeling. In: Proceedings of the 28th International Workshop on Database and Expert Systems Applications (DEXA 2017), Lyon, pp. 93–97. IEEE (2017)
31. Ruohonen, J., Rauti, S., Hyrynsalmi, S., Leppänen, V.: Mining social networks of open source CVE coordination. In: Proceedings of the 27th International Workshop on Software Measurement and 12th International Conference on Software Process and Product Measurement (IWSM Mensura 2017), Gothenburg, pp. 176–188. ACM (2017)
32. Snyk Ltd.: Snyk Vulnerability Database (2018). <https://github.com/snyk/vulnerabilitydb>. Accessed April 2018
33. Squire, M.: Data sets describing the circle of life in Ruby hosting, 2003–2016. *Empir. Softw. Eng.* **23**(2), 1123–1152 (2018)
34. Tsipenyuk, K., Chess, B., McGraw, G.: Seven Pernicious Kingdoms: a taxonomy of software security errors. *IEEE Secur. Priv.* **3**(6), 81–84 (2005)
35. Wen, T., Zhang, Y., Wu, Q., Yang, G.: ASVC: an automatic security vulnerability categorization framework based on novel features of vulnerability data. *J. Commun.* **10**(2), 107–116 (2015)
36. Wu, Y., Gandhi, R.A., Siy, H.: Using semantic templates to study vulnerabilities recorded in large software repositories. In: Proceedings of the 2010 ICSE Workshop on Software Engineering for Secure Systems (SESS 2010), Cape Town, pp. 22–28. ACM (2010)



Investigating the Effect of Attributes on User Trust in Social Media

Jamal Al Qundus¹(✉)  and Adrian Paschke^{1,2}(✉)

¹ Computer Science, Freie Universität Berlin, Berlin, Germany
jamal.alqundus@fu-berlin.de, paschke@inf.fu-berlin.de

² Data Analytics Center (DANA), Fraunhofer FOKUS, Berlin, Germany
adrian.paschke@fokus.fraunhofer.de

Abstract. One main challenge in social media is to identify trustworthy information. If we cannot recognize information as trustworthy, that information may become useless or be lost. Opposite, we could consume wrong or fake information - with major consequences. How does a user handle the information provided before consuming it? Are the comments on a post, the author or votes essential for taking such a decision? Are these attributes considered together and which attribute is more important? To answer these questions, we developed a trust model to support knowledge sharing of user content in social media. This trust model is based on the dimensions of stability, quality, and credibility. Each dimension contains metrics (user role, user IQ, votes, etc.) that are important to the user based on data analysis. We present in this paper, an evaluation of the proposed trust model using conjoint analysis (CA) as an evaluation method. The results obtained from 348 responses, validate the trust model. A trust degree translator interprets the content as very trusted, trusted, untrusted, and very untrusted based on the calculated value of trust. Furthermore, the results show a different importance for each dimension: stability 24%, credibility 35% and quality 41%.

Keywords: Social media · Trust · Conjoint analysis

1 Introduction

Users consume information when they have trust in it. One main challenge in social media is how to identify trustworthy information. For instance, relevant information such as storm warning or medical instruction could not be considered by users, if it is not recognized as trustworthy. Usually, users look at properties (e.g. author, reviews, etc.) to take decision to trust the information. However, many questions arise when it comes to which properties are relevant and how important they are for the user to consume this information.

Social media (SM) have many users, which makes it well suitable for examining user activities on the information provided. Therefore, we considered the SM Genius (www.genius.com) as a case study to measure user's willingness to trust

the information provided on this platform. Interactively, users on Genius create annotations that serve as placeholders for the interpretations of texts, especially lyrics and literature. Annotations provide editing functions such as voting, sharing, adding comments, etc. Participation in this platform is described by certain activities. These activities are linked to certain user authorizations (e.g. roles: whitehat, artist, editor, etc.). Based on these authorizations, a user can perform certain activities and earn Intelligence Quotient (IQ)¹, which indicates the experience required for authorization and acceptance of content [1].

The focus of this article is to determine the properties that are important for user trust in the user-generated content environment (i.e. any content generated by the user on Genius that is an annotation, a comment, or a modification).

Existing research [2–5] tackles this problem of trust by verifying the history of the generated content, reputation and algorithms for detecting vandalism. In the contract, we estimate user preferences on content properties that are relevant in making decision to consume (trust) the information. This gives us an idea of how we can present information in social media using a template that helps identify trusted information.

To obtain such a template, the user’s willingness to trust should be measured by (1) simulating a number of templates that include different properties and (2) estimating the user’s choices. Measuring user’s willingness leads to the construction of a trust model that quantifies the value of trust and at the same time embodies in its structure the construction of the required template. This can be evaluated with the use of the conjoint analysis (CA) [6] method, which simulates the decision-making process of consumers when choosing products in real life.

To build our model, we analyzed first, the data collected from Genius based on user activities (e.g. annotation, voting, comments). Then, we select the metrics used in Genius that correspond to user activities (for instance, “annotation” corresponds to “annotation IQ (see footnote 1)”, “vote” to “edit IQ (see footnote 1)”). Finally, we looked for a correlation between the metrics defined in the dimensions of the existing trust models in the literature and those defined in Genius. With this literature review, we classified the metrics into three dimensions namely: stability, credibility and quality. These dimensions are then integrated into our trust model. This trust mode can be used as well for other social media having the same content properties.

Our main emphasis in this paper is to provide a reliable assessment of the selected dimensions and their acceptance by web users in general. This can be conducted by estimating the user choices using a Discrete Choice Conjoint analysis (DCC), which is a form of CA evaluation method.

The paper is structured as follows: Sect. 2 provides a brief overview of relevant works. Section 3 describes the dimensions of the trust model and an illustrative example to compute trust. Section 4 presents the preparation of the survey. Section 5 reports and discusses the findings. Section 6 contains the summary and concludes with proposals for further investigation.

¹ Intelligence Quotient is a counter of points awarded for activities on Genius.

2 Related Work

Dondio et al. propose a Wikipedia Trust Calculator (WTC) consisting of a data retrieval module that contains the required data of an article. A factor-calculator module calculates the confidence factors. A trust evaluator module transfers the numerical confidence value in a natural language declaration using constraints provided by a logic conditions module [7]. This approach refers exclusively to Wikipedia and cannot be transferred into other domains such as social media. In addition, the aim here is to detect vandalism and not trust in our definition. The trust model of Abdul-Rahman and Hailes is based on sociological characteristics. These are trusted beliefs between agents based on experience (of trust) and reputation (came from a recommended agent) combined to build a trusted opinion to make a decision about interacting with the information provided [8]. This approach combines reputation and agents together to build trust. These metrics are not available in our domain.

These two works are closely linked to our work. Our approach is comparable to that of Dondio et al., and in particular their work has inspired the calculation of the stability dimension. The authors build the stability based on the change (defined as edit) in the text length of an article. While the stability dimension in our work is built on any type of editing (vote, suggestion, creation, etc.) to an annotation. In addition, we have adapted the trust degree translator from Abdul-Rahman and Hailes [8]. Based on data base analysis, we manually defined its constraints, that are used for interpretation of the numeric trust value into a human-readable language.

Cho et al. investigated trust in different contexts and discussed trust from various aspects. The authors employed a survey to investigate social trust from interactions/networks, and captured of quality of service (QoS) and quality of information (QoI) depending on a relation between two entities (trustor and trustee) [9]. However, this is an examination of trust and reputation systems in online services [10]. These works and others must be placed in a restricted domain to find a relationship between the communicating entities. However, this is not always possible in an unlimited domain like social media. Different from the prior works, this survey brings together the aspects of trust from a specific, but open domain and let entities evaluate them from outside of this domain. Additionally, we suggest an advanced equation for measuring trust.

3 Trust Model Construction

We conducted investigation on the social media Genius as a case study to build our trust model, as follows: First, we analyzed the data collected from Genius based on user activities (e.g. annotation, voting, comments). Then, we select the metrics used in Genius that correspond to user activities (for instance, “annotation” corresponds to “annotation IQ (see footnote 1)”, “vote” to “edit IQ (see footnote 1)”). Furthermore, other metrics are considered such as the metric “author role” (e.g. editor, or whitehat) which is not considered as an activity but

as important metric in Genius (for more detail see Genius technical report [11]). Finally, we looked for a correlation between the metrics defined in the dimensions of the existing trust models in the literature [7, 12–15] and those defined in Genius. For instance, the dimension credibility comprised the metrics set: user role, user IQ, attribution and annotation IQ. With this literature review, we classified the metrics into three dimensions namely: stability, credibility and quality. These dimensions are then integrated into our trust model.

The trust model classifies annotations into four classes [8] called trust degrees illustrated in Table 1. These classes were obtained by the database analysis based on the Empirical Cumulative Distribution Function (ECDF) [16]. Next, we present the formulas used to compute trust.

Table 1. Trust degree translator

Trust degree	Percentage	Edits number	Edits IQ	User IQ
vt	25%	>5	>35	>1000
t	43.75%	2 to 5	5 to 35	0 to 1000
u	6.25%	0 to 2	0 to 5	−100 to 0
vu	37.5%	<0	<0	<−100

Table 1 illustrates the trust degree translator for the interpretation of the individual statement trust classes. vt = very trusted, t = trusted, u = untrusted, vu = very untrusted. Percentage results from the ECDF applied data analysis observed on Genius, and illustrates the distribution of statements’ (annotations) trust classes in the data set.

Trust is calculated based on the dimensions stability, credibility and quality. The trust value obtained is then interpreted by the trust degree translator to identify the annotation class: very trusted, trusted, untrusted and very untrusted.

$$trust = \alpha \times stability + \beta \times credibility + \gamma \times quality \tag{1}$$

α , β and γ are the importance factors of each dimension.

- Stability (S) is represented by the annotation edits’ distance (see Eq. 3), which represents several content modifications in a time interval. For example, the interval could be between the initial time stamp of the annotation and the current time. $E(t)$ (see Eq. 2) specifies the number of edits at the time stamp t .

$$\{E(t) : t \rightarrow \Phi | E : edits\ function, t : timestamp, \Phi \in \mathbb{Z}\} \tag{2}$$

Here, \mathbb{Z} is a set of all integers.

$$\{S = \sum_{t=t_0}^{t=p} E(t) | t \ \& \ p : timestamp, E : edits\ function\} \tag{3}$$

- Credibility (C) refers to correctness, authorship and the depth in meaning of information. We consider the type of user activity on an annotation as `editsType`. There are complex activities (e.g. annotation creation) that require agility from the user during execution. We should note that the ranks of these complex activities in Genius are higher than so-called simple activities (e.g. annotation voting). In addition, we applied a User Credibility Correction Factor (UCCF), which is calculated based on the users role², user IQ³ and attribution⁴ for modifying the credibility status.

$$C = \frac{UCCF + editsTypes}{2} \tag{4}$$

where $UCCF = \text{foreach author } a: a.\text{attribution} \times a.\text{rolePower}$. And $a.\text{rolePower} = a.\text{role} \times a.\text{roleFactor} \times a.\text{IQ}$.

