Abdelmonaime Lachkar
Karim Bouzoubaa
Azzedine Mazroui
Abdelfettah Hamdani
Abdelhak Lekhouaja (Eds.)

Communications in Computer and Information Science 782

Arabic Language Processing

From Theory to Practice

6th International Conference, ICALP 2017
Fez, Morocco, October 11–12, 2017
Proceedings

Springer

Communications in Computer and Information Science 782

Commenced Publication in 2007
Founding and Former Series Editors:
Alfredo Cuzzocrea, Xiaoyong Du, Orhun Kara, Ting Liu, Dominik Ślęzak, and Xiaokang Yang

Editorial Board

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

Abdelmonaime Lachkar · Karim Bouzoubaa
Azzedine Mazroui · Abdelfettah Hamdani
Abdelhak Lekhouaja (Eds.)

Arabic Language Processing

From Theory to Practice

6th International Conference, ICALP 2017
Fez, Morocco, October 11–12, 2017
Proceedings

Springer

Editors
Abdelmonaime Lachkar
Ex ENSA-USMBA
Fez
Morocco

and

ENSA-UAE
Tangier
Morocco

Karim Bouzoubaa
EMI, UM5
Rabat
Morocco

Azzedine Mazroui
FS, UMP
Oujda
Morocco

Abdelfettah Hamdani
IERA, UM5
Rabat
Morocco

Abdelhak Lekhouaja
FS, UMP
Oujda
Morocco

ISSN 1865-0929 ISSN 1865-0937 (electronic)
Communications in Computer and Information Science
ISBN 978-3-319-73499-6 ISBN 978-3-319-73500-9 (eBook)
https://doi.org/10.1007/978-3-319-73500-9

Library of Congress Control Number: 2017962905

Printed on acid-free paper

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

Preface

Welcome to the CCIS proceedings of the 6th International Conference on Arabic Language Processing – ICALP 2017. The conference was hosted in the imperial city Fez, Morocco, and organized by the Sidi Mohammed Ben Abdellah University, National School of Applied Sciences (ENSA-USMBA) in collaboration with the Arabic Language Engineering Society in Morocco (ALESM).

ICALP (ex-CITALA) started in 2006 and is held every two years. It is the flagship conference of the Arabic Language Engineering Society in Morocco (ALESM), which is the largest Arabic NLP institute in Morocco.

ICALP (ex-CITALA) is one of the premier international forums for disseminating new scholarly and technological work in Arabic computational linguistics; it has become one of the leading international conferences in the field of Arabic language processing and related applications. Each ICALP (ex-CITALA) conference attracts more authors from different countries in different continents thanks to the success of the preceding editions.

ICALP 2017 (ex-CITALA) provided excellent opportunities for scientists, researchers, industrial engineers, and university students to present their research achievements and to develop new collaborations and partnerships with experts in the field.

In this notable edition, we were indeed privileged to have had not one but two esteemed speakers. Our distinguished speakers were Dr. Khalid Choukri (ELRA Secretary General, ELDA founder, and CEO Evaluations and Language Resources Distribution Agency; ELDA), representing industry, and Prof. Allan Ramsay from Manschether University, UK, representing academia.

The keynote speakers shared their expertise with the aim of presenting to the participants a wide spectrum of their recent research developments in the field of Arabic NLP and related applications.

Also in this sixth edition, we were indeed privileged to have Springer publish the ICALP 2017 conference proceedings in their CCIS series.

ICALP 2017 focused on the following topics:

Information retrieval systems (IRS)
Question/answering systems (QAS)
Information extraction (IE)
Opining mining and sentiment analysis
Text categorization and clustering
Named entity recognition (NER)
Information extraction
Part-of-speech tagging (POS)
Word sense disambiguation (WSD)
Multi-word term extraction (MWTE)
Paraphrasing and textual entailment

Building ontologies
Text summarization
Machine translation systems (MTS)
Cross-language applications: Arabic–English, Arabic–Spanish, Arabic–French
Study of linguistic phenomena
Language resources: linguistic dictionary-based resources (phonology, morphology, semantics)
Development and normalization of linguistic resources and their exploitation in Arabic NLP applications
Spell-check and automatic corrections
Generation and analysis (morphology, syntax, semantics)
Linguistic resources and NLP applications
Comparative linguistic studies
Speech recognition and synthesis
Optical character recognition (OCR)

It is gratifying to note that the program of the ICALP conference covered a wide range of very interesting items (papers and posters) relating to Arabic NLP, from theory to practice, resources, and ontologies.

For this sixth edition, we received 64 submissions from 15 countries (Morocco, Algeria, Belgium, Canada, Egypt, France, Germany, Ireland, Palestine Qatar, Saudi Arabia, Tunisia, Turkey, USA, and UK).

Over 65 technical experts from all over the world participated in the peer-review process. Based on the review results, the Program Chairs accepted only 18 papers for oral presentation and to be included in the CCIS proceedings published by Springer.

In addition, 15 papers were selected for poster presentation. We received many good submissions, but could only accept a small number of them for oral presentation. The overall quality of submissions was extremely high, and we have had to decline a number of very good papers like those selected as posters.

Each paper was assigned to five reviewers, and generally each one was reviewed by no less than three reviewers.

We take this opportunity to extend a heartfelt thanks to the reviewers for providing high-quality and insightful reviews under a tight schedule, and also to thank the authors for their interest and contributions, which resulted in an outstanding technical program.

We are extremely grateful to the sponsors of the conference.

We extend a special word of thanks to all our colleagues of the Organizing Committee and secretariat for their hard work in organizing the conference, and to Springer for their assistance in publishing the ICALP 2017 proceedings in the CCIS series

Finally, we would like to take this opportunity to express our sincere thanks and deep appreciation to the organizers and in particular:

Prof. Omar Assobhei, President of USMBA University in Fez
Prof. Mustapha Ijaali, Director of ENSA, Dean of FST, USMBA University, Fez
Prof. Hassan Zaher, Dean of Faculty of Shari'a Sciences, USMBA University, Fez

All of them worked with us from the very beginning of the planning stage. We truly appreciate their dedication. The conference program would not have been possible without the following:

– Arabic Language Engineering Society in Morocco (ALESM)
– Université Sidi Mohamed Ben Abdellah (USMBA)
– Ecole Nationale des Sciences Appliquées de Fès (ENSA-Fès)
– Faculté des Sciences et Techniques (FST-Fès)
– Faculté des Sciences de la Scharia
– Centre National pour la Recherche Scientifique et Technique (CNRST-Morocco)
– Académie Hassan II des sciences et techniques

On behalf of the Steering, Program, and Organizing Committees, we hope you enjoy the contributions in this volume.

October 2017

Abdelmonaime Lachkar
Karim Bouzoubaa
Azzedine Mazroui
Abdelfettah Hamdani
Abdelhak Lekhouaja

Organization

General Chair

Abdelmonaime Lachkar	(Ex ENSA-USMBA, Fez), ENSA-UAE, Tangier, Morocco

Honorary Committee

Omar Assobhei	President of USMBA University, Fez, Morocco
Moulhime El Bekkali	Vice President of Sidi Mohamed Ben Abdellah University in Fez, Morocco
Brahim Akdim	Vice President of Sidi Mohamed Ben Abdellah University in Fez, Morocco
Mustapha Ijaali	Director of ENSA and Dean of FST, USMBA University, Fez, Morocco
Hassan Zaher	Dean of Faculty of Shari'a Sciences, USMBA University, Fez, Morocco

Steering Committee

Karim Bouzoubaa	EMI, UM5, Rabat, Morocco
Abdelhamid El Jihad	IERA, UM5, Rabat, Morocco
Abdelfettah Hamdani	IERA, UM5, Rabat, Morocco
Azzedine Lazrek	UCA, Marrakech, Morocco
Abdelhak Lekhouaja	UMP, Oujda, Morocco
Azzedine Mazroui	UMP, Oujda, Morocco

Communication and Publicity Committee

Jaoua Ali Mohamed	Qatar University, Qatar
Hassina Aliane	Centre de Recherche sur l'Information Scientifique et Technique, Algeria
Khalid Choukri	ELDA, European Language Resource Association, France
Haikal El Abed	Institute for Communications Technology, Germany
Ali Farghaly	DataFlux A SAS Company, USA
Salwa Hamada	Taibah University, Madinah, Saudi Arabia
Abdul Malik Al-Salman	King Saud University, Riyadh, Saudi Arabia
Elmohajir Mohammed	Chair of IEEE Morocco Section, FSDM, USMBA, Fez, Morocco
Ouafae Nahli	Institute for Computational Linguistics CNR, Pisa, Italy
Nasredine Semmar	CEA Saclay Nano-INNOV, France

Khaled Shaalan	British University in Dubai, UAE
Nadi Tomeh	Paris 13 University, France
Imed Zitouni	Microsoft Research, USA

Local Organizing Chair

Abdelmajid Saka	ENSA, USMBA, Fez, Morocco

Local Organizing Committee

Mohammed Berrada	ENSA, USMBA, Fez, Morocco
Zakariya Chalh	ENSA, USMBA, Fez, Morocco
Saad Bennani Dosse (IEEE Member)	ENSA, USMBA, Fez, Morocco
Hakim El Fadili	ENSA,USMBA, Fez, Morocco
Abdelmajid El Katani	USMBA, Fez, Morocco
Youness Elkhamlichi	ENSA, USMBA, Fez, Morocco
Souad Elkhattabi	ENSA, USMBA, Fez, Morocco
Abdelhakim Essbaai	USMBA, Fez, Morocco
Abdellatif Ezzouhairi (IEEE Member)	ENSA, USMBA, Fez, Morocco
Khalid Fardousse	USMBA, Fez, Morocco
Adil Kenzi (IEEE Member)	ENSA, USMBA, Fez, Morocco
Youness Lakhrissi (IEEE Member)	ENSA, USMBA, Fez, Morocco
Anas Mansouri (IEEE Member)	ENSA, USMBA, Fez, Morocco
Elhaj Ben Ali Safae	ENSA, USMBA, Fez

Finance Committee

Bouklata Ahmed	EST, USMBA, Fez, Morocco
Abdelhamid El Jihad	IERA, Rabat, Morocco
El Faik Hanane	ENSA, USMBA, Fez, Morocco

Webmaster and Social Media

El Bekkali Mohammed	ENSA, USMBA, Fez, Morocco

Program Committee

M'Hamed AitKbir	FST, UAE Tanger, Morocco
Hassina Aliane	Centre de Recherche sur l'Information Scientifique et Technique, Algeria
Hend Alkhalifa	King Saud University, Saudi Arabia

Wasfi Al-Khatib	King Fahd University of Petroleum and Minerals, Saudi Arabia
Aziza Alouaazizi	USMBA, Fez, Morocco
Abdul Malik Al Salman	King Saud University, Saudi Arabia
Mohammed Tabili Alaoui	FS UMP, Oujda, Morocco
Almoataz Bellah Al-Said	Cairo University, Egypt
Yassine Benajiba	Thomson Reuters, IP and Science
Karim Bouzoubaa	EMI, UM5, Rabat, Morocco
Violetta Cavalli-Sforza	Al Akhawayn University, Morocco
Khalid Choukri	ELDA, European Language Resource Association, France
Oumayma Dakkak	Higher Institute for Applied Science and Technology, Syrian Arab Republic
Kareem Darwish	Qatar Computing Research Institute, Qatar
Said Desouki	Higher Institute for Applied Science and Technology, Syrian Arab Republic
Mahmoud El-Haj	Lancaster University, UK
Abdelhamid El Jihad	IERA, UM5, Rabat, Morocco
Abderrahim Elqadi	EST, UMI, Meknès, Morocco
Ramy Eskander	Columbia University, USA
Ali Farghaly	DataFlux A SAS Company, USA
Bilel Gargouri	Université de Sfax, Tunisia
Nada Ghneim	Higher Institute for Applied Science and Technology, Damascus, Syria
Ahmed Guessoum	University of Science and Technology Houari Boumediene, Algeria
Hatem Haddad	Université Libre de Bruxelles, Belgium
Kais Haddar	University of Sfax, Tunisia
Salwa Hamada	Taibah University, Madinah, Saudi Arabia
Abdelfettah Hamdani	IERA, UM5, Rabat, Morocco
Yannis Haralambous	Institut Mines-Télécom and UMR CNRS 6285 Lab-STICC, France
Hamid Jaafar	UCA, Marrakech, Morocco
Mustafa Jarrar	Birzeit University, Sina Institute, Palestine
Ali Mohamed Jaoua	Qatar University, Qatar
Mohammed Kissi	FS, UCD, El Jadida, Morocco
Abdelmonaime Lachkar	ENSA, USMBA, Fez, Morocco
Abouenour Lahsen	EMI UM5, Rabat, Morocco
Azzedine Lazrek	UCA, Marrakech, Morocco
Abdelhak Lekhouaja	UMP, Oujda, Morocco
Mourad Loukam	Hassiba Benbouali University, Algeria
Sabri A. Mahmoud	King Fahd University of Petroleum and Minerals, Saudi Arabia
Azzedine Mazroui	UMP, Oujda, Morocco
Slim Mesfar	Université de Carthage, Tunisie
Farid Meziane	University of Salford, UK

Vito Pirreli	Institute for Computational Linguistics CNR, Pisa, Italy
Noureddine Rais	FSDM, USMBA, Fez, Morocco
Allan Ramsay	University of Manchester, UK
Salim Rami	UCA Marrakech, Morocco
Horacio Rodríguez	Universitat Politècnica de Catalunya, Spain
Paolo Rosso	Universidad Politécnica de Valencia, Spain
Nasredine Semmar	CEA Saclay Nano-INNOV, France
Khaled Shaalan	The British University, United Arab Emirates
Kamel Smaili	University of Lorraine Nancy, France
Abderrahim Tragha	UH2, Casablanca, Morocco
Adnan Yahya	Birzeit University, Palestine
Abdellah Yousfi	Mohammed V University, Morocco
Wajdi Zaghouani	Carnegie Mellon University, Qatar
Mohamed Zakaria Kurdi	University of Delaware, USA
Arssalan Zarghili	USMBA, Fez, Morocco
Petr Zemánek	Charles University in Prague, Czech Republic
Taha Zerrouki	University of Bouira, Algeria
Ahmed ZineEddine	USMBA, Fez, Morocco
Farah Benamara Zitoune	IRIT, Université Paul Sabatier, France
Imed Zitouni	Microsoft Research, USA
Mounir Zrigui	University of Monastir, Tunisia

Sponsors

USMBA University

Université Sidi Mohamed Ben Abdellah (USMBA)

جمعية هندسة اللغة العربية بالمغرب

Arabic Language Engineering Society in Morocco
ALESM

Arabic Language Engineering Society in Morocco (ALESM)

Centre National pour
la Recherche Scientifique
et Technique (CNRST-Morocco)

Académie Hassan II des sciences et techniques

Ecole Nationale des Sciences Appliquées
de Fès (ENSA-Fès)

LISA

Faculté des Sciences et Techniques,
(FST-Fès)

Faculté des Sciences de la Scharia

Contents

Machine Translation Systems

An Automatic Approach for WordNet Enrichment Applied to Arabic WordNet

Mohamed Seghir Hadj Ameur(✉), Ahlem Chérifa Khadir(✉), and Ahmed Guessoum(✉)

NLP, Machine Learning and Applications (TALAA) Group,
Laboratory for Research in Artificial Intelligence (LRIA),
Department of Computer Science,
University of Science and Technology Houari Boumediene (USTHB),
Bab-Ezzouar, Algiers, Algeria
{mhadjameur,akhadir,aguessoum}@usthb.dz

Abstract. This paper introduces an automatic method to extend existing WordNets via machine translation. Our proposal relies on the hierarchical skeleton of the English Princeton WordNet (PWN) as a backbone to extend their taxonomies. Our proposal is applied to the Arabic WordNet (AWN) to enrich it by adding new synsets, and also by providing vocalizations and usage examples for each inserted lemma. Around 12000 new potential synsets can be added to AWN with a precision of at least 93%. As such the coverage of AWN in terms of synsets can be increased from 11269 to around 24000 a very promising achievement on the path of enriching the Arabic WordNet.

Keywords: Arabic WordNet · Princeton WordNet · Ontology
Machine translation · Word alignment · WordNet enrichment
Heuristic search · Multi-feature optimization

1 Introduction

In almost all the applications of Natural Language Processing (NLP), dealing with semantics relies on the availability of lexical-semantic resources. WordNets are among the most important of such resources. They are used in text classification, information retrieval, semantic analysis, etc. [1]. WordNets are lexical databases that organize the words of a language according to their meanings. The WordNet backbone consists of a hierarchy of concepts called taxonomy. Each concept is expressed by a group of lemmas (*words*) that have the same meaning and form what is called a *synset*. As such, WordNets can be viewed as semantic networks where the words are linked to the concepts they belong to.

The issue is to build these useful databases. While, for the English language, building the first WordNet [2] known as the Princeton WordNet (PWN)[1] was

[1] https://wordnet.princeton.edu/.

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 3–18, 2018.
https://doi.org/10.1007/978-3-319-73500-9_1

done by language experts and it subsequently imposed itself as a reference, for many other languages there is a lack of means to build a WordNet as rich as needed in various applications. Building and extending them manually is such a time-consuming and costly task. So researchers try to propose ways to automate these tasks and minimize the human intervention.

The approach we propose here aims at enriching existing WordNets by first translating English lemmas that denote synsets in PWN to a target language using parallel corpora. Then, the translated lemmas are grouped together to form synsets in the target language, and finally find their positions in the hierarchy according to the original positions of PWN synsets. As such, our focus is on poor WordNets that have the same skeleton as PWN and whose existing concepts (synsets) are mapped to the concepts of the latter. We have chosen the Arabic WordNet [3] as an application example for our approach. Despite the efforts already done for its enrichment [4–6], this WordNet still needs improvements to increase its Arabic language coverage. It currently contains only 11k synsets against 117k for PWN even though the Arabic language is very rich both at the morphological and semantic levels.

The remainder of this paper is organized as follows: the next section presents some relevant related works that have been done in this research area. Our proposed enrichment system is described in Sect. 3. In Sect. 4, we present the tests that we have done and the results we have obtained. In the last section, we conclude our work and highlight some possible future improvements.

2 Related Work

WordNet enrichment is still an open issue for many languages, and even the most developed of them for these languages requires improvements and updates. Consequently, there are many approaches in the literature that propose automated and semi-automated methods for WordNet enrichment. Many of these use the PWN as a reference and/or resource. The work of Saveski and Trajkovski [7] aims to build a WordNet from scratch using the PWN as a backbone. The method was applied to the Macedonian language and it consists in translating English lemmas of PWN synsets to Macedonian using a machine readable dictionary. Then, the resulting candidate terms are filtered using the Google Similarity Distance [8] between each of them and the English synset gloss translated to Macedonian with Google Translate. They evaluate the resulting WordNet by using it in an application of text classification which improved the state of the art performances for it. There is also the work of Montazery et al. [9] which presents an automatic method for Persian WordNet construction. It uses Persian and English corpora as well as a bilingual dictionary in order to do a mapping between PWN synsets and Persian words. They calculate a score for each possible matching then select the one with maximum score as a link (PWN synset to Persian word). The precision was estimated at 82.6%. The more recent work of Mousavi and Faili [10] for Persian WordNet construction outperforms this precision reaching 91.18%, by proposing an approach based on supervised learning. They trained a classifier

on the existing WordNet to discriminate between correct and incorrect links. Another work of Niemi et al. [11] proposed to use bilingual resources to enrich Finn WordNet (for Finnish) based on PWN. They chose wikipedia as a bilingual resource to extend it and reached 89% accuracy for the added synonyms. In [12], Lindén et al. support the assumption that most synsets in PWN represent language-independent real-world concepts. As a consequence, the semantic relations between synsets are also assumed to be mostly language-independent. As such, the structure of PWN could be reused as well. This view inspired them to construct an extensive Finnish WordNet which is aligned with PWN. With the aim of improving the translation process, Tarouti et al. [13] proposed to use the word embeddings technique to filter the translation results. Cavalli et al. [14] presented a method for creating a WordNet for the Iraqi dialect. Following the original AWN skeleton, they used a bidirectional dictionary of Iraqi and English and the mapping between AWN and PWN to create an Iraqi WordNet.

For AWN there were also several works for its improvement; we cite in addition to the aforementioned list, the work of Boudabous et al. [15] which presents a linguistic method based on morpho-lexical patterns to add semantic relations between synsets, and the work of Al-Yahya et al. [16] that proposes a system for a semi-automated extraction of lexical relations, specifically antonyms, using a pattern-based approach to support the task of ontological lexicon enrichment.

Since many WordNets still need improvements, in addition to the fact that language lexicons are in continuous expansion, there is continuous demand for better and more efficient methods for WordNet enrichment.

3 The Automatic Enrichment System

In this section, we present the details of our proposed approach for WordNet enrichment. To illustrate the functioning of our system in a practical way, we have applied our proposal to show how to extend AWN based on PWN. Table 1 shows some statistics about the current numbers of lemmas and synsets in PWN and AWN. As shown in Table 1, it is clear that AWN still needs a lot of work to increase its coverage in terms of synsets which is what has motivated our work[2]. Let us point out that we will only be addressing the case of non-ambiguous lemmas[3]. We plan to address the case of ambiguous lemmas in a forthcoming work.

We also bring to the reader's attention that we have used our approach to extend AWN as an illustrative example only; our proposal is applicable to any other language whose WordNet has been mapped to PWN.

Figure 1 highlights the different components of our system. The first step extracts from the parallel corpus all the sentences that contain non-ambiguous lemmas; which will then be tokenized and normalized. Word alignment is then

[2] The version of AWN used in this work is available in this link http://globalwordnet.org/arabic-wordnet/awn-browse/.

[3] A lemma is said to be non-ambiguous if it appears in only one synset; it is considered ambiguous otherwise.

Table 1. Statistics about AWN and PWN

	PWN	AWN
Lemmas	148730	23480
Synsets	117659	11269

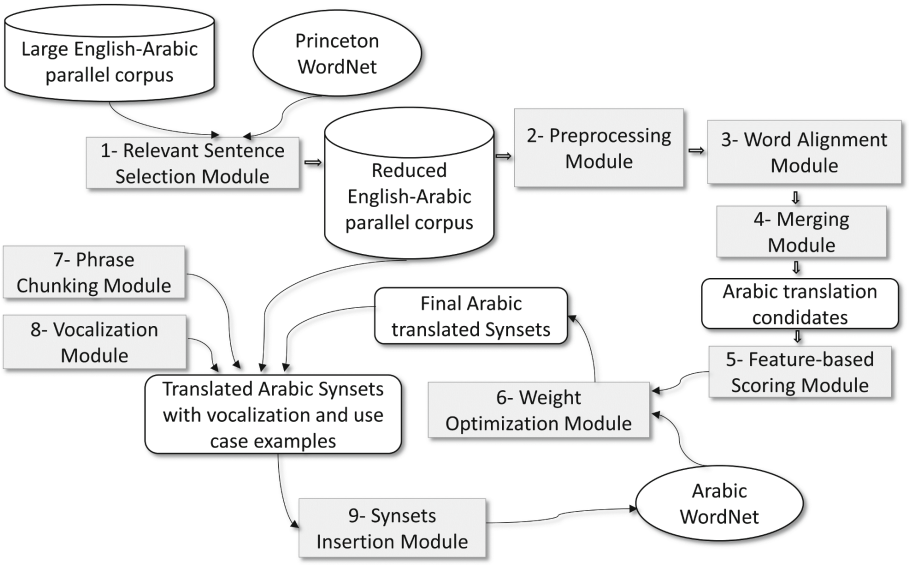


Fig. 1. The full architecture of the thesaurus enrichment system

used to associate all the possible Arabic translations to each English non-ambiguous lemma. The candidates belonging to the same synset will then be merged together to produce a set of candidates per synset. A score is then assigned to each candidate using a set of features. The parameters of these features will be optimized using an adequate metaheuristic. Then, for each selected candidate, we provide its vocalization and a lemma usage example. Finally, the translated synsets and their corresponding details will be added to AWN using an appropriate insertion method that takes into account its structure.

3.1 Sentence Selection

Our goal is to obtain the Arabic translation of the English PWN synsets. To this end, we first need to translate the lemmas contained in these synsets. This module aims at identifying the parallel sentences that contain these lemmas. Formally, given the set of English-Arabic parallel sentences $S = \{(e_1, a_1), ..., (e_n, a_n)\}$ and the set of all PWN non-ambiguous lemmas $L = \{l_1, l_2, ..., l_k\}$ where k is the total number of non-ambiguous lemmas in PWN, the sentence selection module associates to each lemma l_i all the parallel sentences (e_j, a_j) in which l_i appears in e_j.

3.2 Word Alignment and Merging Module

The set of selected sentences will then be preprocessed and used for word-alignment. We run direct and reversed alignment. Then we take their intersection to obtain strong and more reliable alignment points [17]. Next, for each non-ambiguous lemma l_i in the English sentences, we extract its Arabic correspondences (possible translations) according to the word-to-word alignment $\{(t_1, c_1), (t_2, c_2), ..., (t_k, c_k)\}$, where t_i is a candidate Arabic translation and c_i is its corresponding count, which indicates how many times the lemma l_i was aligned to t_i.

Since our goal is to obtain the Arabic translation of an entire synset and not only its individual lemmas, a merging step is needed. The merging consists in assembling all the translations of all the lemmas that belong to the same synset. The count of shared translations will be summed up.

Example 1 (Merging non-ambiguous lemmas). Suppose that we have the synset "card_game.n.01" which contains two non-ambiguous lemmas "cards" and "card_game". Each lemma has its own corresponding translations and counts:

cards: {(بطاقات,25), (البطاقات,20), (لعبة بطاقات,15), (لعبة البطاقات,5)}

card_game: {(لعبة بطاقات,20), (لعبة البطاقات,10), (بطاقات,5)}

When merging these two lemmas we add the counts of the shared translations. For instance, the translation "بطاقات" appears with a count of 25 in the first lemma and a count of 5 in the second lemma. Thus its final count will be 30. The result of merging these two lemmas will be:

card_game.n.01: {(لعبة بطاقات,35), (بطاقات,30), (البطاقات,20) ,(لعبة البطاقات,15)}.

3.3 Feature-Based Scoring Module

The previous module produces a set of translation candidates for each synset. To select the best candidate(s), we need to have a reliable scoring approach. This module aims to evaluate these candidates using several features. Given a synset s_j, we define a set of notations that will be used for all the considered features:

1. $L(s_j) = \{l_{j1}, l_{j2}, ..., l_{jd}\}$ is the set of all non-ambiguous lemmas contained in s_j, where d is the size of $L(s_j)$.
2. $C(s_j) = \{t_{j1}, t_{j2}, ..., t_{jk}\}$ is the set of all Arabic translation candidates obtained from the merging step for synset s_j, where k is the size of $C(s_j)$.

Specific features will be used to assign a certain score to each candidate $t_{ji} \in C(s_j)$. They are now presented.