- Quality (Q) is calculated exactly like C , except for the restriction to n -top active users⁵, who are ordered based on their attribution.

$$Q = \frac{UCCF' + editsTypes'}{2} \tag{5}$$

Before presenting an example that illustrates how the trust is calculated, we introduce the terms utility and importance. The utility or part-worth is a measure of how important an attribute level is for a trade-off decision by the user. Whereas the relative importance of an attribute is the delta percentage compared to all utilities. Each time a respondent makes a choice, an accumulator compiles the numbers. These numbers indicate how often a level has been selected. The algorithm used for calculating the utilities is logit model [17] combined with a Nelder-Mead Simplex algorithm [18].

The utility (U) of an attribute level (l) is calculated by the distance from its selected value (S) to the minimum (min) level selected value of the same attribute (k) as shown in the following Eq. 6:

$$U_{al} = S_{kl} - S_{kl_{min}} \tag{6}$$

The importance (I) of an attribute (k) is calculated by the difference between the maximum selected number ($S_{kl_{max}}$) and the minimum selected number ($S_{kl_{min}}$) of this attribute level, divided by the sum of such differences over all attributes $A = \{a_1, a_2, \dots, k, \dots, a_n\}$ (see Eq. 7).

$$I_k = \frac{S_{kl_{max}} - S_{kl_{min}}}{\sum_{a=1}^{a=n} (S_{al_{max}} - S_{al_{min}})} \times 100\% \tag{7}$$

² Genius members have roles that differ in the permissions assigned to them. The IQ numbers earned for an activity also depend on the role type.

³ The overall earned IQ count of a user.

⁴ Attribution is the percentage of edits made by a user.

⁵ n is a number that the observer can freely select.

3.1 Illustrative Example

In this section, we present an example that illustrates how we apply the results of CA in evaluating the dimensions and as a consequence of that, computing trust of an annotation.

A discrete choice conjoint analysis (DCC) provides a quantity⁶ of tasks. A task consists of a set of concepts, each concept represents a certain number of attribute levels. Users select a concept that they would trust in reality. Figure 1 illustrates a task used in our conjoint design. Each task concept contains the attributes Comments, Reader Rating and Author Rating and their randomly generated levels values. The attributes act in place of the trust dimensions.

Table 2 provides an example of the collected numbers of user trade-offs. The columns Level, Selected and Offered are predefined. We only explain the calculations for one attribute Comments, since the calculations for the attributes Reader Rating and Author Rating are analog.

Utility. Equation 6 is applied as follows: $a = \text{Comment}$, $l = \text{level}$, $S_{al_{max}} = \text{maximum level selected value}$, $S_{al_{min}} = \text{minimum level selected value}$, $U_{al} = 0$

$$\begin{array}{l|l} l = 0 & l = 2 \\ \hline U_{al} = 0 - 0 = 0 & U_{al} = 2 - 0 = 2 \\ \hline l = 5 & l = 10 \\ \hline U_{al} = 5 - 0 = 5 & U_{al} = 10 - 0 = 10 \end{array}$$

Relative Importance (see Eq. 7) is applied as follows: $\sum_{a=1}^A (S_{al_{max}} - S_{al_{min}}) = (61 - 2) + (72 - 5) + (80 - 1) = 205$
 $k = \text{Comments}$, $S_{kl_{max}} = 61$, $S_{kl_{min}} = 2$.⁷

$$I_k \in \{\alpha, \beta, \gamma\} = \frac{61 - 2}{205} \times 100\% \approx 29\%$$

Trust (ω) can be now calculated based on the dimensions' Eqs. 3, 4, 5 and 1 as follows:

Let the number of edits of an annotation be 50 from the creation time to the current time. From these edits are 10 complex edits (CE) (e.g. content modification) and 40 simple edits (SE) (e.g. voting). The edits IQ equals 30. The users' roles are (editor (25), whitehat (3) and staff (38)), the sum total of users' (authors') IQ equals 160 (10, 30 and 120, respectively) and their attributions (70% with 2 CE and 7 SE, 28% with 7 CE and 30 SE and 2% with 1 CE and 3 SE, respectively). The $n = 2$ for the n-top active user. Based on this input the stability, credibility and quality can be calculated, as follows:

$$\begin{aligned} \text{Stability} &= 50 \\ \text{Credibility} &= \frac{7.285 + 7.5}{2} = 7.392 \end{aligned}$$

⁶ Number depends on the conjoint analysis design.

⁷ $\beta = 0.33$, $\gamma = 0.39$.

where $UCCF = \text{foreach author } a: a.\text{attribution} \times a.\text{rolePower}$

And $a.\text{rolePower} = a.\text{role} \times a.\text{roleFactor} \times a.IQ$

$$UCCF = 0.7 \times (25 \times 0.025 \times 10) + 0.28 \times (3 \times 0.025 \times 30) + 0.02 \times (38 \times 0.025 \times 120) = 7.285$$

$$\text{EditsTypes} = \text{edit IQ} \times \frac{CE}{SE} = 30 \times \frac{10}{40} = 7.5$$

$$\text{Quality}_{n=2} = \frac{5.005 + 7.297}{2} = 6.151$$

$$\text{where } UCFF' = UCCF - 0.02 \times (38 \times 0.025 \times 120) = 5.005$$

$$\text{EditsTypes}' = \text{editIQ} \times \frac{CE'}{SE'} = 30 \times \frac{9}{37} = 7.297$$

$$\text{Trust} = 0.29 \times 50 + 0.33 \times 7.392 + 0.39 \times 6.151 = 19.338$$

Trust degree translator interprets the trust value $\omega \geq 15$ as very trusted, $15 > \omega \geq 13.5$ as trusted, $13.5 > \omega \geq 12$ as untrusted and $\omega < 12$ as very untrusted.

Comments ▶	0	+2	+10	0
Reader Rating ▶	+70	+5	0	+5
Author Rating ▶	+2000	0	-100	-100
	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>	<input type="radio"/>

Fig. 1. Presents a task that illustrates one step in the conjoint analysis profile. This task is displayed to the respondents to make a trade-off on the provided concepts. A concept consists of attributes (Comments, Reader Rating and Author Rating) and randomly generated levels values combined to alternatives.

4 Methodology

Our approach applies DCC as follows: Using e-mail, we announced a link to the online survey in Arabic, English and German. In the DCC we described the attributes (see Table 3): (1) Comments as “a number that indicates improvement edits created by other readers”, (2) Reader Rating as “a number of other readers’ approval” and (3) Author Rating as “a number of voting that the author earned for his activities in the social network”. We also stated that “The greater the number, the greater the satisfaction”. “Each number represents the sum of negative and positive assertions. There are comments that were rated negatively and other comments that were rated positively by readers. Negatives were marked with minus and positives with plus numbers. Subsequently, both numbers were summed up. This applies to all properties”.

Table 2. Attributes and levels calculation example

Attribute	Level	Selected	Utility	Offered	rel.Importance
Comment	0	2	0	5%	29%
	2	33	2	24%	
	5	44	5	31%	
	10	61	10	40%	
Reader Rating	0	5	0	4%	33%
	10	24	10	17%	
	30	39	30	28%	
	70	72	70	51%	
Author Rating	-100	1	0	5%	39%
	0	7	100	7%	
	1000	52	1100	37%	
	2000	80	2100	51%	

Table 2 gives an example of calculating the attributes relative importance. Attribute = the property of a statement, Level = one possible value an attribute can take, Selected = the number of selection frequency of respondents, Utility = level’s important to the respondents choice decision, Offered = display frequency to the respondents, relative Importance = measure of how preferred an attribute is to the respondents choice decision.

Due to the amount of information in a full-profile design ($4_{levels}^{3attributes}$ makes 64 alternatives), a compiled questionnaire would become too extensive. Therefore, we decided to use a fractional factorial-design within the factor $\frac{1}{2}$. The conducted DCC consists of 32 concepts and the respondents were also given four alternatives to choose from in each task.

5 Results and Discussion

Johnson and Orme recommend a rule-of-thumb for the minimum sample sizes for CBC modeling ($\frac{nta}{c} \geq 500$) [19]. Where n : the respondents number, t : the tasks number, a : the number of alternatives per task and c : the highest level number over all attribute. Accordingly, our questionnaire response has a satisfactory number of participants ⁸. The questionnaire received responses that were distributed over 12 countries, which supports the results to be more significant and we experienced responses from a widely distributed audience.

The areas of emphasis of the individual statements met all our expectations and are in line with the proposed theoretical trust model. The selected properties included in the dimensions is important to the users by making a decision to trust the information provided. Table 3 presents the best profile (levels: +10, +70 and +2000) and the worst profile (levels: 0, 0, -100). These two results are

⁸ 348 responses with a completion rate of 40.65%.

interpreted as very trusted and the worst profile as very untrusted respectively by the trust degrees translator. As a reminder, the values of levels in the questionnaire and the numbers applied into the trust degree translator are obtained from the analysis of the database collected from Genius. This table provides information regarding the importance of the attributes and the utilities of the levels. The analysis of the respondents' choice decisions has been conducted by the authors of this paper. This analysis provides a summary from which we can conclude the following:

- I *None of the attributes was excluded from the choice decision.* All defined dimensions have significance in the proposed trust model. Importance of stability, credibility and quality are 24%, 35% and 41% respectively (see Table 3). If a dimension had turned out less significance, this would mean that it has no relevancy for the model and should not be considered. This result is an indicator that the model is accepted and confirmed by the evaluators.
- II *None of the attributes has an extremely high value.* None of the dimensions alone make up the model. That would mean that we can neglect all other dimensions and focus exclusively on one. In addition, it would indicate that other components or dimensions are essential for trust in this context and our model did not consider them.
- III *The importance of the attributes is about equally dispersed.* This confirms our preliminary consideration when we weighted the dimensions nearly equal (see Eq. 1). Nevertheless, it was not possible to determine a more precise weighting for the trust calculation until this evaluation. In which, the respondents printed out their importance weight distribution of each dimension. Applying this weighting into the equation we can improve the calculation with coefficients that are derived from the percentages of the individual domain rankings.
- IV *There is a distinct subdivision of the utilities of an attribute into four parts.* The resulted subdivision is in line with the classes of the trust degrees. The levels of an attribute exhibit clear differences on how often they have been selected (see Table 3). We can reclassify the distribution of the levels utilities of individual attribute in the classes. This applies to all attributes.
- V *The distinct subdivisions agree over all attributes respectively.* This corresponds with the prior statement and represents an extension that the subdivision of the levels utilities on an attribute stage continues to move across all attributes levels and in the same order (see Table 3). If we number the trust classes and the levels of each attribute consecutively, we realize that the first level of each attribute can be assigned to the first trust class (very untrusted) and the second attributes levels can be assigned to the second class (untrusted) and so forth. For instance, The utility (-0.90) of the first level (-100) of the attribute author rating, the utility (-0.82) of the first level (0) of the attribute reader rating and the utility (-0.67) of the first level (0) of the attribute comments, present together a concept that was the least rated once by the respondents. This concept is classified as very untrusted by our trust model. This is applied for the second, third and fourth levels of each attribute respectively.