Alignment Score. This first feature is used to evaluate the translation candidates according to their word-alignment frequencies. We use the following formula to calculate this score:

$$\phi(t_{ji}|s_j) = \frac{c(s_j, t_{ji})}{\sum_{m=1}^{k} c(s_j, t_{jm})}$$

where $\phi(t_{ji}|s_j)$ is the relative frequency of t_{ji} in $C(s_j)$ and $c(s_j, t_{jm})$ is the count of the candidate t_{jm} for the synset s_j, respectively.

This feature will favor the Arabic translations that have been aligned more frequently to the synset s_j and penalize low frequency alignments.

External Translation Score. This feature is used to score each translation candidate according to some external translation resources. In this work, we have considered two translation APIs: the Microsoft Translator API[4] and the Yandex Translation API[5].

The external translation APIs are used to translate each lemma $l_{ij} \in L(s_j)$ into Arabic, which produces a set of translations by synset. We define $T(s_j) = \{a_{j1}, a_{j2}, ..., a_{jp}\}$ of cardinality p as the set of translations obtained by the external translation APIs. We consider $T(s_j)$ as references and use a character-based Blue Score [18] to score each candidate in $C(s_j)$ based on the maximum similarity level that it shares with the references of T(s_j). This is done as stated in Algorithm 1.

Algorithm 1. Scores the Arabic candidates according to the external translation references

Input : s_j: a synset
$C(s_j) = \{t_{j1}, ..., t_{jk}\}$: s_j translation candidates
$T(s_j) = \{a_{j1}, ..., a_{jp}\}$: s_j translation references
Output: $C\_scored(s_j) = \{(t_{j1}, score_{j1}), ..., (t_{jk}, score_{jk})\}$: a scored candidates list

```
Function BlueEstimation(C(s_j), T(s_j)):
    foreach candidate t_ji in C(s_j) do
        top_sim_{t_ji} = 0
        foreach reference a_ji in T(s_j) do
            char_blue_ji = char-based Blue Score between a_ji and t_ji
            if char_blue_ji > top_sim_{t_ji} then
                top_sim_{t_ji} = char_blue_ji
            end
        end
        C_scored(s_j).append(t_ji, top_sim_{t_ji})
    end
return (C_scored(s_j))
```

[4] https://www.microsoft.com/en-us/translator/translatorapi.aspx.
[5] https://tech.yandex.com/translate/.

External Dictionaries Score. This feature is used to score each candidate according to some external English-Arabic dictionaries. We have used four dictionaries: Universal dictionary database, Wiktionary database, Omegawiki database, Wikipedia interlanguage links. All these dictionaries are freely available[6].

The downloaded dictionaries are merged into one English-Arabic dictionary to provide for each English word a set of reference Arabic translations. We note $T(w_j) = \{a_{j1}, a_{j2}, ..., a_{jp}\}$ as the set of all possible Arabic translations for a given English dictionary word w_j.

For a given synset s_j, we create a list TW_{s_j} containing all the translated words of each lemma $l \in L(s_j)$. We define three numerical entities:

1. For each Arabic candidate $t_{ji} \in C(s_j)$ having d words $\{w_1,w_2,...,w_d\}$, we count the number of words that appear in TW_{s_j}; we name this number $correct\_tran_{t_{ji}}$.
2. We define $cond\_len_{t_{ji}}$ to be the number of words in t_{ji}.
3. We define $ref\_len_{t_{ji}}$ to be the average number of words in all the lemmas in $L(s_j)$.

Using these defined entities, we define the estimated recall and precision [19] as follows:

$$E\_precision_{t_{ji}} = \frac{correct\_tran_{t_{ji}}}{cond\_len_{t_{ji}}}$$

$$E\_recall_{t_{ji}} = min(1, \frac{correct\_tran_{t_{ji}}}{ref\_len_{t_{ji}}})$$

Using the estimated recall and precision, we define the final score of the candidate t_{ji} using the $F\_measure$ [19] as follows:

$$F\_measure_{t_{ji}} = 2 * \frac{E\_precision_{t_{ji}} * E\_recall_{t_{ji}}}{E\_precision_{t_{ji}} + E\_recall_{t_{ji}}}$$

3.4 Feature Weights Optimization

After assigning a score to each candidate by means of several features, we need to combine these scores together using a simple linear combination $\alpha_1 * F_1 + \alpha_2 * F_2 + ... + \alpha_k * F_k$ where the $F_i's$ are all the considered features and the $\alpha_i's$ are their corresponding weight factors, with $\sum_{i=1}^{k} \alpha_i = 1$ and $0 <= \alpha_i <= 1$. For example, the weight configuration $[0.1, 0.2, 0.7]$ indicates that the first, second and third features have an importance of 10%, 20% and 70%, respectively. Our goal is to find the optimal weight configuration that yields the best translation performance. To this end, the set of values of each continuous weight F_i that varies between 0 and 1 is treated as a discrete interval of values defined according to a certain step p. This step will decide the number of values that can be taken by each weight such that $|F_i| = \frac{1}{p} + 1$. For example, if we consider the step to be

[6] http://www.dicts.info/uddl.php.

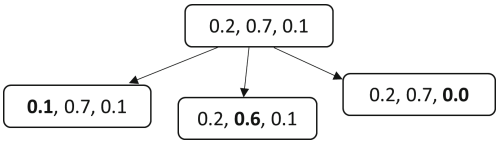


Fig. 2. Local search in the space of weight configurations

$p = 0.1$ then $|F_i| = 11$, which means that feature F_i can take one of 11 values i.e. $F_i \in \{0, 0.1, 0.2, ..., 1\}$.

To be able to automatically evaluate the quality of a given weight configuration, we need to have data that provides for each synset the correct outputs (references) that our system should produce. Based on the system outputs and the references, the system precision can be calculated. We use the preexisting AWN synsets as gold references and we monitor our system precision when the weight configuration varies.

A Tabu search metaheuristic [20] is used for weight optimization. Starting from a random point in the space of all possible weight configurations. We perform a local search where the neighboring states are all the configurations that can be reached from the current state using a step p.

Figure 2 shows the accessible states from a given initial random weight configuration [0.2, 0.7, 0.1] with a step $p = 0.1$. In this initial state, the sum of its weights is 1; thus a step can be applied only to decrease one of the possible weights producing the three neighbor states as shown in the same figure. We note that the sum of the weights can never exceed 1, however it can take any value between 0 and 1.

We use a Tabu search [21] that takes as input a set of Arabic synsets $S = \{s_1, s_2, ..., s_n\}$ where each synset s_j is associated with its translation candidates $C(s_j) = \{t_{j1}, ..., t_{jk}\}$ and its AWN references $R(s_j) = \{a_{j1}, a_{j2}, ..., a_{jp}\}$. Our goal is to find the optimal weight configuration which yields the best translation performance. Starting from a random weight configuration, a fitness function is used to estimate the precision between the selected translations and the AWN references. In each iteration a local search is applied by moving forward or backward a certain parameter in the weight configuration with a specific step p, as shown in Fig. 2. When no more improvements can be achieved in the local configuration space, a diversification process is used to select another random starting point and update the global optimum if needed. If no more improvement can be achieved after several iterations, or the maximum number of iterations is met, the search process finishes and the optimal global weight configuration will be returned.

3.5 Vocalization and Phrase Chunking

We aim to extend the AWN with not only the translated Arabic synsets but also vocalizations and lemma usage examples illustrating how these lemmas can be employed in real sentences.

Given the set of selected Arabic translated lemmas $\{t_{j1}, t_{j2}, ..., t_{jk}\}$ for an English synset s_j, we associate with each translation its original Arabic sentence to get a set of pairs $\{(t_{j1}, sent_{j1}), (t_{j2}, sent_{j2}), ..., (t_{jk}, sent_{jk})\}$ where each translated lemma t_{ji} is provided with one sentence that contains it.

To vocalize each lemma t_{ji}, we use the Arabic vocalization tool proposed in [22][7]. The entire Arabic sentence $sent_{ji}$ is first vocalized[8]; then we extract the vocalized Arabic lemma from it.

To provide an illustrating example (lemma usage example) that shows how the lemma t_{ji} is used in a real sentence, we first phrase-chunked the Arabic sentence $sent_{ji}$ using the AMIRA toolkit [23]. Then we took the sequence of chunks which can encompass the whole Arabic lemma t_{ji} and also has an extra chunk at each one of its two extremes. These extra chunks can be seen as a context which encapsulates the lemma.

3.6 Synset Insertion Module

The system presented previously produces Arabic synsets (groups of lemmas). These synsets have to be linked to the existing taxonomy of AWN. Before attempting to insert the obtained synsets in AWN, a preprocessing step is applied to keep only the synsets that are not present in AWN (i.e. the synsets that do not have an Arabic equivalent). Algorithm 2 is proposed to handle this insertion step. It relies on the fact that AWN and PWN have the same taxonomy. It takes as input the translated English synsets and returns as output a dictionary with two keys. Its primary key indicates the scan level and its secondary key provides the relevant synset. The pair of keys [scan level, synset] provides the synset parent(s) denoting its position in the AWN taxonomy. The proposed algorithm has two submodules: $subModule1$ and $subModule2$.

Algorithm 2. Synset insertion module

Input : TS: the set of translated synsets
Output: $dicAddedSynsets$: a dictionary of all the linked synsets with their parent nodes

Function `addSynsetsToTaxonomy(`TS`):`
  $dicAddedSynsets \leftarrow subModule1\ (TS)$
  $dicAddedSynsets \leftarrow subModule2\ (dicAddedSynsets, TS)$
return $(dicAddedSynsets)$

[7] The vocalization toolkit is available at https://github.com/Ycfx/Arabic-Diacritizer under the GNU License.

[8] We vocalize the entire sentence to obtain a more reliable vocalization since the vocalization model proposed in [22] uses a statistical model which tends to give better results when a wider context is available.

Algorithm 3 describes the first submodule, which performs the first scan of the translated synset. For each synset (line 3) we search for the parent(s)[9] in PWN. Then, for each parent (line 6) we search for an equivalent Arabic synset in AWN. We group all the parents with their Arabic equivalents in a list. If this list is empty (line 12), then the current translated synset cannot be added to the AWN taxonomy because it has no parent that already exists in the taxonomy. Otherwise, the position in AWN of the translated synset is found and the links are implicitly contained in the list of the parents that have Arabic equivalents. We return the result in a dictionary of added synsets in line 13.

Algorithm 3. First insertion submodule

Input : TS: a set of translated synsets
Output: *dicAddedSynsets*: a dictionary of all translated synsets directly linked to AWN

```
1  Function subModule1(TS):
2      //First scan
3      foreach synset s_j in TS do
4          Hypers = The parent(s) of s_j in PWN
5          HypersWithAREq = []
6          foreach parent Hyp_i in Hypers do
7              HypAR_i = An Arabic equivalent synset in AWN of s_j if one
                exists
8              if Hyp_i has an Arabic Equivalent Synset HypAR_i then
9                  HypersWithAREq.append((Hyp_i, HypAR_i))
10             end
11         end
12         if HypersWithAREq ≠ ∅ then
13             dicAddedSynsets[0][s_j] = HypersWithAREq
14             //The 0 refers to the scan level
15         end
16     end
17 return (dicAddedSynsets)
```

The algorithm for *subModule*2 is similar to Algorithm 3. It is used to handle the next scans. The remaining translated synsets have second chances to be linked to AWN. We follow the same approach as previously: we look for a parent(s) of the English synset but, this time, a synset will be linked if it has a parent(s) in the previously added synsets (the synsets that were added in the first scan). We repeat this process of trying to link the new remaining synsets to the previously added ones, until no further synsets can be added, i.e. until we obtain a scan level in the dictionary with no new synsets.

Figure 3 presents a sample of the added synsets with their hierarchical links and shows the equivalence between PWN and AWN.

[9] The WordNet taxonomy is not a strict hierarchy; in other words, a synset in the taxonomy can have multiple parents.

4 Experiments

In this section, we provide the details of the data we have used and preprocessing we have done. We then present the evaluation results obtained on the extended version of AWN that we have produced. We have applied a semi-automatic evaluation strategy.

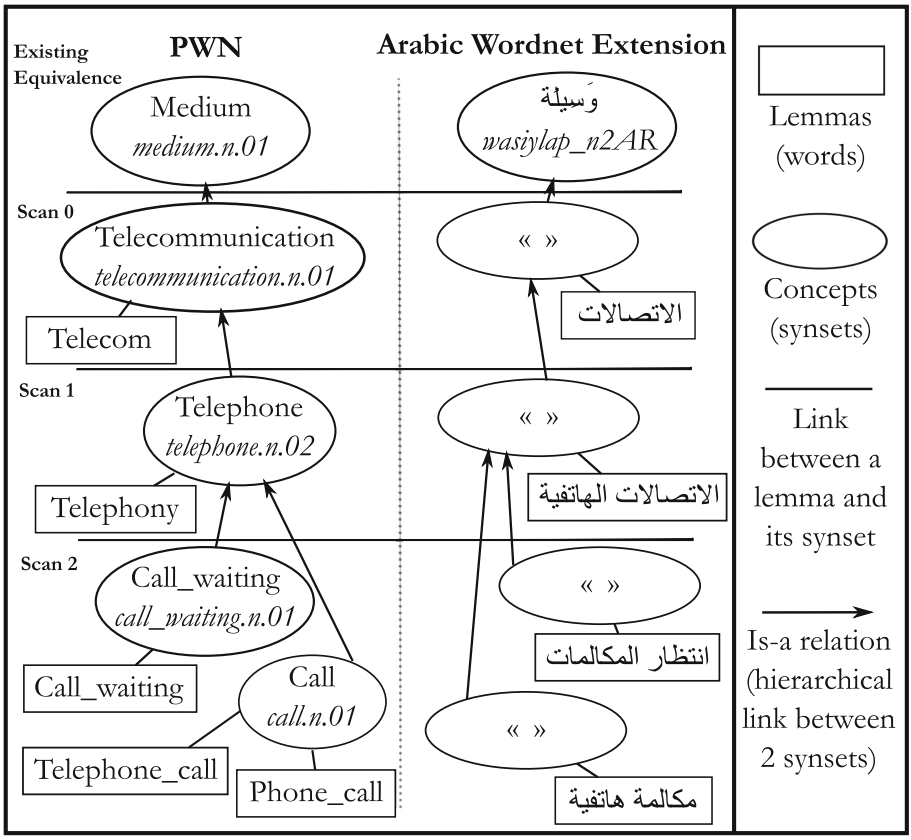


Fig. 3. A sample of the synsets added to AWN

4.1 Data and Preprocessing

We have used a set of four English-Arabic parallel corpora obtained from the "lingfil" website[10]. The statistics of these corpora are provided in Table 2. After the task of data filtering, we ended up with about 3 million parallel sentences which have then been preprocessed and prepared for word alignment. For the Arabic part of the bi-texts, the preprocessing has included diacritic signs removal

[10] http://opus.lingfil.uu.se.

Table 2. Statistics about the used English-Arabic parallel corpora

English-Arabic parallel corpus	Number of sentences
United nation	10.6M
Open subtitles	24.4M
News commentary	0.2M
IWSLT2016	0.2M
All	35.4M

and character normalization. Word segmentation has also been preformed by means of the MADAMIRA toolkit [24] using the ATB tokenization scheme, in which conjunctions, prepositions, suffixes and markers of the future tense were all separated. For the English part of the parallel corpus, we have only used word-tokenization by means of the python NLTK toolkit[11]. We have also added a number class <*nbr*> and a link class <*url*> to denote all numbers and links found in the parallel corpus, respectively. Sentence lengths have also been limited to 50 words for the Arabic as well as English sentences.

Word alignment has been done using the fast_align toolkit [25][12] which is based on the first and the second IBM alignment models [26]. For the external dictionaries used in this work the statistics are presented in Table 3 (see footnote 6).

Table 3. Sizes of the English-Arabic dictionaries used in the system

English-Arabic dictionary	Number of entries
Universal dictionary database	1646
Wiktionary database	2027
Omegawiki database	1008
Wikipedia interlanguage links	5426

4.2 Evaluation

The quality of our translation is controlled via a confidence threshold which is the lowest accepted value for the candidate score obtained with the multi-feature formula that was presented in Sect. 3.4. We note that the increase of the confidence threshold will decrease the number of admitted synsets. Figure 4 shows the statistics about the produced synsets when varying the value of the confidence threshold. To balance the number of produced synsets and the translation quality, we have set the confidence threshold to 0.7 which has resulted in about 24k synsets.

[11] http://www.nltk.org/.

[12] The code source for fast_alig is publicly available at http://github.com/clab/fastalign.

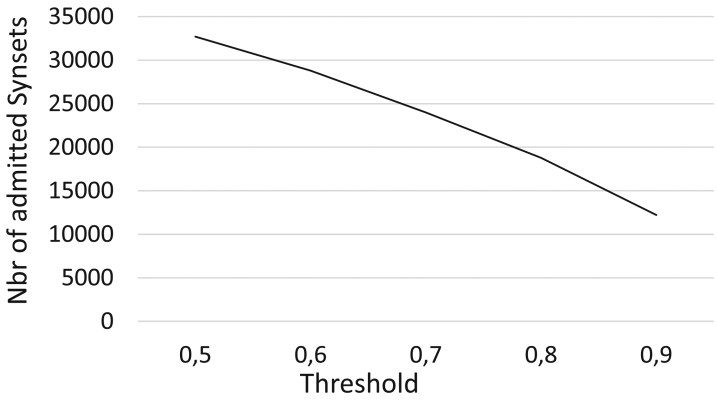


Fig. 4. Variation in the number of produced synsets as a function of the confidence threshold

We have considered the original AWN synsets as (gold) references for the parameter optimization (Sect. 3.4) as well as for the evaluation of the translations we have obtained. A set of 2850 AWN synsets has been considered, out of which 1700 have been used for parameter tuning (training) and 1150 for evaluation.

First, we have used an automatic evaluation that matches the produced translations with the AWN references using exact string matching. This has allowed us to validate around 60% of our test set which corresponds to 699 synsets. The remaining 40% (451 synsets) are the translations that are not identical string-wise to those of the AWN references. We note that these translations are not necessarily wrong; thus a manual evaluation has been carried out by human experts to validate them. Out of these synsets, 375 have been judged to be correct. We end up with a final precision of 93%. We note that this precision should reflect the overall system accuracy given that the test set can be seen as a random sub-sample of the translated synsets.

Around 12k of the translated 24k synsets that have been obtained with the confidence threshold of 0.7 have been successfully inserted via the insertion module. The statistics about the added synsets are provided in Table 4. The latter shows the number of added synsets in each scan level when applying the insertion module. We note that the projected precision of all the added synsets is around 93%.

Table 4. Statistics about the number of added synsets in each scan

Scan level	Added synsets	Scan level	Added synsets
0	9784	3	38
1	1761	4	1
2	252	5	0

4.3 Error Analysis

The percentage of wrong translations was about 7%. We have observed two kinds of errors. As an example of the first kind, the "amortize" and "amortise" lemmas which belong to the synset "amortize.v.01", have been translated to "استهلاك" and "سداد" which indeed express the same meaning but they are not verbs. So they cannot be considered as being correct translations. We note that the correct translations found in AWN are "استهلك الدين", "سدد" and "سدد اقساط" The second kind of errors is for those caused by a wrong alignment, which generally results in an incomplete translated lemma. This is the case for instance for the lemma "antimycotic" for which the alignment returns only "الفطريات" instead of the composed word "مضاد الفطريات".

5 Conclusion

In this work, we have presented a system for automatic WordNet enrichment. Machine translation has been used to automatically translate the Princeton WordNet (PWN) synsets to the target WordNet language. The novelty of this work in comparison to previous works is that our method has the advantage of using the wider resource of parallel corpora. Our method is also more general since it does not need to treat each type of lemma (named entities, adjectives, etc.) differently. We have also used multiple features for the selection of candidates which increase the chance of having the appropriate translation, and a metaheuristic for weight optimization to ensure that the system will favour the most relevant features. We have applied our proposal to enrich the Arabic WordNet (AWN) by adding new synsets along with their vocalizations and examples that illustrate how they can be used. We have managed to add a total of 12k potential synsets to the AWN taxonomy with a precision of 93%.

This work can be developed in various directions. First, the added synsets need to be validated by multiple linguistic experts to create a fully correct and reliable WordNet. Second, this work has only tackled the non-ambiguous lemmas; the case of ambiguous lemmas can be handled using Word Sense Disambiguation (WSD) methods to identify their senses according to their occurrence contexts. Third, synset labeling can be incorporated to provide adequate labels (names) for the added synsets. Finally, the incorporation of richer parallel corpora can be very useful to our approach.

References

1. Morato, J., Marzal, M.A., Lloréns, J., Moreiro, J.: Wordnet applications. In: Proceedings of GWC, pp. 20–23 (2004)
2. Fellbaum, C.: WordNet. Wiley Online Library, New York (1998)
3. Black, W., Elkateb, S., Rodriguez, H., Alkhalifa, M., Vossen, P., Pease, A., Fellbaum, C.: Introducing the Arabic wordnet project. In: Proceedings of the Third International WordNet Conference, pp. 295–300 (2006)

4. Rodríguez, H., Farwell, D., Ferreres, J., Bertran, M., Alkhalifa, M., Martí, M.A.: Arabic wordnet: Semi-automatic extensions using bayesian inference. In: LREC (2008)
5. Alkhalifa, M., Rodríguez, H.: Automatically extending ne coverage of arabic wordnet using wikipedia. In: Proceedings of the 3rd International Conference on Arabic Language Processing CITALA2009, Rabat, Morocco (2009)
6. Abouenour, L., Bouzoubaa, K., Rosso, P.: On the evaluation and improvement of Arabic wordnet coverage and usability. Lang. Resour. Eval. **47**(3), 891–917 (2013)
7. Saveski, M., Trajkovski, I.: Automatic construction of wordnets by using machine translation and language modeling. In: Proceedings of the 13th International Multiconference Information Society on Seventh Language Technologies Conference, vol. C (2010)
8. Cilibrasi, R.L., Vitanyi, P.M.: The google similarity distance. IEEE Trans. Knowl. Data Eng. **19**(3), 370–383 (2007)
9. Montazery, M., Faili, H.: Automatic persian wordnet construction. In: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, pp. 846–850. Association for Computational Linguistics (2010)
10. Mousavi, Z., Faili, H.: Persian Wordnet Construction using Supervised Learning. ArXiv e-prints, April 2017
11. Niemi, J., Lindén, K., Hyvärinen, M., et al.: Using a bilingual resource to add synonyms to a wordnet. In: Proceedings of the Global Wordnet Conference (2012)
12. Lindén, K., Niemi, J.: Is it possible to create a very large wordnet in 100 days? an evaluation. Lang. Resour. Eval. **48**(2), 191–201 (2014)
13. Al Tarouti, F., Kalita, J.: Enhancing automatic wordnet construction using word embeddings. In: Proceedings of the Workshop on Multilingual and Cross-lingual Methods in NLP (2016)
14. Cavalli-Sforza, V., Saddiki, H., Bouzoubaa, K., Abouenour, L., Maamouri, M., Goshey, E.: Bootstrapping a wordnet for an arabic dialect from other wordnets and dictionary resources. In: 2013 ACS International Conference on Computer systems and Applications (AICCSA), pp. 1–8. IEEE (2013)
15. Boudabous, M.M., Kammoun, N.C., Khedher, N., Belguith, L.H., Sadat, F.: Arabic wordnet semantic relations enrichment through morpho-lexical patterns. In: 2013 1st International Conference on Communications, Signal Processing, and their Applications (ICCSPA), pp. 1–6. IEEE (2013)
16. Al-Yahya, M., Al-Malak, S., Aldhubayi, L.: Ontological lexicon enrichment: The badea system for semi-automated extraction of antonymy relations from arabic language corpora. Malays. J. Comput. Sci. **29**(1), 56–73 (2016)
17. Koehn, P.: Statistical Machine Translation. Cambridge University Press, Cambridge (2009)
18. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 311–318. Association for Computational Linguistics (2002)
19. Powers, D.M.: Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. J. Mach. Learn. Technol. **2**(1), 37–63 (2011)
20. Glover, F.: Future paths for integer programming and links to artificial intelligence. Comput. Oper. Res. **13**(5), 533–549 (1986)
21. Glover, F.: Tabu search—part I. ORSA J. Comput. **1**(3), 190–206 (1989)

22. Ameur, M.S.H., Moulahoum, Y., Guessoum, A.: Restoration of Arabic diacritics using a multilevel statistical model. In: Amine, A., Bellatreche, L., Elberrichi, Z., Neuhold, E.J., Wrembel, R. (eds.) CIIA 2015. IAICT, vol. 456, pp. 181–192. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-19578-0_15
23. Diab, M.: Second generation AMIRA tools for arabic processing: Fast and robust tokenization, POS tagging, and base phrase chunking. In: 2nd International Conference on Arabic Language Resources and Tools, vol. 110 (2009)
24. Pasha, A., Al-Badrashiny, M., Diab, M.T., El Kholy, A., Eskander, R., Habash, N., Pooleery, M., Rambow, O., Roth, R.: MADAMIRA: A fast, comprehensive tool for morphological analysis and disambiguation of arabic. In: LREC, vol. 14, pp. 1094–1101 (2014)
25. Dyer, C., Chahuneau, V., Smith, N.A.: A simple, fast, and effective reparameterization of IBM model 2. Association for Computational Linguistics (2013)
26. Brown, P.F., Pietra, V.J.D., Pietra, S.A.D., Mercer, R.L.: The mathematics of statistical machine translation: Parameter estimation. Comput. Linguist. **19**(2), 263–311 (1993)

Word Embedding-Based Approaches for Measuring Semantic Similarity of Arabic-English Sentences

El Moatez Billah Nagoudi[1(✉)], Jérémy Ferrero[2,3], Didier Schwab[3], and Hadda Cherroun[1]

[1] Laboratoire d'Informatique et de Mathématique LIM, Amar Telidji University, Laghouat, Algeria
{e.nagoudi,h.cherroun}@lagh-univ.dz
[2] Compilatio, 276 rue du Mont Blanc, 74540 Saint-Félix, France
[3] LIG-GETALP, Univ. Grenoble Alpes, Grenoble, France
{jeremy.ferrero,didier.schwab}@imag.fr

Abstract. Semantic Textual Similarity (STS) is an important component in many Natural Language Processing (NLP) applications, and plays an important role in diverse areas such as information retrieval, machine translation, information extraction and plagiarism detection. In this paper we propose two word embedding-based approaches devoted to measuring the semantic similarity between Arabic-English cross-language sentences. The main idea is to exploit Machine Translation (MT) and an improved word embedding representations in order to capture the syntactic and semantic properties of words. MT is used to translate English sentences into Arabic language in order to apply a classical monolingual comparison. Afterwards, two word embedding-based methods are developed to rate the semantic similarity. Additionally, Words Alignment (WA), Inverse Document Frequency (IDF) and Part-of-Speech (POS) weighting are applied on the examined sentences to support the identification of words that are most descriptive in each sentence. The performances of our approaches are evaluated on a cross-language dataset containing more than 2400 Arabic-English pairs of sentence. Moreover, the proposed methods are confirmed through the Pearson correlation between our similarity scores and human ratings.