Table 3. Attributes importance and levels utilities results

Author Rating (quality)		Reader Rating (credibility)		Comments (stability)	
40.85%		34.8%		24.35%	
Level	Utility	Level	Utility	Level	Utility
-100	-0.90	0	-0.82	0	-0.67
0	-0.39	+5	-0.18	+2	-0.04
+1000	+0.44	+30	+0.29	+5	+0.24
+2000	+0.86	+70	+0.71	+10	+0.47

Table 3 gives the attributes importance, the levels and their utilities as the measure for the respondents’ preferences. This refines the weights of the attributes acting in place of the trust model dimensions.

6 Conclusion and Future Work

This work carried out an evaluation of the trust model using a conjoint analysis to determine the respondents’ choice. The information gained from the results confirm our trust model. In its structure is the required template included that helps identify trusted information according to user estimated preferences. This model consists of three dimensions: stability, credibility and quality which were adopted from the literature and the Genius platform. This model is intended to support the development of successful applications.

The used logit model provides effective analysis and convincing results. Nevertheless, an analysis using the hierarchical bayes would allow us to take a fresh look at the selections of individuals. With hierarchical bayes we would be able to trace the history of respondent’s decisions and possibly expose some more details about their behavior. After the respondents’ conclusions about the developed model have been drawn, it is necessary to investigate the database from which the model was created. A clustering procedure should be used to identify which text content belong to which trust category and why. The content-related parameters are to be investigated.

Acknowledgment. The work supported in this paper was partially supported by Data4Water H2020 project.



References

1. Al Qundus, J.: Generating trust in collaborative annotation environments. In: Proceedings of the 12th International Symposium on Open Collaboration Companion, p. 3. ACM, August 2016
2. Li, H., Jiang, J., Wu, M.: The effects of trust assurances on consumers initial online trust: a two-stage decision-making process perspective. *Int. J. Inf. Manag.* **34**(3), 395–405 (2014)
3. Miltgen, C.L., Smith, H.J.: Exploring information privacy regulation, risks, trust, and behavior. *Inf. Manag.* **52**(6), 741–759 (2015)

4. Liu, H., et al.: Predicting trusts among users of online communities: an Epinions case study. In: Proceedings of the 9th ACM Conference on Electronic Commerce, pp. 310–319. ACM, July 2008
5. Adler, B.T., de Alfaro, L., Mola-Velasco, S.M., Rosso, P., West, A.G.: Wikipedia vandalism detection: combining natural language, metadata, and reputation features. In: Gelbukh, A. (ed.) CICLing 2011 Part II. LNCS, vol. 6609, pp. 277–288. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-19437-5_23
6. Luce, R.D., Tukey, J.W.: Simultaneous conjoint measurement: a new type of fundamental measurement. *J. Math. Psychol.* **1**(1), 1–27 (1964)
7. Dondio, P., Barrett, S., Weber, S., Seigneur, J.M.: Extracting trust from domain analysis: a case study on the Wikipedia project. In: Yang, L.T., Jin, H., Ma, J., Ungerer, T. (eds.) ATC 2006. LNCS, vol. 4158, pp. 362–373. Springer, Heidelberg (2006). https://doi.org/10.1007/11839569_35
8. Abdul-Rahman, A., Hales, S.: Supporting trust in virtual communities. In: Proceedings of the 33rd Annual Hawaii International Conference on System Sciences, p. 9–pp. IEEE, January 2000
9. Cho, J.H., Chan, K., Adali, S.: A survey on trust modeling. *ACM Comput. Surv. (CSUR)* **48**(2), 28 (2015)
10. Jsang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. *Decis. Support Syst.* **43**(2), 618–644 (2007)
11. Al Qundus, J.: Technical analysis of the social media platform genius. Technical Report. Freie Universität Berlin (2018). http://edocs.fu-berlin.de/docs/receive/FUDOCS_document_000000029280
12. Metzger, M.J., Flanagin, A.J., Eyal, K., Lemus, D.R., McCann, R.M.: Credibility for the 21st century: integrating perspectives on source, message, and media credibility in the contemporary media environment. *Ann. Int. Commun. Assoc.* **27**(1), 293–335 (2003)
13. Pranata, I., Susilo, W.: Are the most popular users always trustworthy? The case of Yelp. *Electron. Commer. Res. Appl.* **20**, 30–41 (2016)
14. Warncke-Wang, M., Cosley, D. and Riedl, J.: Tell me more: an actionable quality model for Wikipedia. In: Proceedings of the 9th International Symposium on Open Collaboration, p. 8. ACM, August 2013
15. Fogg, B.J., et al.: What makes Web sites credible?: a report on a large quantitative study. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 61–68. ACM, March 2001
16. Castro, R.: The empirical distribution function and the histogram. Lecture Notes, 2WS17-Advanced Statistics. Department of Mathematics, Eindhoven University of Technology (2015)
17. Louviere, J.J., Woodworth, G.: Design and analysis of simulated consumer choice or allocation experiments: an approach based on aggregate data. *J. Mark. Res.* **20**, 350–367 (1983)
18. Nelder, J.A., Mead, R.: A simplex method for function minimization. *Comput. J.* **7**(4), 308–313 (1965)
19. Johnson, R.M., Orme, B.K.: How many questions should you ask in choice-based conjoint studies. In: *Art Forum*, Beaver Creek (1996)



Analysing Author Self-citations in Computer Science Publications

Tobias Milz¹(✉)  and Christin Seifert² 

¹ University of Passau, 94030 Passau, Germany
tobias.milz@uni-passau.de

² University of Twente, PO BOX 217, 7500 AE Enschede, The Netherlands
c.seifert@utwente.nl

Abstract. In scientific papers, citations refer to relevant previous work in order to underline the current line of argumentation, compare to other work and/or avoid repetition in writing. Self-citations, e.g. authors citing own previous work might have the same motivation but have also gained negative attention w.r.t. unjustified improvement of scientific performance indicators. Previous studies on self-citations do not provide a detailed analysis in the domain of computer science. In this work, we analyse the prevalence of self-citations in the DBLP, a digital library for computer science. We find, that approx. 10% of all citations are self-citations, while the rates vary with year after publication and the position of the author in the list as well as with the gender of the lead author. Further, we find that C-ranked venues have the highest incoming self-citation rate, while the outgoing rate is stable across all ranks.

Keywords: Citations · Self-citations · Analysis · DBLP · CORE

1 Introduction

Scientific work is iterative, findings and discoveries usually lead to new questions waiting to be answered [7]. Consequently, scientific publications, which are the communication of the scientific process and its findings are also iterative and built upon each other. This means, that self-citations, e.g. authors citing their own previous work in publications are an inherent property of the scientific process [10, 11]. They, however, also gained a negative connotation due to their abuse regarding performance indicators of scientists [2, 6, 10, 11, 20]. As a result, scores for detecting likely unfair self-citations have been developed e.g. [3]. In this paper, we characterise the prevalence of self-citations in the domain of computer science according to different factors such as publication year, author gender, conference/journal rank and author positions. We base our analysis on the DBLP computer science bibliography [17], a major bibliographic reference for computer science papers containing more than 3 million publication records as of October 2017. Additionally, we use the conference ranking provided by the Computing

Research and Education Association of Australasia (CORE rankings)¹ to enrich the DBLP data.

Previous work identified different reasons for self-citations (for a complete list see [5]). For instance, in scientific fields with incremental contributions, (own) previous work is cited to avoid duplicate material. Also, in small research areas with a limited number of publication outlets and researchers, citing one’s own work is highly prevalent. Thus, we want to emphasise that we provide a quantitative analysis of self-citations, but we do not make an assessment whether a self-citation is justified or not.

The next section outlines some already available statistics, which are mostly available for domains other than computer science. Those statistics are impractical to compare, because of different quantifications. We define our notions in Sect. 3 and continue with the description of our approach followed by the experiments, which are ought to answer questions similar to “how does factor X influence self-citation rates?”

2 Related Work

Different studies have analysed *self-citation rates and their influencing factors*. The most comprehensive study w.r.t to time coverage comprised approx. 1.8 million JSTOR papers from the period of 1779 to 2011 and found an average self-citation rate of 23% (counting outgoing author self-citations) [12]. The only study explicitly covering computer science found a rate of 24% of incoming author self-citations in Norway [1]. Self-citation rates have been found to decrease over time (e.g. [4]), and vary across countries [21, 22] and research fields [8, 12, 21, 22]. Also, collaboration was identified as an influencing factor (e.g. [9, 15, 18]), albeit it seems only important whether papers are single-author or multi-author papers and not how many authors collaborated [9]. While age does not seem to have an influence on self-citation rates [4] authors’ gender has a clear impact. According to [8, 12] men cite themselves more often. Ghiasi et al. [8] found a gender gap in self-citation behaviour of approx. 35% (with variance across scientific fields and time). The quality of publication venue has also been identified as influencing factor in a small-scale study (643 paper) in the domain of ecology: self-citation rate increased with the impact factor of journals [15]. Apart from the average citation rate of 24% in Norway [1] there is no analysis available for the domain of computer science. In this paper, we analyse the computer science bibliography to this end and compare the findings with results from previous studies in different domains.

3 Problem Statement

Following previous work [5, 9] we distinguish between author and paper self-citations as follows: Let s and t be papers with s citing t and let $A(s)$ and $A(t)$

¹ www.core.edu.au, accessed 2018-03-02.

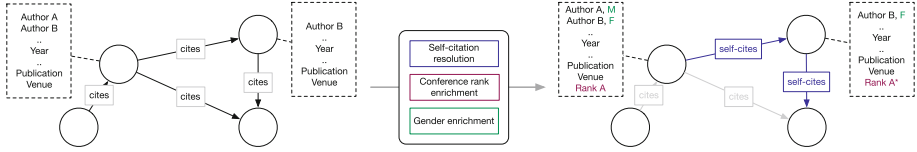


Fig. 1. Overview of the graph structures before and after self-citation resolution and enrichment. Changes are colored.

be the respective sets of authors. A *paper self-citation* scp is any citation for which the author sets of both papers overlap, i.e., $A(s) \cap A(t) \neq \emptyset$.

Let s and t be papers with s citing t and let a be an arbitrary author of paper s . A citation is an *author self-citation* sca if author a is also in the author list of paper t , i.e., $a \in A(s) \cap A(t)$. Author self-citations are also termed authorship self-citations [12] or self-citations at micro level [9]. We further distinguish between incoming and outgoing self-citations, also referred to as synchronous and asynchronous self-citations [14]. We define sc^{in} as the number of self-citations a paper receives and sc^{out} as the number of self-citations in its reference list. Further, we define self-citation rates as the number of self-citations normalized by the number of citations. Thus, $sc_r^{in} = \frac{sc^{in}}{in_G}$ and $sc_r^{out} = \frac{sc^{out}}{out_G}$, with in_G being the in-degree and out_G being the out-degree of the respective node in the citation graph G . Note, that both definitions can be applied to author self-citations sca in a similar fashion.

The goal of this paper is to analyse the occurrences of self-citations with respect to different characteristics of papers (publication year, number of authors, conference/journal rank) and authors (gender, position in author list).

4 Approach

For characterizing self-citations, we use the DBLP citation graph, identify self-citations, enrich the papers' metadata with ranking information from the CORE database and assign a gender attribute to the authors.

An overview of the procedure is depicted in Fig. 1. On the left, the structure of the DBLP citation graph is shown. The graph $G = (V, E)$ consists of nodes $v \in V$ representing papers, and directed edges $e \in E$ representing the “cite”-relationship. An edge (u, v) means, that paper u cites paper v . For all nodes, the following metadata is available: paper title, author list, publication venue and year². The graph is directed and in general acyclic, however, the latter property can be violated, as some accepted publications might be referenced before they are published. After the identification of self-citations and the enrichment of conference/journal ranks we obtain a graph $G' = (V, E')$ with $E' \subset E$ and $e' \in E'$ representing the “self-cite”-relation. Additionally, the metadata for each node with conference/journal information either contains the corresponding rank

² Additional metadata is available, but not relevant for this work.

or “unknown” if the publication venue cannot be found in the CORE database (cf. Fig. 1, right). Importing this graph into a graph-based database such as Neo4j allows easy extraction of both, paper self-citations and author self-citations according to the definitions in Sect. 3.

4.1 Author Name Disambiguation

A core problem of citation analysis is to identify the person an author’s name is referring to. Name homography, i.e., two equal strings referring to different persons, leads to mixed citation errors, while name variability, i.e. two different strings referring to the same person, leads to split citation errors [16]. The DBLP database uses a semiautomatic approach including feedback collection from the scientific community to author name disambiguation [19]. While 100% accuracy cannot be ensured, especially since >1000 publications are added each day [19], we consider the remaining errors to be insignificant w.r.t. aggregated statistics, however, this assumption remains to be proven.

4.2 Rank Enrichment

Most publications in the DBLP database contain the name of their publishing venue (conference/journal), which is used to determine the rank of the publication, based on the year it was published and the rating the corresponding venue had at that time. In order to match the correct venue from the CORE database, a string comparison method is used that takes abbreviations and small string differences into account. We found that in almost all non-trivial cases the DBLP database uses very similar, but smaller sub-strings of the actual venue name taken from the CORE database. As it is very important that these non-trivial matches do not add any false positives, a match is found only if each word of the sub-string is present and they account for 70% of the words of the actual name. Out of all 3,079,007 papers, 506,699 did not include any information about their publishing venue. From the remaining publications, a CORE conference rank was assigned to 437,613 (17.01%) (including the rank “Unranked”).

4.3 Gender Enrichment

In order to determine the gender of an author, we match the author’s first name (given name) to country-specific name lists following the approach from [13]. First, all authors were matched to the list from the US Census³, which contains 1219 male names and 4275 female names. Ambiguous names (names that appear in both lists) were labelled as unisex unless they are 10 times more frequently used for one gender. If an author’s name is not present in the US census list, then it is matched with a comprehensive list of international names taken from Wikipedia and two popular baby name databases⁴. Combining both sets we

³ www2.census.gov/topics/genealogy/1990surnames/, accessed 2018-04-04.

⁴ www.rq.gov.qc.ca/en, www.babycenter.com/baby-names, accessed 2018-04-04.

Table 1. Data set after enrichment. Papers - P, Authors - A, binary gender - G (M/F)

# P	# A	# P w Rank	# A w G(M/F)	Time period
3,079,007	1,766,540	440,356	954,705	1936–2018
(100%)	(100%)	(14.30%)	(54.04%)	

gathered 223,428 international given names and their associated gender, including the unisex label for ambiguous names. After our enrichment we identified 266,584 (15.09%) female authors, 688,121 (38.95%) male authors and 287,784 (16.29%) unisex authors in our dataset. 524,051 (29.67%) author names could not be found in any list and their gender remained unknown. 154,497 (29.48%) of these names contained abbreviated first names or only initials. In total, 54% of all author names and 59% of non-abbreviated names could be assigned a binary gender. This is comparable to other work, where 56% of all authors could be assigned a binary gender [12]. Table 1 summarises the data set after enrichment.

5 Experiments

The leading questions for the experiments were:

- Q-C:*** How does the number of self-citations relate to the total number of citations (C)?
- Q-Y:*** How do self-citations depend on publication year (Y), and age (the year after publication)?
- Q-R:*** How are self-citations influenced by venue rank (R)?
- Q-A:*** How does the number of authors (A) relate to self-citations?
- Q-P:*** Do first or last authors have more self-citations, i.e., what is the influence of author position (P)?
- Q-G:*** Do self-citations differ across genders (G)?

We expect the number of self-citations to increase when the number of citations increases (*Q-C*), but the self-citation rate sc_r to be decreasing with the number of citations as found in other studies, e.g. [1]. For question *Q-Y* we expect an increase of the self-citation rates over time for two reasons: First, with the increasing complexity of the scientific discoveries research becomes more incremental. Second, the academic culture has changed over years and emphasizes citation counts and performance metrics over publications leading to phenomena like least-publishable units and publish-and-perish. With respect to rank (*Q-R*) we do not expect any differences in self-citations rates. On the one hand, lower ranked conferences are used to publish minor improvements and adaptations and therefore boost self-citations, on the other hand, higher ranked conferences publish larger, more comprehensive bodies of scientific work, which might also result in more self-citations. We expect multi-author papers to have a higher self-citation rate than single-author papers, while the number of authors does not have much influence as has been found in other corpora [9]. Further, we

expect last authors to have more self-citations than first or middle authors (Q - P) because in computer science last authors are usually supervisors, they are more senior and have published more papers themselves. For the gender-aspect of self-citations (Q - Y) we expect a higher self-citation rate for male authors than for females as indicated by previous work for other research fields [8, 12].

6 Results

In this section we show the results of the analysis organised by the previously introduced experiment questions.

6.1 Self-citations by Total Number of Citations (Q-C)

In the corpus, 18.67% of all papers do not have any outgoing citations, 51.56% do not have any outgoing and 60.28% do not have any incoming self-citations. Figures 2 and 3 provide an overview of the relation between citations and self-citations. Both, the number of outgoing and incoming self-citations increases with the number of citations, while the outgoing ones show a steeper increase (see Fig. 2). On average, the rate of outgoing and incoming self-citations for papers with at most 50 outgoing citations is 14.16% and 14.75% respectively (9.44% and 11.40% for all papers), while both rates are not constant over the number of citations (cmp. Fig. 3). These numbers are considerably less than the numbers reported in a previous study (24% incoming), which included only 283 computer science papers from Norway [1]. Similar research on other domains has shown, that the number of outgoing self-citations varies significantly depending on the research field (e.g. 6.3% for law and 12.3% for math) [12].

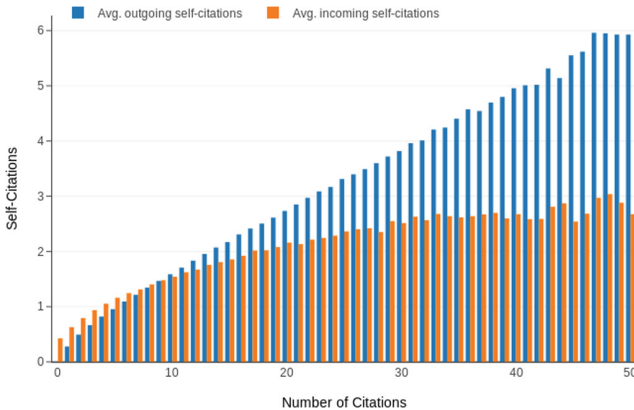


Fig. 2. Relation of number of citations and self-citation rate.

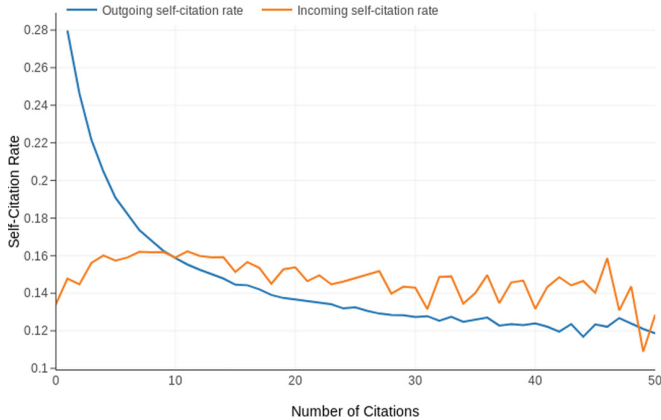


Fig. 3. Relation of self-citations and self-citation rate.

6.2 Self-citations by Year (Q-Y)

From the 1960s onwards the rate of outgoing self-citations remains relatively constant over time. This period covers most of the publications in the database (cmp. Fig. 4). On the contrary, the rate of incoming self-citations started to increase in approx. 2000. As Fig. 5 shows, papers receive most citations up to 10 years after publication, and the highest rates of self-citations in the first years, which is in line with previous observations in other domains [1, 9]. Surprisingly, we found papers that were cited before they were published. We investigated those papers in more detail and found that this is due to citations made to books or collections containing older publications. The DBLP database seems to misidentify very few of these publications as recent works, which however only affects 0.7% of all citations.

6.3 Self-citations by Rank (Q-R) and Author Gender (Q-G)

The self-citation rates with respect to rank are shown in Fig. 6, “Other” aggregates the ranks *New*, *National:Spain*, *L* and *Regional*. The outgoing citation rate is relatively stable across all ranks, but the incoming self-citation rate varies, being 11.69% for A* conferences and 26.35% for C conferences. Figure 7 illustrates the outgoing and incoming self-citation rates with a first author of a specific gender. Publications with a male first author tend to have significantly more outgoing self-citations than papers with female lead authors. However, publications with a female first author receive more incoming self-citations. This observation is in line with findings in domains [9].

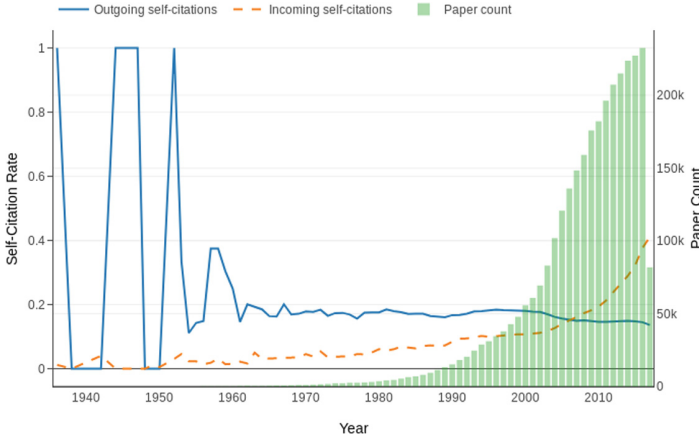


Fig. 4. Relation of number of self-citations and absolute time.