Keywords: Semantic sentences similarity · Cross-language Arabic-English · Machine translation · Word embedding

1 Introduction

Semantic Textual Similarity (STS) is the task of measuring the degree of semantic equivalence between two textual units (texts, paragraphs or sentences) [1]. STS is a core field of Natural Language Processing (NLP) and plays an important role in several application areas, such as Information Retrieval (IR), Word

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 19–33, 2018.
https://doi.org/10.1007/978-3-319-73500-9_2

Sense Disambiguation (WSD), Question Answering (QA), and Text Summarization (TS) among others. There are two known types of STS: monolingual and cross-language [3]. The first one estimates the degree to which the underlying semantics of two textual units written in the same language, are equivalent to each other, while the STS cross-language aims to quantify the degree to which two textual units are semantically related, independent of the languages they are written in [15].

Determining the similarity between sentences has been extensively reviewed in a monolingual domain [4,20,37,43]. While cross-language semantic similarity is relatively more difficult to identify since the relatedness of words are investigated between two different languages [15]. Thus, it is necessary to address this task to enhance the performance in several applications, such as Machine Translation (MT), Cross-Language Plagiarism Detection (CLPD) and Cross-Language Information Retrieval (CLIR).

In this paper we focus our investigation on measuring the semantic similarity between Arabic-English cross-language sentences using machine translation and word embedding representations. We also consider words alignment, term frequency weighting and Part-of-Speech tagging to improve the identification of words that are highly descriptive in each sentence.

The rest of this paper is organized as follows, the next section describes work related to STS cross-language detection and word embedding models. In Sect. 3, we present our proposed cross-language word embedding-based methods. Section 4 describes the experimental results of these systems. Finally, our conclusion and some future research directions are drawn in Sect. 5.

2 Related Work

In this section, we review the most relevant approaches for measuring cross-language semantic textual. Then, we study those dedicated to the Arabic-English semantic similarity. Finally, we recall some concepts related to word embedding.

2.1 Cross-Language Semantic Textual Similarity Detection

In the literature, many approaches are proposed for cross-language textual similarity detection. We can classify them according to the strategy they used to detect such similarity into five classes: Syntax-Based, Dictionary-Based, Parallel and Comparable Corpora-Based and MT-Based Models [10]. Figure 1 shows the taxonomy of different approaches for cross-language similarity detection. In the following, we will review the most commonly used methods.

Concerning the syntax-based models, the key idea lies in comparing multilingual texts without translation. For instance, Pouliquen et al. [16] have proposed a *"Length Model"* to estimate cross-language text similarity. It is mainly based on comparing the texts size. They observed the fact that the length of texts in different languages are closely linked by a factor, and there is a different factor for each language pair. McNamee and Mayfield [22] have introduced Cross-Language

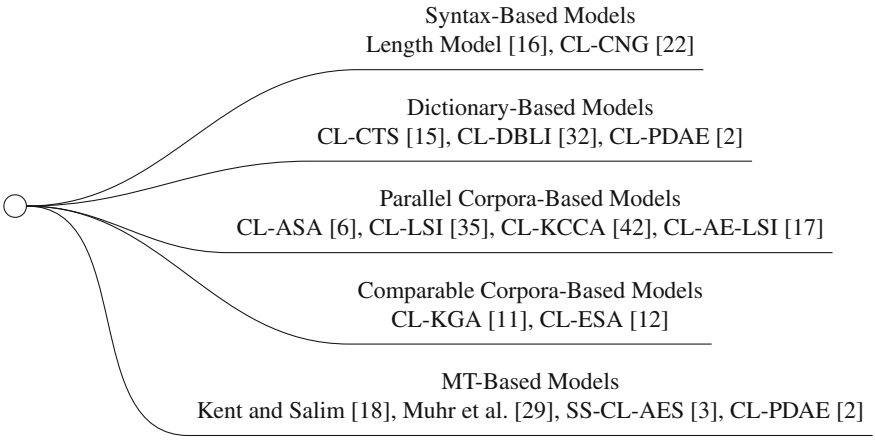


Fig. 1. Taxonomy of different approaches for cross-language similarity detection [10].

Character N-Gram (CL-CNG) model to compare two textual units by using their n-gram vectors representation. This technique achieves a good performance with languages that are close to each other, because of common root words.

In dictionary-based models, the semantic similarity is measured by constructing a vector space model of the textual units. For that, a vector of concepts is built for each textual unit using dictionaries or thesaurus. The similarity between the vectors of concepts can be measured using the Cosine similarity, Euclidean distance, or any other similarity measure. In [15] a Cross-Language Conceptual Thesaurus-Based Similarity model (CL-CTS) is proposed to measure the similarity between textual units written in different languages (Spanish, English and German). CL-CTS is based on the thesaurus concepts vectors presented in Eurovoc[1] where a Cosine similarity is computed between these vectors. In the same context, Pataki [32] have proposed a Cross-Language Dictionary-Based Language-Independent (CL-DBLI) model. CL-DBLI considers a translation synonym dictionary to extract the abstract concepts from words in textual units.

For comparable corpora-based models, Gabrilovich and Markovitch [12] have presented a Cross-Language Explicit Semantic Analysis (CL-ESA) model. CL-ESA is based on the Explicit Semantic Analysis (ESA), which represent the meaning of text by a vector of concepts derived from Wikipedia. In a cross-lingual context, Potthast et al. [36] use Wikipedia as comparable corpus to estimate the similarity of two documents by calculating the similarity of their two ESA representations. Another model called Cross-Language Knowledge Graph Analysis (CL-KGA), is introduced for the first time by Franco-Salvador et al. [11]. CL-KGA uses knowledge graphs built from multilingual semantic network (the authors use BabelNet [31]) to represent texts, and then compare them in a common lingual semantic graph space.

[1] http://eurovoc.europa.eu/.

Regarding parallel corpora-based models, several approaches are proposed. For instance, Barrón-Cedeño et al. [6] have introduced a Cross-Language Alignment Similarity Analysis (CL-ASA) approach. CL-ASA estimates the similarity between two textual units using bilingual statistical dictionary extracted from parallel corpus. The same idea was used independently by Pinto et al. [34]. A Cross-Language Latent Semantic Indexing model (CL-LSI) is developed by Potthast et al. [35]. CL-LSI uses a parallel corpora with the common Latent Semantic strategy applied in IR systems for term-textual unit association. Another model named Cross-Language Kernel Canonical Correlation Analysis (CL-KCCA) model due to Vinokourov et al. [42], it analyzes the correspondences between two LSI spaces to measure the correlation of the respective projection values.

The main idea of the machine translation-based models consists in using MT tools to translate textual units into the same language (pivot language) in order to apply a monolingual comparison between them [5]. For this purpose, Kent and Salim [18] have used Google Translate API to translate texts, while Muhr et al. [29] replace each word of the original text by its most likely translations in the target language.

2.2 Arabic-English Cross-Language Semantic Similarity

In context of the Arabic-English cross-language semantic similarity, Hattab [17] has used Latent Semantic Indexing (LSI) to build cross-language Arabic-English semantic space (CL-AE-LSI), from which it checks the contextual similarity of two given texts, one in Arabic and the other in English.

Recently, Alzahrani [3] presented two models of Semantic Similarity for Arabic-English Cross-Language Sentences (SS-CL-AES). The first one used a dictionary-based translation, where an Arabic sentence is translated into English terms, then the semantic similarity is computed by using the maximum-translation similarity technique. In the second model, MT is applied on the Arabic sentence. After that, the algorithms proposed by Lee [19], and Liu et al. [21] are used to calculate the semantic similarity.

Alaa et al. [2] are interested in Cross-Language Plagiarism Detection of Arabic-English documents (CL-PDAE). In fact, after a candidate document retrieval step by key phrase extraction, they translate a source text by getting for a word all the available translations of all its available synonyms from WordNet [27], and then they use a combination of monolingual measures (Longest Common Subsequence (LCS), Cosine similarity and N-Gram) to detect similar phrases.

2.3 Word Embedding-Based Models

Recently, Word Embedding (WE) technique has received a lot of attention in the NLP community and has become a core building to many NLP applications. WE represents words as vectors in a continuous high-dimensional space. These representations allow capturing semantic and syntactic properties of the

language [23]. In the literature, several techniques are proposed to build word embedding models.

For instance, Collobert and Weston [9] have presented a unified system based on a deep neural network, and jointly trained with many NLP tasks, such as: POS tagging, Semantic Role Labeling and Named Entity Recognition. Their model is stored in a matrix $M \in R^{d*|D|}$, where D represents the dictionary of all unique words in the training data, and each word in D is embedded into a *d-dimensional* vector. The sentences are represented using the embeddings of their forming words. A similar idea was independently proposed and used by Turian et al. [41].

Mnih and Hinton [28] have introduced another form to represent words in vector space, named Hierarchical Log-Bilinear Model (HLBL). Like almost all neural language models, the HLBL model is used to represent each word by a real-valued feature vector. HLBL concatenates the $(n-1)$ first embedding words $(w_1 \dots w_{n-1})$ and learns a neural linear model to predicate the last word w_n.

In Mikolov et al. [26] a recurrent neural network (RNN) [24] is used to build a neural language model. The RNN model encode the context word by word and predict the next word. Afterwards, the weights of the trained network are considered as the word embedding vectors.

Based on the simplified neural language model of Bengio et al. [7], Mikolov et al. [23,25] presented two other techniques to build a words representations model. In their work, two models are proposed: the continuous bag-of-words (CBOW) model [23], and the skip-gram (SKIP-G) model [25]. The CBOW model, predicts a pivot word according to the context by using a window of contextual words around it. Given a sequence of words $S = w_1, w_2, ..., w_i$, the CBOW model learns to predict all words w_k from their surrounding words $(w_{k-l}, ..., w_{k-1}, w_{k+1}, ..., w_{k+l})$. The second model, SKIP-G, predicts surrounding words of the current pivot word w_k [25].

Pennington et al. [33] proposed a Global Vectors (GloVe) model to representing words in vector space. GloVe model builds a co-occurrence matrix M using the global statistics of word-word co-occurrence. Afterwards, the matrix M is used to estimate the probability of word w_i to appear in the context of another word w_j, this probability $P(i/j)$ represents the relationship between words.

In a comparative study conducted by Mikolov et al. [23] all the methods [9, 23,25,26,28,41] have been evaluated and compared, and they show that CBOW [23] and SKIP-G [25] models are significantly faster to train with better accuracy. For this reason, we have used the CBOW word representations for Arabic model, proposed by Zahran et al. [45]. In order to train this model, they have used a large collection from different sources counting more than 5.8 billion words[2].

In the Arabic CBOW model [45] each word w is represented by a vector v of *d-dimension*. The similarity between two words w_i and w_j (e.g. synonyms, singular, plural, feminization or closely related semantically) is obtained by comparing their vector representations v_i and v_j respectively [23]. This similarity can be evaluated using the Cosine similarity, Euclidean distance, Manhattan distance

[2] https://sites.google.com/site/mohazahran/data.

or any other similarity measure functions. For example, let "الجامعة" (*university*), "المساء" (*evening*) and "الكلية" (*faculty*) be three words. The similarity between them is measured by computing the cosine similarity between their vector as follows:

$$Sim(\text{المساء}, \text{الجامعة}) = Cos(v(\text{المساء}), v(\text{الجامعة})) = 0.13$$

$$Sim(\text{الكلية}, \text{الجامعة}) = Cos(v(\text{الكلية}), v(\text{الجامعة})) = 0.72$$

That means that, the words "الكلية" (*faculty*) and "الجامعة" (*university*) are semantically closer than "المساء" (*evening*) and "الجامعة" (*university*). In the following, we exploit this property to measure the semantic similarity at sentence level.

3 Proposed Methods

In this section, we present our two proposed methods for Arabic-English cross-language sentence similarity. These methods use Machine Translation-Based Model, followed by a monolingual semantic similarity analysis based on word embedding. They consist of three steps, including translation, preprocessing and similarity score attribution. First, MT is used to translate English sentences into Arabic. Afterwards, our two word embedding-based methods are employed to measure the semantic similarity of Arabic sentences. In the first one, we propose to use the words alignment technique proposed by Sultan et al. [39] with the words weighting methods of Nagoudi and Schwab [30], we call this method *Weighting Aligned Words* (W-AW). The second generate a Bag-of-Words for the aligned words to construct a vector representation of each sentence. Then the similarity is obtained by comparing the two sentence vectors, we name this method *Bag-of-Words Alignment* (BoW-A). Figure 2 gives an overview of the proposed methods.

Let $S_E = w_{e_1}, w_{e_2}, ..., w_{e_i}$ and $S_A = w_{a_1}, w_{a_2}, ..., w_{a_j}$ be an English and Arabic sentence, and their word vectors are $(v_{e_1}, v_{e_2}, ..., v_{e_i})$ and $(v_{a_1}, v_{a_2}, ..., v_{a_j})$ respectively. The semantic similarity between S_E and S_A is computed in three steps: translation, preprocessing and a monolingual similarity score attribution.