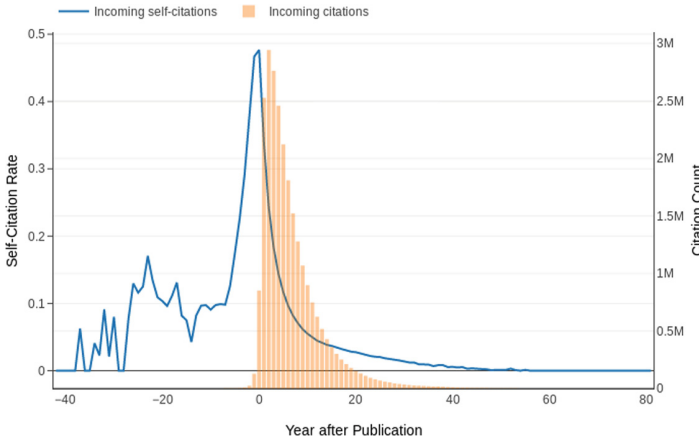


Fig. 5. Relation of number of self-citations and year after publication.

6.4 Self-citations and Authors (Q-A and Q-P)

Figure 8 depicts the relation between the number of authors and the self-citation rates, showing papers with at most 10 authors (99.5% of all papers). As a tendency, the outgoing and incoming self-citation rate increases with the number of authors (denotes n_a). The Pearson correlation coefficients in regards to self-citations are $\rho(n_a, sc_r^{out}) = 0.89$, $\rho(n_a, sc_r^{in}) = 0.81$ and $\rho(n_a, c_r^{out}) = 0.50$, $\rho(n_a, c_r^{in}) = 0.48$ regarding the total number of citations (normalized by the number of papers). The general tendency is in line with previous studies in other domains [9], but we additionally observe a correlation between number of authors and (self-) citation rates.

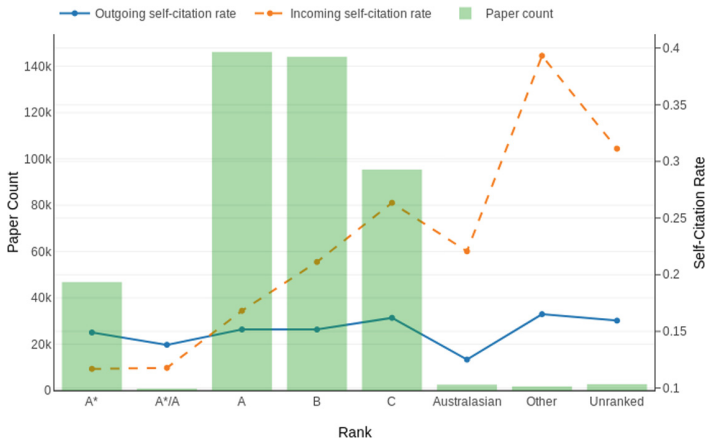


Fig. 6. Relation of venue rank and self-citation rate.

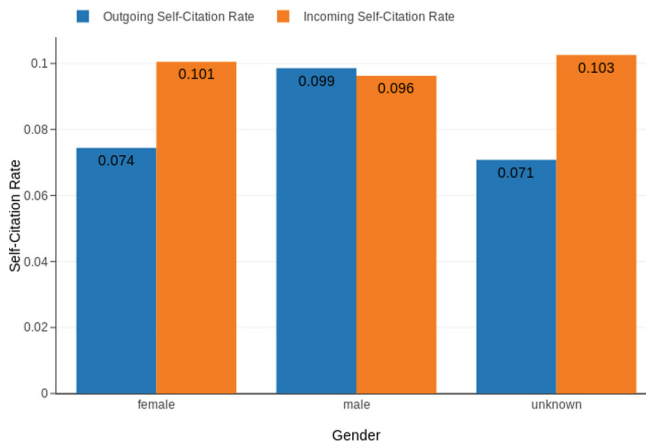


Fig. 7. Relation of gender and self-citation rate.

Figure 9 depicts the relation of author position and self-citation rate for papers with at least one outgoing self-citation. We depict author self-citations w.r.t. to the total number of outgoing citations (blue bars) and the total number of outgoing self-citations (orange bars) in the corresponding subset of the data. These subsets contain all single-author papers (condition “Single”) and all papers with at least two authors (conditions “First” and “Last”). From all citations in single-author papers, 14.8% are self-citations. From all citations in multi-author papers, 7.7% are self-citations that include the first author and 9.1% that include the last author. With this, the first author is part of 50% of all self-citations in multi-author papers, while the last author is involved in 60% of all self-citations. These values do not sum up to 100% as both authors can be part of the same self-citation. This means single authors do more self-citations than first or last authors, but last authors have more self-citations than first authors.



Fig. 8. Relation of number of authors and self-citation rate.

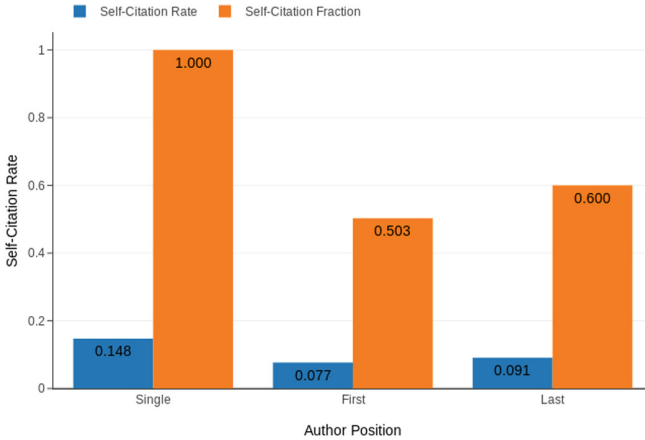


Fig. 9. Relation of author position and self-citation rate. Normalized by the number of citations in single and multi-author papers (blue) and by the number of self-citations (orange). (Color figure online)

7 Summary and Future Work

In this paper, we analysed the prevalence of self-citations in the domain of computer science based on the DBLP citation graph. We found an average incoming and outgoing self-citation rate of 9.44% and 11.40%, respectively. As in other domains, self-citation rates were highest in the first years after publication. The outgoing self-citation rate is relatively stable across conference rank, but the incoming rate differs across ranks, with C-rated venues having the highest rate. Further, we found, that the more authors, the higher the rates of outgoing and incoming self-citations are, while last authors have higher self-citation rates than

first authors for multi-author papers. Papers with male first authors have a higher outgoing self-citation rate, while papers with female first authors have a higher incoming self-citation rate.

In this study, we presented statistical observations without in-depth interpretation. Future work could address the questions that arise from these findings. One might consider investigating the reasoning behind the opposing self-citation rates of male and female lead authors or find arguments for the amount of incoming self-citation of C ranked venues. Furthermore, we considered every attributed (e.g., rank, year) on its own. Further analysis could reveal how attribute combinations influence the (self-) citation rates.

Acknowledgment. We would like to thank Moritz Grünbauer for his preliminary analysis and help with constructing the queries for the graph database.

References

1. Aksnes, D.W.: A macro study of self-citation. *Scientometrics* **56**(2), 235–246 (2003)
2. Alonso, S., Cabrerizo, F., Herrera-Viedma, E., Herrera, F.: h-index: a review focused in its variants, computation and standardization for different scientific fields. *J. Inf.* **3**(4), 273–289 (2009)
3. Bartneck, C., Kokkelmans, S.: Detecting h-index manipulation through self-citation analysis. *Scientometrics* **87**(1), 85–98 (2010)
4. Costas, R., van Leeuwen, T.N., Bordons, M.: Self-citations at the meso and individual levels: effects of different calculation methods. *Scientometrics* **82**, 517–537 (2010)
5. Ferrara, E., Romero, A.E.: Scientific impact evaluation and the effect of self-citations: Mitigating the bias by discounting the h-index. *JASIST* **64**(11), 2332–2339 (2013)
6. Fowler, J.H., Aksnes, D.W.: Does self-citation pay? *Scientometrics* **72**(3), 427–437 (2007)
7. Gauch Jr., H.G.: *Scientific Method in Practice*. Cambridge University Press, Cambridge (2002). <https://doi.org/10.1017/CBO9780511815034>
8. Ghiasi, G., Larivière, V., Sugimoto, C.R.: Gender differences in synchronous and diachronous self-citations. In: *Proceedings International Conference on Science and Technology Indicators* (2016)
9. Glänzel, W., Debackere, K., Thijs, B., Schubert, A.: A concise review on the role of author self-citations in information science, bibliometrics and science policy. *Scientometrics* **67**, 263–277 (2006)
10. Hemmat Esfe, M., Wongwises, S., Asadi, A., Karimipour, A., Akbari, M.: Mandatory and self-citation; types, reasons, their benefits and disadvantages. *Sci. Eng. Ethics* **21**, 1581–1585 (2015)
11. Ioannidis, J.P.: A generalized view of self-citation: direct, co-author, collaborative, and coercive induced self-citation. *J. Psychosom. Res.* **78**, 7–11 (2015)
12. King, M.M., Bergstrom, C.T., Correll, S.J., Jacquet, J., West, J.D.: Men set their own cites high: gender and self-citation across fields and over time. *Socius* **3** (2017)
13. Larivire, V., Ni, C., Gingras, Y., Cronin, B., Sugimoto, C.: Bibliometrics: global gender disparities in science. *Nature* **504**, 211–213 (2013)
14. Lawani, S.M.: On the heterogeneity and classification of author self-citations. *J. Am. Soc. Inf. Sci.* **33**(5), 281–284 (1982)

15. Leblond, M.: Author self-citations in the field of ecology. *Scientometrics* **91**, 943–953 (2012)
16. Lee, D., On, B.W., Kang, J., Park, S.: Effective and scalable solutions for mixed and split citation problems in digital libraries. In: *Proceedings of the International Workshop on Information Quality in Information Systems*, pp. 69–76. ACM, New York (2005)
17. Ley, M.: The DBLP computer science bibliography: evolution, research issues, perspectives. In: Laender, A.H.F., Oliveira, A.L. (eds.) *SPIRE 2002*. LNCS, vol. 2476, pp. 1–10. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45735-6_1
18. Medoff, H.M.: The efficiency of self-citations in economics. *Scientometrics* **69**, 69–84 (2006)
19. Müller, M.C., Reitz, F., Roy, N.: Data sets for author name disambiguation: an empirical analysis and a new resource. *Scientometrics* (2017). <https://doi.org/10.1007/s11192-017-2363-5>
20. Purvis, A.: The h index: playing the numbers game. *Trends Ecol. Evol.* **21**(8), 422 (2006). <https://doi.org/10.1016/j.tree.2006.05.014>
21. Schubert, A., Glänzel, W., Thijs, B.: The weight of author self-citations. A fractional approach to self-citation counting. *Scientometrics* **67**, 503–514 (2006)
22. Thijs, B., Glänzel, W.: The influence of author self-citations on bibliometric meso-indicators. The case of European universities. *Scientometrics* **66**, 71–80 (2006)



A Semantic-Based Personalized Information Retrieval Approach Using a Geo-Social User Profile

Tahar Rafa^{1(✉)} and Samir Kechid²

¹ Department of Mathematics and Computer Science,
University of Medea, Medea, Algeria

rafa.tahar@gmail.com, rafa.tahar@univ-medea.dz

² LRIA Laboratory, Department of Computer Science, USTHB, Algiers, Algeria
kechidsam@yahoo.fr, skechid@usthb.dz

Abstract. The user's search history includes some latent semantics that can be used to improve the user's interests representation. In this paper we present a personalized information retrieval approach that highlights and use this latent semantics. This semantics, which we call personal semantics, is expressed by the different co-occurrence relationships between relevant terms, according to the different user's search contexts. This personal semantics is integrated in a geo-social user profile for giving it more representativeness of the user interests. Then, to improve the search results relevance, the user profile is used to reformulate the user query for broadening the search scope without going further than the user needs.