(1) **Translation:** in this step, we used Google Translate API[3] to translate English sentences into Arabic language, we denote the translated sentence $S_{E'}$. By this translation, the problem is reduced into a mono-lingual semantic similarity one.
(2) **Preprocessing:** in order to normalize the sentences for the similarity evaluation step, a set of preprocessing are performed:
   – Tokenization: input sentences are broken up into words;
   – Removing punctuation marks, diacritics, and non alphanumeric characters;
   – Normalizing أ ، إ ، آ to ا and ة to ه , as in the Arabic CBOW model [45];
   – Replacing final ى followed by ء by ئ.

[3] https://cloud.google.com/translate/.

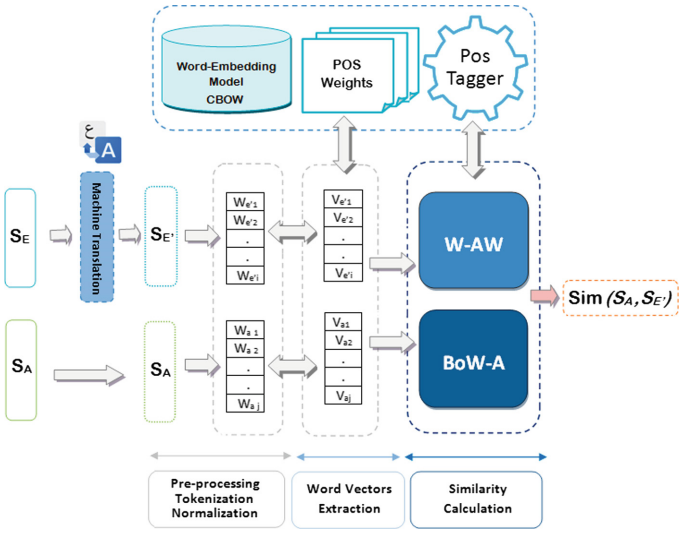


Fig. 2. Overview of the proposed methods

At this point we should mention that we will not remove the stop words because they can affect the similarity score, For example: S_E ="Joseph went to university" and S_A ="يوسف لم يذهب إلى الجامعة" (Joseph does not go to university). If we remove the words لم, إلى and **to** as a stop words, both sentences become similar, whereas they have contradictory meanings.

(3) **Sentences similarity:** we propose two methods for measuring the semantic similarity between $S_{E'}$ and S_A: Weighting Aligned Words Method (W-AW) and Bag-of-Words Alignment Method (BoW-A). In the following, we develop our proposed methods, and we provide for each one how the semantic similarity is measured.

3.1 Weighting Aligned Words Method (W-AW)

A simple way to compare the translated sentence $S_{E'}$ and the Arabic one S_A is by summing their words vectors [30]. Then, the similarity is obtained by calculating the Cosine similarity $Cos(V_{E'}, V_A)$, where:

$$\begin{cases} V_{E'} = \sum_{k=1}^{i} v_{e'_k} \\ V_A \ = \sum_{k=1}^{j} v_{a_k} \end{cases}$$

For example, let S_E and S_A be two sentences:

S_E = "Joseph went to college".

S_A = "يوسف يمضى مسرعا للجامعة" (*Joseph goes quickly to university*).

Step 1: Translation
In this step Google Translate API is used to translate the English sentence S_E into Arabic $S_{E'}$ = "ذهب يوسف إلى الكلية".
Step 2: Sum of the word vectors

$$V_{E'} = V(\text{الكلية}) + V(\text{إلى}) + V(\text{يوسف}) + V(\text{ذهب})$$
$$V_A = V(\text{للجامعة}) + V(\text{مسرعا}) + V(\text{يمضي}) + V(\text{يوسف})$$

Step 3: Similarity Score
The similarity between $S_{E'}$ and S_A is obtained by calculating the cosine similarity between the sentence vectors $V_{E'}$ and V_A as follows:

$$Sim(S_E, S_A) = Sim(S_{E'}, S_A) = Cos(V_{E'}, V_A) = 0.71$$

In order to improve the similarity results, we have used the words alignment method presented by Sultan et al. [39], with the difference that we align the words based on their semantic similarity in the word embedding model, and not in a dictionary. We assume also that all words don't have the same importance for the meaning of the sentences. For that, we use three weighting functions (*idf*, *pos* and *idf-pos*) proposed by Nagoudi and Schwab in [30] for weighting the aligned words. Finally, the similarity between $S_{E'}$ and S_A is calculated as follows:

$$Sim(S_{E'}, S_A) = \frac{1}{2}\left(\frac{\sum_{w \in S_{E'}} WT(w) * BM(w, S_A)}{\sum_{w \in S_{E'}} WT(w)} + \frac{\sum_{w \in S_A} WT(w) * BM(w, S_{E'})}{\sum_{w \in S_A} WT(w)}\right) \quad (1)$$

where $WT(w)$ is the function which return the weight of the word w. WT uses three weighting methods: *idf*, *pos* and a mix of both. The $BM(w, S_k)$ function represent the *Best Match* score between w and all words in the sentence S_k. Therefore, BM function aligns words based on their semantic similarity included in the word embedding model. The function BM is defined as:

$$BM(w, S_k) = Max\{Cos(v, v_k),\ w_k \in S_k\} \quad (2)$$

For example, let us continue with the same example above, the similarity between $S_{E'}$ and S_A is obtained in four steps as follows:

Step 1: POS Tagging
Firstly, the POS tagger of Gahbiche-Braham et al. [13] is used to predict the part-of-speech tag of each word w_k in S_k.

$$\begin{cases} Pos\_tag(S_{E'}) = verb\ noun\_prop\ noun \\ Pos\_tag(S_A) = noun\_prop\ verb\ adj\ noun \end{cases}$$

Step 2: IDF & POS Weighting
For weighting the descriptive aligned words, we retrieve for each word w_k in the S_k its IDF weight $idf(w_k)$, we also use the POS weights proposed in [30].
Step 3: Words Alignment
In this step, we align words that have similar meaning in both sentences. For that, we compute the similarity between each word in $S_{E'}$ and the semantically closest word in S_A by using the BM function, e.g. $BM(\text{يمضي}, S_{E'}) = Max\{Cos(\text{يمضي}, v_k),\ w_k \in S_A\} = Cos(v(\text{يمضي}), v(\text{ذهب})) = 0.85$.

Step 4: Calculate the similarity
The similarity between $S_{E'}$ and S_A is obtained by using the formula (1), which gives us: $Sim(S_{E'}, S_A) = 0.82$.

3.2 Bag-of-Words Alignment Method (BoW-A)

Among the advantages of word embedding is that it allows to retrieve a list of words that are used in the same contexts with respect to a given word w [14]. We named this list of words the *k-closest* words to w. For example, Table 1 shows the 10-closest words of الجامعة and الكلية in the Arabic CBOW model.

Table 1. 10-closest words of الجامعة and الكلية.

BoW(الجامعة)	BoW(الكلية)
الجامعات, للجامعة, الكلية, بالجامعة, العمادة, جامعة, الاكاديمية, جامعتنا, حرم الجامعة, الجامعات	الاكاديمية, الجامعة, كلية الطب, كليتنا, للجامعة, جامعة, الجامعات, للجامعة, كليات الجامعة, العمادة

We used this property to evaluate the degree of semantic similarity between $S_{E'}$ and S_A, we first proceeded to construct a representation vector RV for each sentence. Let $RV_{E'}$ and RV_A be the representation vectors of $S_{E'}$ and S_A respectively, the size of each vector is the number of words in its corresponding sentence. The value of an entry in the representation vector, is determined as follows:

1. For each word w we retrieve its aligned word w' in the other sentence by using BM function defined by formula (2).
2. We use the embedding model to construct for both w and w' their Bag-of-Words BoW_w and $BoW_{w'}$. The BoW_w ($BoW_{w'}$) contains the *k-closest* words to w (w') in the embedding model.
3. We compute the Jaccard similarity between BoW_w and $BoW_{w'}$:

$$Jacc(BoW_w, BoW_{w'}) = \frac{BoW_w \cap BoW_{w'}}{BoW_w \cup BoW_{w'}}$$

4. The value of the entry RV[w] is set to $Jacc(BoW_w, BoW_{w'})$.
5. This process is applied for all words in both sentences to build $RV_{E'}$ and RV_A.
6. Finally, the similarity between $S_{E'}$ and S_A is obtained by:

$$Sim(S_{E'}, S_A) = \frac{1}{2}\left(\frac{\sum_{w \in S_{E'}} WT(w) * RV[w]}{\sum_{w \in S_{E'}} WT(w)} + \frac{\sum_{w \in S_A} WT(w) * RV[w']}{\sum_{w \in S_A} WT(w)}\right) \quad (3)$$

4 Experiments and Results

In order to evaluate our systems and monitor their performances, we have used four datasets drawn from the STS shared task SemEval-2017 (Task1: STS Cross-lingual Arabic-English)[4] [8], with a total of 2412 pairs of sentences. The sentence pairs have been manually labeled by five annotators, and the similarity score is the mean of the five annotators' judgments. This score is a float number between "0" (indicating that the meaning of sentences are completely independent) to "5" (indicating meaning equivalence). More information about the datasets used is listed in Table 2.

Table 2. Arabic-English evaluation sets.

Dataset	Source	Pairs
MSRvid	Microsoft research video description corpus	736
MSRpar	Microsoft research paraphrase corpus	1020
SMTeuroparl	WMT2008 development dataset	406
STS evaluation data	SNLI corpus	250

4.1 Experimental Results

We investigated the performance of both Weighting Aligned Words (W-AW) and Alignment Bag-of-Words (A-BoW) systems with three weighting functions: IDF, POS and mix of both. In addition, for the A-BoW method, we have used four different values of k to generate the 5-*closest*, 10-*closest*, 15-*closest* and 20-*closest* words. Afterwards, in order to evaluate the accuracy of each method, we calculate the Pearson correlation between our assigned semantic similarity scores and human judgments on the SemEval STS task datasets. Table 3 reports the results of the proposed methods.

These results indicate that when the IDF weighting method is used the mean correlation rate does not fall below 70% in all tested methods. When applying the POS and mixed weighting, the correlation rate of IDF weighting is outperformed in both methods A-AW and A-BoW with a mean of +2.35% and +3.91% respectively. Interestingly, increasing the parameter k to generate the k-*closest* words in the A-BoW method, leads each time to an enhancement in the correlation rate. For instance, the use of 15-*closest* words outperforms the 5-*closest* system by +2.01% of correlation in average. However, when k is raised to 20, the mean correlation rate gets a bit lower. This is due to the rise of the number of words with different meaning in the BoW.

From the above results, we can see that the estimated similarity provided by our approaches is fairly consistent with human judgments. However, the correlation is not good enough when two sentences share nearly the same words, but with a totally different meaning, for example:

[4] http://alt.qcri.org/semeval2017/task1/index.php?id=data-and-tools.

Table 3. Our methods vs. human judgments

Method	MSRvid	MSRpar	SMTeuro.	STS Eval.	Mean
W-AW-IDF	0.6895	0.7019	0.7274	0.6951	0.7034
W-AW-POS	0.6924	0.7402	0.7478	0.7205	0.7252
W-AW-IDF-POS	0.7015	0.7385	0.7512	0.7375	**0.7321**
$k = 5$					
A-BoW-IDF	0.6863	0.7119	0.7174	0.6881	0.7009
A-BoW-POS	0.6933	0.7349	0.7364	0.7187	0.7218
A-BoW-IDF-POS	0.7074	0.7365	0.7482	0.7362	**0.7320**
$k = 10$					
A-BoW-IDF	0.6879	0.7131	0.7291	0.7114	0.7103
A-BoW-POS	0.7084	0.7437	0.7514	0.7305	0.7335
A-BoW-IDF-POS	0.7216	0.7418	0.7603	0.7565	**0.7450**
$k = 15$					
A-BoW-IDF	0.6954	0.7089	0.7284	0.7254	0.7145
A-BoW-POS	0.7124	0.7402	0.7578	0.7391	0.7398
A-BoW-IDF-POS	0.7575	0.7485	0.7672	0.7739	**0.7603**
$k = 20$					
A-BoW-IDF	0.6912	0.7055	0.7283	0.7254	0.7244
A-BoW-POS	0.7254	0.7382	0.7514	0.7351	0.7351
A-BoW-IDF-POS	0.7525	0.7477	0.7689	0.7613	**0.7576**

"يقرأ سعد كتابا عن عمر بن الخطاب" and *(Saad reads a book about Omar Ibn Al-Khattab)* "سعد يقرأ كتابا لعمر بن الخطاب" *(Saad reads a book for Omar Ibn Al-Khattab)*. In this example, the sentences share the same vectors, POS and IDF weights. This fact leads to a high correlation score, which is not the case. This issue is left for future work.

4.2 Comparison with SemEval-2017 Winners

We compared our optimal results with the three best systems proposed in SemEval-2017 Arabic-English cross-lingual evaluation task [8] (ECNU [40], BIT [44] and HCTI [38]) and the baseline system [8]. In this evaluation, ECNU obtained the best performance with a correlation score of 74.93%, followed by BIT and HCTI with 70.07% and 68.36% respectively. Table 4 shows a comparison of our best results with those obtained by the three systems were tested on the STS Evaluation Data[5].

The observed results indicate that our mixed weighted method with $k = 15$ is the best performing method with a correlation rate of 77.39%. The W-BoW-

[5] http://alt.qcri.org/semeval2017/task1/data/uploads/sts2017.eval.v1.1.zip.

Table 4. Comparison of the correlation results with three best systems in SemEval-2017.

Methods	STS Eval.
W-BoW-IDF-POS ($k = 15$)	**77.39%**
ECNU	74.93 %
W-AW-IDF-POS	73.75%
BIT	70.07 %
HCTI	68.36%
Cosine baseline	51.55 %

IDF-POS ($k = 15$) method yields a gain of +9.03%, +7.32% and +2.46% on the correlation rate compared with ECNU, BIT and HCTI respectively.

5 Conclusion and Future Work

In this paper, we have presented two methods for measuring the semantic relations between Arabic-English cross-language sentences using Machine Translation (MT) and word embedding representations. The main idea is based on the usage of semantic properties of words included in the word-embedding model. In order to make further progress in the analysis of the semantic sentence similarity, we have used a combination of words alignment, IDF and POS weighting to support the identification of words that are most descriptive in each sentence. Additionally, we evaluated our proposals on the four datasets of the STS shared task SemEval-2017. In the experiments we have shown how the Bag-of-words method clearly enhanced the correlation results. The performance of our proposed methods was confirmed through the Pearson correlation between our assigned semantic similarity scores and human judgments. In fact, we reached the best correlation rate compared to all the participating systems in STS Arabic-English cross-language subtask of SemEval-2017. As future work, we are going to combine these methods with those of other classical techniques in NLP field, including word sense disambiguation, linguistic resources and document fingerprint in order to make more improvement in the cross-language plagiarism detection.

References

1. Agirre, E., Banea, C., Cer, D., Diab, M., Gonzalez-Agirre, A., Mihalcea, R., Rigau, G., Wiebe, J.: Semeval-2016 task 1: semantic textual similarity, monolingual and cross-lingual evaluation. In: Proceedings of SemEval, pp. 497–511 (2016)
2. Alaa, Z., Tiun, S., Abdulameer, M.: Cross-language plagiarism of Arabic-English documents using linear logistic regression. J. Theor. Appl. Inf. Technol. **83** (2016)
3. Alzahrani, S.: Cross-language semantic similarity of Arabic-English short phrases and sentences. J. Comput. Sci. **12**, 1–18 (2016)

4. Bär, D., Biemann, C., Gurevych, I., Zesch, T.: UKP: computing semantic textual similarity by combining multiple content similarity measures. In: Proceedings of the First Joint Conference on Lexical and Computational Semantics, pp. 435–440. Association for Computational Linguistics (2012)
5. Barrón-Cedeño, A., Gupta, P., Rosso, P.: Methods for cross-language plagiarism detection. Knowl. Based Syst. **50**, 211–217 (2013)
6. Barrón-Cedeno, A., Rosso, P., Pinto, D., Juan, A.: On cross-lingual plagiarism analysis using a statistical model. In: PAN, pp. 1–10 (2008)
7. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. J. Mach. Learn. Res. **3**, 1137–1155 (2003)
8. Cer, D., Diab, M., Agirre, E., Lopez-Gazpio, I., Specia, L.: Semeval-2017 task 1: semantic textual similarity multilingual and crosslingual focused evaluation. In: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017), Vancouver, Canada, pp. 1–14. Association for Computational Linguistics, August 2017
9. Collobert, R., Weston, J.: A unified architecture for natural language processing: deep neural networks with multitask learning. In: Proceedings of the 25th International Conference on Machine learning, pp. 160–167. ACM (2008)
10. Ferrero, J., Agnes, F., Besacier, L., Schwab, D.: A multilingual, multi-style and multi-granularity dataset for cross-language textual similarity detection. In: 10th Edition of the Language Resources and Evaluation Conference (2016)
11. Franco-Salvador, M., Gupta, P., Rosso, P.: Cross-language plagiarism detection using a multilingual semantic network. In: Serdyukov, P., Braslavski, P., Kuznetsov, S.O., Kamps, J., Rüger, S., Agichtein, E., Segalovich, I., Yilmaz, E. (eds.) ECIR 2013. LNCS, vol. 7814, pp. 710–713. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-36973-5_66
12. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007), Hyderabad, India, pp. 1606–1611. Morgan Kaufmann Publishers Inc., January 2007
13. Gahbiche-Braham, S., Bonneau-Maynard, H., Lavergne, T., Yvon, F.: Joint segmentation and POS tagging for Arabic using a CRF-based classifier. In: LREC, pp. 2107–2113 (2012)
14. Ganguly, D., Roy, D., Mitra, M., Jones, G.J.: Word embedding based generalized language model for information retrieval. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 795–798. ACM (2015)
15. Gupta, P., Barrón-Cedeño, A., Rosso, P.: Cross-language high similarity search using a conceptual thesaurus. In: Catarci, T., Forner, P., Hiemstra, D., Peñas, A., Santucci, G. (eds.) CLEF 2012. LNCS, vol. 7488, pp. 67–75. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33247-0_8
16. Happe, A., Pouliquen, B., Burgun, A., Cuggia, M., Le Beux, P.: Automatic concept extraction from spoken medical reports. Int. J. Med. Inform. **70**, 255–263 (2003)
17. Hattab, E.: Cross-language plagiarism detection method: Arabic vs. English. In: 2015 International Conference on Developments of E-Systems Engineering (DeSE), pp. 141–144. IEEE (2015)
18. Kent, C.K., Salim, N.: Web based cross language plagiarism detection. In: 2010 Second International Conference on Computational Intelligence, Modelling and Simulation (CIMSiM), pp. 199–204. IEEE (2010)
19. Lee, M.C.: A novel sentence similarity measure for semantic-based expert systems. Expert Syst. Appl. **38**, 6392–6399 (2011)

20. Li, Y., McLean, D., Bandar, Z.A., O'shea, J.D., Crockett, K.: Sentence similarity based on semantic nets and corpus statistics. IEEE Trans. Knowl. Data Eng. **18**, 1138–1150 (2006)
21. Liu, C., Chen, C., Han, J., Yu, P.S.: GPLAG: detection of software plagiarism by program dependence graph analysis. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 872–881. ACM (2006)
22. Mcnamee, P., Mayfield, J.: Character n-gram tokenization for European language text retrieval. Inf. Retr. **7**, 73–97 (2004)
23. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Proceeding of the International Conference on Learning Representations Workshop Track, ICLR 2013, pp. 1301–3781 (2013)
24. Mikolov, T., Karafiát, M., Burget, L., Cernockỳ, J., Khudanpur, S.: Recurrent neural network based language model. In: Interspeech, vol. 2, p. 3 (2010)
25. Mikolov, T., Sutskever, I., Chen, K., Corrado, G.S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
26. Mikolov, T., Yih, W.-T., Zweig, G.: Linguistic regularities in continuous space word representations. In: HLT-NAACL, vol. 13, pp. 746–751 (2013)
27. Miller, G.A.: Wordnet: a lexical database for english. Commun. ACM **38**, 39–41 (1995)
28. Mnih, A., Hinton, G.E.: A scalable hierarchical distributed language model. In: Koller, D., Schuurmans, D., Bengio, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems 21, pp. 1081–1088. Curran Associates Inc. (2009)
29. Muhr, M., Kern, R., Zechner, M., Granitzer, M.: External and intrinsic plagiarism detection using a cross-lingual retrieval and segmentation system. In: Notebook Papers of CLEF 2010 LABs and Workshops (2010)
30. Nagoudi, E.M.B., Schwab, D.: Semantic similarity of arabic sentences with word embeddings. In: Proceedings of the Third Arabic Natural Language Processing Workshop, pp. 18–24. Association for Computational Linguistics (2017)
31. Navigli, R., Ponzetto, S.P.: BabelNet: the automatic construction, evaluation and application of a wide-coverage multilingual semantic network. In: Proceedings of Artificial Intelligence, vol. 193, pp. 217–250 (2012)
32. Pataki, M.: A new approach for searching translated plagiarism (2012)
33. Pennington, J., Socher, R., Manning, C.D.: Glove: global vectors for word representation. In: EMNLP, vol. 14, pp. 1532–1543 (2014)
34. Pinto, D., Civera, J., Barrón-Cedeno, A., Juan, A., Rosso, P.: A statistical approach to crosslingual natural language tasks. J. Algorithms **64**, 51–60 (2009)
35. Potthast, M., Barrón-Cedeño, A., Stein, B., Rosso, P.: Cross-language plagiarism detection. Lang. Resour. Eval. **45**, 45–62 (2011)
36. Potthast, M., Stein, B., Anderka, M.: A Wikipedia-based multilingual retrieval model. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) ECIR 2008. LNCS, vol. 4956, pp. 522–530. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78646-7_51
37. Rios, M., Specia, L.: UoW: multi-task learning Gaussian process for semantic textual similarity. In: Proceedings of SemEval, pp. 779–784 (2014)
38. Shao, Y.: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval 2017) (2017)
39. Sultan, M.A., Bethard, S., Sumner, T.: DLS@CU: sentence similarity from word alignment and semantic vector composition. In: Proceedings of the 9th International Workshop on Semantic Evaluation, pp. 148–153 (2015)

40. Tian, J., Zhou, Z., Lan, M., Wu, Y.: Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval 2017) (2017)
41. Turian, J., Ratinov, L., Bengio, Y.: Word representations: a simple and general method for semi-supervised learning. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 384–394. Association for Computational Linguistics (2010)
42. Vinokourov, A., Shawe-Taylor, J., Cristianini, N.: Inferring a semantic representation of text via cross-language correlation analysis. In: NIPS 2002: Advances in Neural Information Processing Systems, pp. 1473–1480 (2003)
43. Wali, W., Gargouri, B., Hamadou, A.B.: Enhancing the sentence similarity measure by semantic and syntactico-semantic knowledge. Vietnam J. Comput. Sci. **4**, 51–60 (2016)
44. Wu, H., Huang, H., Jian, P., Guo, Y., Su, C.: In: Proceedings of the 11th International Workshop on Semantic Evaluation (semeval 2017) (2017)
45. Zahran, M.A., Magooda, A., Mahgoub, A.Y., Raafat, H., Rashwan, M., Atyia, A.: Word representations in vector space and their applications for Arabic. In: Gelbukh, A. (ed.) CICLing 2015. LNCS, vol. 9041, pp. 430–443. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18111-0_32

A POS-Based Preordering Approach for English-to-Arabic Statistical Machine Translation

Mohamed Seghir Hadj Ameur(✉), Ahmed Guessoum(✉), and Farid Meziane(✉)

NLP, Machine Learning and Applications (TALAA) Group
Laboratory for Research in Artificial Intelligence (LRIA),
Department of Computer Science, University of Science and Technology Houari Boumediene (USTHB), Bab-Ezzouar, Algiers, Algeria
{mhadjameur,aguessoum}@usthb.dz, f.meziane@salford.ac.uk

Abstract. In this work, we present a POS-based preordering approach that tackles both long- and short-distance reordering phenomena. Syntactic unlexicalized reordering rules are automatically extracted from a parallel corpus using only word alignment and a source-side language tagging. The reordering rules are used in a deterministic manner; this prevents the decoding speed from being bottlenecked in the reordering procedure. A new approach for both rule filtering and rule application is used to ensure a fast and efficient reordering. The tests performed on the IWSLT2016 English-to-Arabic evaluation benchmark show a noticeable increase in the overall Blue Score for our system over the baseline PSMT system.

Keywords: Machine translation · Arabic NLP · Preordering
Reordering rules · Statistical translation

1 Introduction

When translating between two languages that are noticeably different in terms of their grammatical structures, the task of producing a high-quality translation in a correct word order becomes a serious challenge. Finding a better way to model these grammatical transformations or what is known as reordering phenomena, was and still is a long standing problem which receives a great deal of attention from the machine translation community. The classic Phrase-based Statistical Machine Translation System (PSMT) [1–3] has two means for word reordering: it can either learn the whole bi-phrase as an entry in the phrase table (generally a bi-phrase length does not exceed a certain limit) or via the distortion model which allows limited phrase movements (reorderings) in the output, but with a certain penalty. These means cannot address reorderings that involve a long distance jump or what is known as long-distance reordering. This problem represents a well-known limitation for the standard PSMT system, hence the need to provide a more sophisticated model to solve it.

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 34–49, 2018.
https://doi.org/10.1007/978-3-319-73500-9_3

Syntactic reordering in machine translation is a research area that aims to find more efficient solutions so as to handle both short- and long-distance reordering problems. One common way to perform reordering as a standalone process is known as preordering. Preordering is a preprocessing step that precedes the PSMT phase; its goal is to minimize the syntactic gap between languages and make their grammatical construction as close as possible. Preordering is commonly used to address the long-distance reordering problems, it has the advantage of being easy to use and independent from the used translation system.

Figure 1 gives an example of short- and long-distance word reordering phenomena performed on an aligned English-to-Arabic sentence pair. Both the English and Arabic texts are written from left-to-right to keep the alignment order consistent. In the first example (a) the word "announced" appears at the end of the English sentence, aligned to the Arabic word "أعلن" by means of the 6^{th} alignment link. Preordering will attempt to swap the position of the first and the 6^{th} alignment links, which moves the word "announced" to the beginning of the English sentence to match the Arabic sentence order as shown in the second example (b). Since these two links are separated by a large margin we call this reordering a long-distance word reordering. Short reordering cases such as the one involving the second and the third links are considered as performing short-distance reordering.