Keywords: Personalized information retrieval · Personal semantics
Personalized query reformation · User profile

1 Introduction

The objective of personalized information retrieval (PIR) is to infer the real user information need hidden behind his query, and then tailor search results to this need [1, 2]. The problem of classic search is that it returns an important number of results that naively seem to match the user query. To be able to filter these results and then to retain only those most relevant according to the user intentions, personalized search systems use a structure, conventionally called user profile [3], allowing a detailed description of user interests. The user profile that gathers data explicitly given by the user and other data implicitly extracted from his relevant search history will be integrated into the search process to improve the search results relevance [4].

The user profile is represented, in the majority of PIR approaches, by a set of terms without taking into account the different semantic links between these terms. This syntactic representation diminishes the ability of the user profile to well represent the user's cognitive needs [3, 5].

Several works in PIR are oriented, in recent years, towards the introduction of semantics in the user profile to improve the accuracy and relevance of search results by

better understanding the user's intentions [13]. Indeed, semantics does not consider documents as a simple set of unrelated terms, but rather it takes into account the meaning conveyed by these terms [6].

In this paper we propose a new semantic-based PIR approach in which we use a personal semantics extracted from the different contexts of the user's previous searches (thematic, social and situational). From our viewpoint, the co-occurrence relationships between relevant terms themselves, between terms and user's social annotations and between terms and locations are meaning carriers. We believe that this personal semantics latent in the user's search history can help well to improve the identification of the user's information needs.

Our goal is to use the user profile to improve the important phase of the query expansion to better express the user's information needs, which improves the recall performance by broadening the search scope, without going further than the user needs.

The remainder of this paper is organized as follows: Sect. 2 is dedicated to related works, followed by the representation of the proposed user profile in Sect. 3. In Sect. 4 we present our proposed research process. Section 5 is devoted to the approach evaluation, and we end in Sect. 6 with a conclusion and perspectives.

2 Related Works

2.1 On User Profile Modeling

The user-oriented information retrieval approaches offer an important variant of models to designing user profiles. In addition to data explicitly introduced by the user, each approach has its viewpoint concerning implicit relevance sensors that can better characterize the user.

Some works consider the social environment as a discriminating factor to identify the user interests. Bouhini et al. [4] propose a user profile based on the combination of user social annotations and the relevant terms of search history. The authors of [7] design a user profile with combination of the user's social tags and social relations with other users.

Other works exploit situational information (GPS coordinates, location name, location type, etc.) to improve the search results relevance. Akermi et al. [8] propose a situational user profile based on the user's locations and preferences. It is used to recommend the right information depending on the user current location. Also, Buidguaguen [9] proposes a situational user profile composed of the different search locations and the attached interest centers.

Halfway between these two types of approaches, some approaches combine the relevance sensors from the social and situational user contexts. In our previous work [10], we presented a geo-social user profile that combines relevant information issued from the thematic, social and situational aspects of the user's previous searches. Aneja and Gambhir [11] propose a geo-social user profile combining the browsing history and location information. Also, Bao et al. [12] is presented a user profile that combines the user's preferences and the user's social annotations on the most visited places.

2.2 On Using Semantics in Personalized Information Retrieval

To improve the representation accuracy of the user's cognitive needs, some approaches of the personalized information retrieval introduce the semantics in the user profile.

Mohamed and Abdelmoty [13] propose a spatio-semantic user profile in a location-based social network. This profile is constructed by combining information about the different user's visited locations, and the user's social annotations on these locations. The authors use the "Foursquare" location database to extract the location's semantic information (name, category, region...), and the "WordNet" ontology to filter and categorize the user's annotations. Also, the authors of [2] propose to model the user profile by a graph of concepts. They use the concepts of ODP ontology. Hopfgartner and Jose [14] propose a semantic user profile for a personalized news video retrieval. This profile is based on the user history of relevant videos and the user's interactions, and the semantics links are introduced by using the Linked Open Data Cloud ontology to identify the similarities between videos.

3 Proposed User Profile

In this paper, we adopt the same geo-social user profile proposed in our previous work [10] that combines the relevance sensors from the three aspects: (i) Thematic aspect represented by the relevant terms of the search history. (ii) Social aspect, represented by the user's most used tags to annotate the relevant documents. And (iii) Situational aspect, represented by information about the different locations from where the user has made information searches.

To improve the representation of the multiple and dynamic user's interests, we reinforce this geo-social user profile by introducing a personal semantics in which we extract the semantic links from the thematic, social and situational contexts of the user search history.

The semantic geo-social user profile that we propose consists of: (i) a base of research situations (B_{RS}) that serves to store the relevant history, and (ii) a semantic representation space which serves to filter and classify the relevant terms of (B_{RS}) according to the user's search contexts qualified as importance providers for these terms.

3.1 Base of Research Situations (B_{RS})

This base is used to save the user relevant search history. It groups all the research situations (RS) related to the different locations from where the user has made information searches.

A research situation (RS) represents the relevant history related to the set of user queries sent in the same location. Each RS consists, as shown in Fig. 1, of three components:

Situational Component: It includes data about the user location. One location is represented by its parts: (location name, location type, city and country). Location data can be derived by input the user's GPS coordinates in a geographic database.

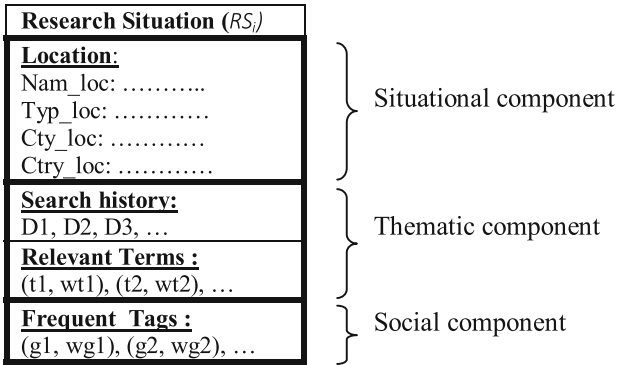


Fig. 1. The research situation model.

Thematic Component: is used to save the relevant search history of the concerned RS. This history is represented by the set of user’s relevant documents (long visited, printed, shared, annotated, marked, etc.), and the set of K most weighted terms in the RS’s search history.

The weight of a term *t* in one RS’s search history is calculated using the *tf * idf* formula of the vector model [15] as following:

$$W(t) = \left(05 + 0.5 * \frac{tf}{Max\ tf} \right) * \log \left(\frac{N}{n} \right) \tag{1}$$

With:

- *tf*: total frequency of term *t* in the RS’s documents.
- *Max tf*: the highest frequency of terms in RS’s documents.
- *N*: number of documents in the RS’s history.
- *n*: number of documents containing the term *t* in the RS’s history.

Social Component: represented by the set of K’ tags most used by the user to annotate the RS’s relevant documents.

The weight of a tag *g* is also calculated using the *tf * idf* formula of the vector model as following:

$$W(g) = \left(05 + 0.5 * \frac{gf}{Max\ gf} \right) * \log \left(\frac{N}{n} \right) \tag{2}$$

With:

- *gf*: total frequency of tag *g* in the RS’s documents.
- *Max gf*: the highest total frequency of the tags in the RS’s documents.
- *N*: number of RS’s documents.
- *n*: number of RS’s documents annotated by the tag *g*.

3.2 Semantic Representation Space of Relevant Search History

As we have already explained, we introduce in the geo-social user profile a personal semantic which is interested by the valorization of the latent semantics in the relevant history of the user's previous searches. We believe that the following three facts are meaning carriers: (i) the joint appearance of two relevant terms in many documents, (ii) the annotation by the same tag of many documents containing the same term, and (iii) the frequent occurrence of a relevant term in many documents of different searches made in the same location.

In this phase of semantic representation (Fig. 2) we proceed to classify the relevant terms of the search history into the following 3 bases: (i) base of thematic semantics, (ii) base of social semantics, and (iii) base of location-linked semantics.

Base of the Thematic Semantics (Term-Term Link): The thematic semantics is represented by a link connecting each relevant term t_i to each one of other relevant terms t_j of the user profile history. We consider the frequent co-occurrence relationship between terms as semantic link.

The thematic semantics between two relevant terms is saved in a base called (B_Sem_Them), and is expressed by a similarity noted $sim_them(t_i, t_j)$ and calculated as follows:

$$Sim_Them(t_i, t_j) = \frac{|d(t_i, t_j)|}{|d(t_i)|^2 + |d(t_j) - d(t_i, t_j)|^2} \quad (3)$$

With:

- $|d(t_i, t_j)|$: number of documents containing t_i and t_j .
- $|d(t_i)|$: number of documents containing t_i .
- $|d(t_j) - d(t_i, t_j)|$: number of documents containing t_j and not containing t_i .

This thematic similarity is not bidirectional ($sim_them(t_i, t_j) \neq sim_them(t_j, t_i)$), and it favors, as terms more linked to a given term t_i , the terms satisfying the 3 criteria: (i) appearing in the maximum of documents containing t_i , (ii) absent in the minimum of documents containing t_i , and (iii) appearing in the minimum of documents not containing t_i .

Base of the Social Semantics (Term-Tag Link): The social semantics represent the link connecting each relevant term t_i of the user profile with each one of tags g_j most used by the user to annotate the documents containing t_i . We consider the frequent co-occurrence relationship between one term and one tag as semantic link.

The social semantics between a term and a tag is stored in a base called (B_Sem_Scl), and is expressed by a similarity noted $sim_scl(t_i, g_j)$, and is calculated as follows:

$$Sim_Scl(t_i, g_j) = \frac{|d(t_i, g_j)|}{|d(t_i)|^2 + |d(g_j) - d(t_i, g_j)|^2} \tag{4}$$

With:

- $|d(t_i, g_j)|$: number of documents containing t_i and annotated by the tag g_j .
- $|d(t_i)|$: number of documents containing t_i .
- $|d(g_j) - d(t_i, g_j)|$: number of documents annotated by g_j and not containing t_i .

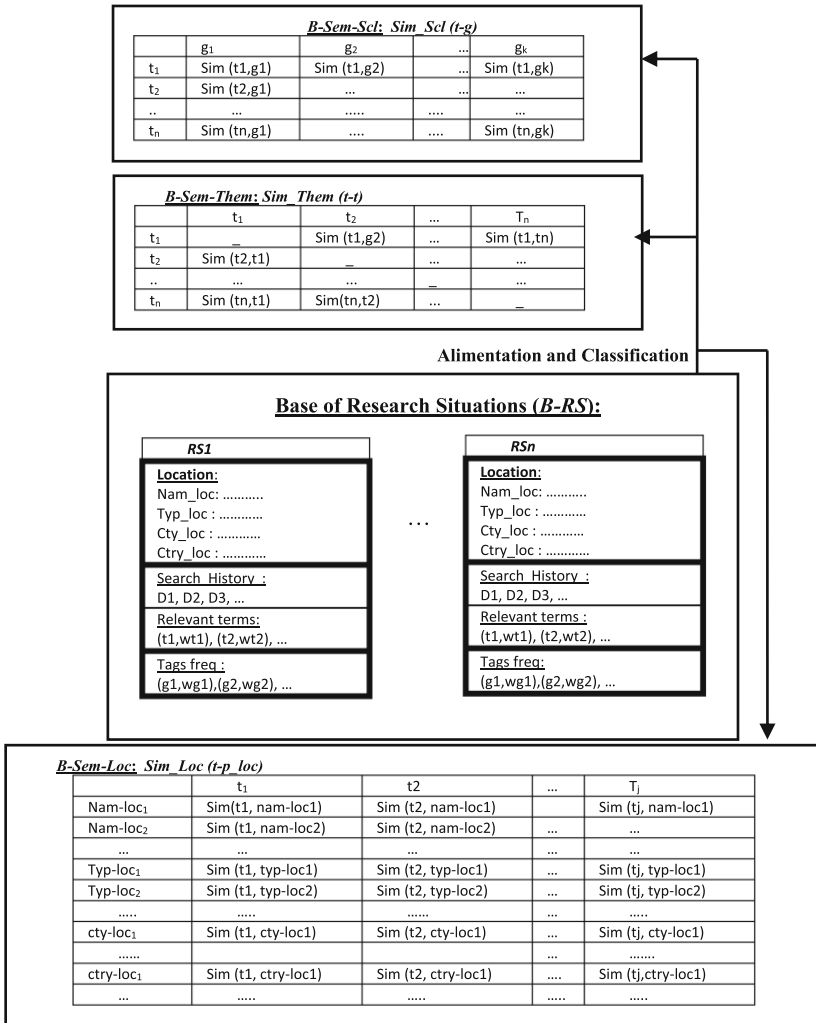


Fig. 2. Semantic representation space.

The proposed formula to calculate the social similarity favors, as tags more linked to a given term t_i , the tags satisfying the 3 criteria: (i) used to annotate the maximum of documents containing t_i , (ii) not used to annotate the minimum of documents containing t_i , and (iii) used to annotate the minimum of documents not containing t_i .

Base of the Location-Linked Semantics (Term-Location Link): The Location-linked semantics represent the link between the relevant terms and the different locations from where the user has made information searches. We consider the frequent occurrence of one term in many research situations that made in a same location as semantic link.

The Location-linked semantics is saved in a base called (B_Sem_Loc). For each relevant term of the user profile, we compute its connection degree with each one of the location parts among (name, type, city and country) of all the locations of the B_RS . And so, for each relevant term t_i and each location part $part_loc_j$, we propose the following similarity formula:

$$Sim_loc(t_i, p_loc_j) = \frac{|RS(t_i, p_loc_j)|}{|RS(t_i)|^2 + |RS(p_loc_j) - RS(t_i, p_loc_j)|^2} * \frac{|dist_loc_RS|}{|RS|^2} \quad (5)$$

With:

- $|RS(t_i, p_loc_j)|$: number of RS in which t_i is relevant and the location includes the concerned p_loc_j among (name, type, city or country of location).
- $|RS(t_i)|$: number of RS containing t_i .
- $|RS|$: total number of RS in B_RS .
- $|RS(p_loc_j) - RS(t_i, p_loc_j)|$: number of RSs containing the concerned p_loc_j , and in which t_i does not appear.
- $|dist_loc_RS|$: total number of distinct locations in the B_RS .

We notice that the formula (5) proposed to calculate the location-linked similarity gives importance to the number of distinct locations in the user profile, because if the user does not change place, we can't judge that the relevance of a given term is related to the user location.

This formula favors, as terms more linked to a given part of location $part_loc_j$ among (name, type, city and country of location), the terms satisfying the 3 criteria: (i) appearing in the maximum of RS whose location includes $part_loc_j$, (ii) absent in the minimum of RS whose location includes $part_loc_j$, and (iii) appearing in the minimum of RS whose location does not include $part_loc_j$.

4 The Search Process

Our search process (Fig. 3) follows the following scenario:

1. The system receives the user query Q, and retrieves the current user location.
2. The system selects among (B_Sem_Them , B_Sem_Scl and B_Sem_Loc) the most appropriate profile base to the user query Q.

3. The query Q is reformulated (enriched), according to the selected base.
4. The reformulated query Q' is sent to the classic search engine.
5. The system observes the reactions and the interactions of the user with the final returned results, to update his profile (identify and classify by contexts the new relevant terms).

In the following subsections, we present the details of each step of our search process.

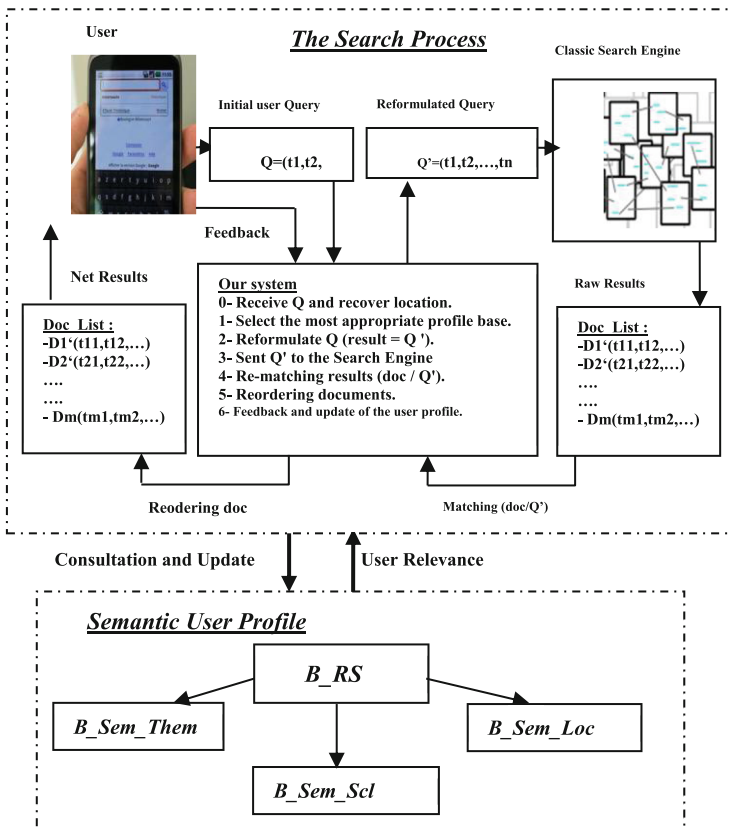


Fig. 3. The search process.

4.1 Selection of the Most Appropriate Profile Base for the User Query

To select from the 3 profile bases (B_Sem_Them , B_Sem_Scl and B_Sem_Loc) the most appropriate to the user query Q , our system calculates a similarity score between the user query Q and each of these bases respectively as follows:

$$Sim(B_Sem_Them, Q) = \sum_{t_i \in Q} \sum_{t_j \in B_Sem_Them} sim_them(t_i, t_j) \quad (6)$$

$$Sim(B_Sem_Scl, Q) = \sum_{t_i \in Q} \sum_{g_j \in B_Sem_Scl} sim_Scl(t_i, g_j) \quad (7)$$

$$Sim(B_Sem_Loc, Q) = \sum_{t_i \in Q} \sum_{loc_j \in current\ location} sim_Loc(t_i, loc_j) \quad (8)$$

We retain as user's current search context, the base most similar to the user query.

4.2 User Query Reformulation

The initial user query is reformulated with the aim of widening the search field to select more relevant results without going further than the user interests. The reformulation consists in enriching the initial query Q by the injection of some terms, tags or location's data according to the selected profile base.

Thus, the new reformulated query Q' is obtained on the basis of the initial query Q as follows:

$$Q' = Q + P \quad (9)$$

With:

- In case of B_Sem_Them : P is the term that maximizes the similarity (3) with the terms of Q .
- In case of B_Sem_Scl : P is the tag that maximizes the similarity (4) with the terms of Q .
- In case of B_Sem_Loc : P is the location part (name, type, city or country) that maximizes the similarity (5) with the terms of Q .

4.3 Update of the User Profile

Concerning the update of the user profile, we must ensure on one side, its enrichment and its evolution over time by the addition of new research situations (RS) in the B_RS base, and the classification of their relevant terms in the different bases of the semantic representation space. On the other side, we must optimize the space and time occupation, by removing the old RSs.

Alimentation of User Profile: when the final results are transmitted to the user, our system observes the user's interaction with these results to identify the new documents, explicitly or implicitly, deemed relevant. These documents are indexed, and their terms are weighted using the formula (1), and also the tags used to annotate these documents are weighted using the formula (2). Also, a new research situation RS is instantiated with its three components (situational, thematic and social). Then, the K new most weighted terms are retained in the B_Sem_Them base, the K' new most weighted tags are retained in the B_Sem_Scl base, and the new parts of the RS's location are saved in the B_Sem_Loc base.

Cleansing of User Profile: The proposed formulas for measuring the semantic similarities between relevant terms and the different contexts: thematic (3), social (4) and location-based (5) are defined as a proportion of the number of RS containing the concerned semantic relationship relative to the total number of RS in the B_RS. Thus, these formulas allow growth and decay of these similarities.

The evolutionary nature of the formulas quoted above allows us to adopt a very simple strategy to lighten the profile bases. This strategy consists of removing any RS that is not repeated for one year of its creation. So, the terms extracted only from a deleted RS are also deleted. By transitivity, the terms, tags and locations associated only to a deleted term will be also deleted.

5 Evaluation of the Approach

According to [16, 17], the relevance to be improved in the context of personalized information retrieval is very subjective because it depends particularly on user. This user is the unique able to judge this relevance according to his viewpoint. This reality makes the evaluation of a personalized information retrieval approach a very problematic task. Indeed, with the absence of test collections with queries and pre-judgments made by the concerned users themselves, we can not evaluate the improvements made by a new approach for the user.

From our viewpoint, the exploitation of test collections with queries and relevance pre-judgments made by domain experts, as is done in the majority of personalized information retrieval approaches, does not solve the problem, because it is not the opinion of the concerned users.

5.1 Evaluation Strategy

In the absence of an evaluation framework dedicated to a personalization context as already indicated, and as is mentioned in several recent PIR approaches [2, 4, 7, 9], where each approach develops its own evaluation framework, we will conduct our evaluation phase according to the following steps:

- (i) *Construction of the User Profile:* we built differently 30 semantic geo-social profiles for 30 users from different professional domains with different interests. We built each profile with the collaboration of the concerned user. The user's interests are distributed in a balanced way on the 4 bases of the semantic representation space. For each user, some interests are defined in a very specific way, in order to show the need to the personal semantics. Profile sizes vary between empty, poor, medium, and rich profiles.
- (ii) *Queries and Search:* we use *Google* as search engine. Each user sent 5 queries from different locations. For each user, at least one query must express specific information needs. In sum, 100 queries and 20 locations (some queries and some locations are common).
- (iii) *Evaluation:* to have a real feedback, the relevance judgments about returned results are made manually by users. The evaluation consists of calculating the

precision factor (rate of returned relevant documents) at the 5, 10 and 15 first documents. This represents the equivalent of the first 3 pages of results that can be visited by the user in the average.

- (iv) *Comparison of Performances*: to compare the contribution of our approach with respect to classical approaches, to non-semantic PIR approaches, and to non-personalized semantic-based approaches, we calculate the precision according to the following 4 scenarios: (A) query sent without reformulation, (B) query reformulated only by the location data according to the location-based personalized search approach proposed in [9], (C) query reformulated using the concepts derived from the *WordNet* ontology according to the semantic search approach proposed in [18], and (D) query reformulated using our semantic geo-social profile.

5.2 Results and Discussion

The following Table 1 presents the statistics of search results obtained according to the 4 scenarios already explained.

Table 1. Results and statistics

	Scen (A)	Scen (B)	Scen (C)	Scen (D)	Rate (D/A)	Rate (D/B)	Rate (D/C)
P @5	0.24	0.21	0.30	0.36	0.12	0.15	0.06
P @10	0.27	0.24	0.33	0.37	0.10	0.13	0.04
P @15	0.30	0.26	0.35	0.39	0.09	0.13	0.04
P-avg.	0.27	0.236	0.326	0.373	0.103	0.137	0.047

We note that the use of our semantic geo-social user profile has improved the precision factor compared to other approaches, particularly the precision at the level of top 5 documents that are most likely to be examined by the user.

This improvement is due to the way with which we designed the user profiles, as well as the set of sent queries. We notice that the results according to scenario (B) are less efficient than the classical scenario (A) because very few profiles are location-linked.

The details of the simulation show that our results are good in the case where the query is too short and the profile is very rich and our results are not very interesting in the case of a new profile.

It should be noted that the use of an evaluation framework larger than ours in terms of the number of users, the size of the profiles, the number of queries and locations, may give results that better reflect the reality.

6 Conclusion and Perspectives

In this paper we have proposed a new semantic approach for a personalized information retrieval. This approach is based on the definition of a semantic-based geo-social user profile that combines the user's interest indicators from different thematic, social and situational contexts of user's previous searches.

The proposed user profile uses a personal semantics represented by the semantic links between the relevant terms themselves, between relevant terms and user's social annotations, as well as between relevant terms and locations from where the user has made information searches.

The objective of the proposed approach is to improve the search results relevance by using the user profile to reformulate the initial user query.

The evaluation of the proposed approach shows that more than the user profile is richer, the user query is better reformulated and the search results are more relevant.

This work opens way to other research tracks. We plan to: (i) study the stability and the evolution over time of the user profile, to estimate how much this profile can be useful to infer the user's future information needs, and (ii) implement an online evaluation framework dedicated to the personalized information retrieval.

References

1. Hadjouni Krir, M.: Un système de recherche d'information personnalisée basé sur la modélisation multidimensionnelle de l'utilisateur. Ph.D. thesis Université Paris Sud – paris XI, France (2012)
2. Daoud, M., Tamine, L., Boughanem, M.: A personalized search using a semantic distance measure in a graph-based ranking model. *J. Inf. Sci. (JIS)* **37**(6), 614–636 (2011). <https://doi.org/10.1177/0165551511420220>
3. Uddin, M.N., Duong, T.H., Sean, V., Jo, G.-S.: Construction of semantic user profile for personalized web search. In: Nguyen, N.-T., Hoang, K., Jędrzejowicz, P. (eds.) ICCCI 2012. LNCS (LNAI), vol. 7654, pp. 99–108. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-34707-8_11
4. Bouhini, C., Géry, M., Largeton, C.: Personalized information retrieval models integrating the user's profile. In: 10th International Conference on Research Challenges in Information Science (RCIS 2016), June 2016, Grenoble, France, pp. 1–9. IEEE (2016). <https://doi.org/10.1109/rcis.2016.7549310>, <ujm-01377061>
5. Aimé, X., Fürst, F., Kuntz, P., Trichet, F.: Enrichissement sémantique de requêtes au moyen d'ontologies de domaine personnalisées. In: Actes de l'atelier «Personnalisation du Web», 10ième Journées francophones d'Extraction et de Gestion de Connaissances (EGC 2010), Hammamet, Tunisie (2010)
6. Zargayouna, H., Roussey, C., Chevallet, J.-P.: Recherche d'information sémantique: état des lieux. In: TAL, vol. 56, no. 3, pp 49–73 (2015)
7. Saoud, Z., Kechid, S.: Integrating social profile to improve the source selection and the result merging process in distributed information retrieval. *Inf. Sci.* **336**, 115–128 (2016). <https://doi.org/10.1016/j.ins.2015.12.012>
8. Akermi, I., Boughanem, M., Faiz, R.: Une approche de recommandation proactive dans un environnement mobile. In: INFORSID 2015, France, pp. 301–316 (2015)

9. Buidguaguen, O.: Accès contextuel à l'information dans un environnement mobile: approche basée sur l'utilisation d'un profil situationnel de l'utilisateur et d'un profil de localisation des requêtes. Ph.D. thesis, Univ Toulouse III - Paul Sabatier, France (2011)
10. Tahar, R., Samir, K.: A geo-social user profile for a personalized information retrieval. In: ACM ICIME 2016, Istanbul, Turkey, pp. 62–66 (2016). <http://dx.doi.org/10.1145/3012258.3012270>
11. Aneja, N., Gambhir, S.: Geo-social profile matching algorithm for dynamic interests in Ad-Hoc social network. In: Social Networking-3, pp. 240–247 (2014). <http://dx.doi.org/10.4236/sn.2014.35029>
12. Bao, J., Zheng, Y., Mokbel, M.F.: Location-based and preference-aware recommendation using sparse geo-social networking data. In: ACM SIGSPATIAL GIS 2012, Redondo Beach, CA, USA (2012)
13. Mohamed, S., Abdelmoty, A.I.: Spatio-semantic user profiles in location-based social networks. *Int. J. Data Sci. Anal.* **4**, 127–142 (2017). <https://doi.org/10.1007/s41060-017-0059-9>
14. Hopfgartner, F., Jose, J.M.: Semantic user profiling techniques for personalised multimedia recommendation. *Multimed. Syst.* **16**(4–5), 255–274 (2010). <https://doi.org/10.1007/s00530-010-0189-6>
15. Salton, G., Buckley, C.: Term weighting approaches in automatic text retrieval. *Inf. Process. Manag.* **24**, 513–523 (1988)
16. Toms, E., Agosti, M., Fuhr, N., Vakkari, P.: Evaluation methodologies in information retrieval. In: *Agstuhl Reports*, vol. 3, no.10, pp. 92–126 (2013)
17. Khazri, M., Tmar, M., Abid, M., Boughanem, M.: Proposition pour l'intégration des réseaux petits mondes en recherche d'information. In: Assogba, M.K. (ed.) Numéro spécial CARI, ARIMA, vol. 11, pp. 69–81 (2008)
18. Boubekeur, F., Azzoug, W.: Concept-based indexing in text information retrieval. *Int. J. Comput. Sci. Inf. Technol. (IJCSIT)* **5**(1), 119–136 (2013). <https://doi.org/10.5121/ijcsit.2013.5110>

Author Index

- Abddallah, Loai 115
Al-Amin, Sikder Tahsin 88
Al-Ghezi, Ahmed 65
Alirezai, Marjan 207
Aouabed, Haithem 191
- Bahra, Guryash 150
Barber, David 126
Beck, Tilman 230
Bellatreche, Ladjel 5, 88
Benndorf, Dirk 168
Bertolazzi, Paola 138
Böschen, Falk 230
Broneske, David 168
- Camilleri, Carl 21
Cappelli, Eleonora 138
Casturi, Rao 46
Charbonnier, Jean 243
Chen, Xiao 76
Cumbo, Fabio 138
- Dalpasso, Marcello 105
Durand, Gabriel Campero 76, 168
- Elloumi, Mourad 191
- Fatima, Alia 161
Felici, Giovanni 138
Fenske, Wolfram 168
Ferretti, Marco 179
- Galke, Lukas 218
Gerstenkorn, Gunnar 218
- Heyer, Robert 168
Horn, Christopher 38
- Jönsson, Arne 207
Josi, Frieda 243
Judd, Michael 126
- Kechid, Samir 301
Koh, Eunyee 254
- Lancia, Giuseppe 105
Lee, Tak Yeon 254
Leppänen, Ville 265
Lipka, Nedim 254
- Milz, Tobias 289
Musci, Mirto 179
- Nazir, Aiman Khan 161
Nezval, Vitezslav 21
Nyström, Mikael 207
- Ordonez, Carlos 5, 88
- Paschke, Adrian 278
Patrikar, Apoorva 168
- Qamar, Usman 161
Qundus, Jamal Al 278
- Rafa, Tahar 301
Rajaram, Akshay 126
Rapuru, Kirity 76
Rehman, Saad 161
Ruohonen, Jukka 265
- Saake, Gunter 76, 168
Santamaria, Rodrigo 191
Santini, Marina 207
Schallehn, Eike 76
Schallert, Kay 168
Scherp, Ansgar 218, 230
Seifert, Christin 289
Strandqvist, Wiktor 207
Sunderraman, Rajshekhar 46

Tropmann-Frick, Marina 38

Vella, Joseph G. 21

Wartena, Christian 243

Weitschek, Emanuel 138

Wiese, Lena 65, 150

Wolfrom, Brent 126

Yousef, Malik 115

Zoun, Roman 168

Zulkernine, Farhana 126