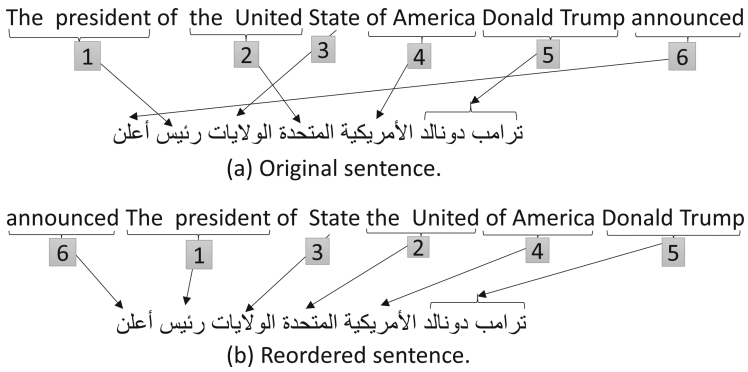


Fig. 1. An example illustrating the process of word reordering performed on an English-to-Arabic word-aligned sentence pair. (The English and Arabic texts are written from left-to-right to keep the alignment order consistent)

The goal of word preordering is to perform a reordering process on the English side of the corpus prior to the translation phase. This is done by finding and applying a set of syntactic rules which helps tweak the grammatical construction of the English-side language making it as close as possible to the Arabic-side language structure, which can also be seen as the task of minimizing the alignment links, as shown in Fig. 1(b).

In this work, we propose a preordering method that can efficiently handle both long- and short-distance word reorderings. The reordering rules are learned automatically from a parallel corpus using word alignment and a basic part-of-speech source language tagging. The test results showed a noticeable improvement over the baseline PSMT system which proves the consistency and adequacy of our proposal. The remainder of this paper is organized as follows: the next section gives an overview of the state-of-the-art preordering methods. The used PSMT baseline is then presented in Sect. 3. Section 4 presents our proposed reordering system and explains the details of each of its components. In Sect. 5, we present and discuss the tests we have done and the results we have obtained. Finally, in the last Section, we conclude our work and highlight some possible future improvements.

2 Related Work

Preordering methods can be classified into two main categories: the deterministic approaches which provide the decoder with only one optimal reordering; and the non-deterministic methods which feed the decoder with multiple candidate sentences in the form of a weighted lattice, and it is then up to the decoder to find the best choice among them.

In terms of deterministic methods, one of the earlier works was done by Xia et al. [4], whose system deals with the task of French-to-English machine translation. They automatically extracted syntactic rules (which they called rewrite patterns) from a bilingual corpus, using syntactic parsers of the source and target languages along with word alignment. They reported a 10% relative improvement in the Blue Score. Habash [5] proposed a preordering method for Arabic-to-English translation. He used word alignment and a source dependency parse tree to automatically extract syntactic reordering rules. The extracted rules were used to reorder the Arabic training and testing data. He investigated various alignment strategies and parsing representations and provided a comparative analysis of the different combinations of the investigated strategies. Genzel [6] defined the reordering task as a dependency parse tree transformation, in which the goal is to find the best children order for each internal node that has more than two children. He proposed a number of metrics for rule quality estimation which allows the filtering and selection of higher quality reordering rules. His proposal was tested on the task of translation from English to various other languages. In a similar fashion, Yang et al. [7] performed the reordering task on a dependency parse tree by reordering the children of each internal node. They handled the position of each node as its rank making the reordering a ranking problem in which the task is to find a certain function f that determines the best rank of each child. Then, the children get sorted according to their ranks. For the task of translation from Chinese to Japanese, Sudoh et al. [8] used a learning-to-rank model based on a pairwise classification method to predict the target Japanese word order. In the same spirit, Jehl et al. [9] proposed a feature-based reordering model for English-to-Japanese and English-to-Korean translation. Their model

predicts whether a pair of sibling nodes on the source-side of the parse tree needs to be swapped. Based on the node swapping probabilities, a global branch-and-bound search is applied to find the best ordering of the children. Fuji et al. [10] proposed a global reordering model that captures language-specific sentence structure directly from non-annotated corpora and use it to boost the performance of conventional syntactic reordering system.

In terms of non-deterministic methods, Zhang et al. [11] presented a preordering strategy for Chinese-to-English translation using chunk-based syntactic rules. They used a source-reordering lattice instead of a single best reordering, and a reordering source language model as an additional feature to score each path in the lattice. Elming [12] presented a preordering approach for English-to-Danish translation. His proposed approach automatically learns probabilistic rules from a parallel corpus. The reordered sentences are fed via a lattice to the SMT decoder. He reported an absolute improvement in the translation quality of 1.1% in Blue Score.

Despite the existing work on word preordering, to the best of our knowledge, no strategy has appeared to give ideal reordering results, hence the continuing efforts to improve them. This work aims to introduce a new, efficient way for both rule identification and application, along with a method for estimating rules usefulness in the reordering process.

3 PSMT Baseline System

Given a source sentence f that we want to translate into a target sentence e. The phrase-based statistical machine translation [1–3] finds the best translation $\hat{e}$ from the space of all possible translations of f.

$$\hat{e} = argmax_e = p(e|f) \tag{1}$$

This can be decomposed using the noisy channel decomposition [13] into a translation model $p(f|e)$ and a language model $p(e)$.

$$\hat{e} = argmax_e = (p(e) * p(f|e)) \tag{2}$$

The translation model ensures the accuracy of the translation between the source and the target languages, and the language model ensures the fluency of the generated target sentences.

A more common generalization is the log-linear model [13] which, instead of splitting the problem into a translation and a language model, it enables the incorporation of arbitrary components (or features), with the assumption that these components are independent from each other,

$$\hat{e} = argmax_e = \left(\frac{exp^{\sum_1^M \alpha_m \varphi_m(f,e)}}{\sum_{e'} exp^{\sum_1^M \alpha_m \varphi_m(f,e')}}\right) \tag{3}$$

where M is the number of components, $\varphi_m(f,e)$ is the m^{th} component and α_m is its corresponding weight in the log-linear model.

The denominator $\sum_{e'} exp^{\sum_1^M \alpha_m \varphi_m(f,e')}$ being constant for all possible translations e', it can be omitted at decoding.

$$\hat{e} = argmax_e = exp^{\sum_1^M \alpha_m \varphi_m(f,e)} \quad (4)$$

Since the log-linear components are supposed to be independent, they can be trained separately. After training each component, an optimization technique such as the Minimum Error Rate Training (MERT) [14] can be used to find the optimal component weights. Our baseline log-linear model includes the following components:

- Phrase translation model
- Language model
- Distance-based reordering model
- Word penalty.

4 Preordering System

Our proposed preordering system automatically learns syntactic reordering rules and uses them to change the grammatical structure of the English source-side sentences making them as close as possible to the target Arabic one.

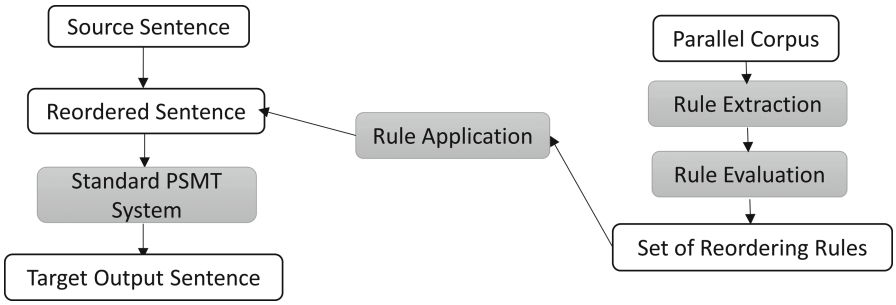


Fig. 2. Architecture of the preordering framework

Figure 2 shows the architecture of our proposed preordering system. There are two main steps: first, a set of reordering rules will be extracted from a parallel corpus, then each rule will be evaluated using a specific rule evaluation mechanism. The second step applies the selected rules to reorder both the training and testing data prior to their exploitation in the PSMT.

4.1 Reordering Rules Definition

In this work, the reordering rules are composed of part-of-speech tags only; as such all the used rules are unlexicalized. Two tagsets are considered: the English

Penn Treebank (PTB) tagset (48 tags) [15], and the English Universal (Univ) tagset (17 tags) [16].

Using high-level tags (more general tag classes) will result in more general rules, while using more specific tags will allow the rules to capture more accurate contextual information albeit with a low generalization ability. The intuition behind using these two different tagsets is to investigate in a practical way the impact of the tags fineness on the reordering performance.

Our reordering rules are composed of three parts: the condition, the reordering, and an optional context. A rule condition may have more than one possible reordering, each reordering having its own specific context.

Table 1. An example of reordering rules

Rule number	Rule condition	Rule action (Reordering)	Rule context
1	DT NNP NNPS	2, 0, 1	(IN, IN)
2	IN DT NNP NNPS	3, 0, 1, 2	(NN, IN)
3	DT NN IN ... VBD	10, 0, 1, ..., 9	(Non, Non)

Table 1 shows an example of reordering rules with PTB part-of-speech tags. The first column presents the condition part of the rule and the second column shows all its corresponding reorderings. The context is presented as a pair (previous tag, next tag) which need to appear before and after the condition part of the rule. For example, for the first rule, the sequence of tags "DT NNP NNPS" need to be present in the sentence. Additionally, the left and right contextual tags "**IN** DT NNP NNPS **IN**" also need to appear before and after the condition. In such case, the reordering "2, 0, 1" can be applied to reorder the tags producing a new order "NNPS DT NNP".

4.2 Reordering Rules Extraction

The reordering rules are extracted using only word alignment and a tagged source text of the parallel corpus. Figure 3 shows the overall process of rules extraction.

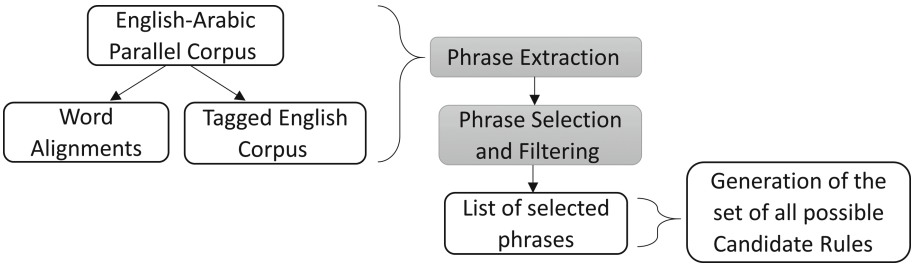


Fig. 3. Rules extraction mechanism

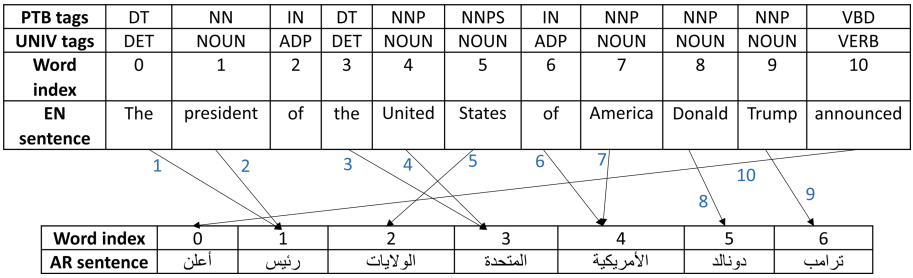


Fig. 4. Word alignment with tagged source-side, in which both the English and Arabic texts are written from left-to-right to keep the alignment order consistent

First, the bi-phrases are extracted using word alignment with a tagged source text, the phrases are then filtered by imposing some restrictions, and finally, the candidate rules are formed from the selected phrases.

Figure 4 shows an example of word alignment with a tagged source text. The Univ and PTB tags are presented for each word in the English source sentence. In the first step, phrase extraction is done using the standard phrase extraction algorithm described by Koehn [13], which uses word alignment to extract bi-phrases from a parallel corpus. From our previous example of Fig. 4, the phrase extraction algorithm extracts a total of 25 bi-phrases, some of which are shown in Table 2.

Table 2. Some extracted bi-phrases from the example given in Fig. 4 using the standard phrase extraction algorithm described in [13]

English Phrase	Arabic Phrase
Donald Trump	دونالد ترامب
the United States	الولايات المتحدة
announced	أعلن

Having a set of English-Arabic bi-phrases, we select the pair of bi-phrases that contain a crossing. For instance, in Fig. 4 the phrases denoted by the links 4 and 5 cross each other (by abuse of language, since the links cross each other); thus swapping them will minimize the number of crossing links; which makes the English sentence structure more similar to the Arabic one. We will be using this kind of crossings of bi-phrases in-order to form our syntactic rules.

Given two bi-phrases $p_1 = (s_1, t_1)$ and $p_2 = (s_2, t_2)$ where s_i and t_i are phrases from the source and target sentences, respectively, the two bi-phrases p_1 and p_2 are considered valid to form a syntactic rule if they satisfy the following conditions:

1. If s_1 precedes s_2 in the source sentence, then t_2 must precede t_1 in the target sentence. In other words, the two bi-phrases must cross each other.
2. The two bi-phrases must be consecutive in both the source and the target sentences. In other words, s_2 must follow s_1 and t_2 must follow t_1.

For example in Fig. 4, the two bi-phrases denoted by the links 5 and 4 respect these two conditions. The two bi-phrases 5 and 3 do not respect the second condition (they are not consecutive in the English text).

The set of selected phrases are then used to generate unlexicalized syntactic rules; the left and right tags that precede and follow the two phrases are used as context. Table 1 shows some valid rules that can be extracted from Fig. 4.

4.3 Reordering Rules Evaluation

The extracted rules are not always useful for reordering. In fact, most of them are very specific, which makes their coverage quite limited. Another issue resides in the errors introduced by the automatic word alignment which increases the rate of incorrect rules.

To tackle these problems, a number of metrics that estimate rules quality have been proposed. The metric that is most used is the Crossing Score (CS) [6] which determines the quality of a rule based on the decrease in crossing alignment links after its application.

In practice, the quality of a rule is tested on the whole training corpus by applying the rule to each of its sentences and evaluating the change in the number of crossing alignments. This should give a solid estimation of the rule quality.

Applying this kind of metric directly will be computationally expensive since each rule is generally evaluated separately. Another issue is to perform the reordering task when given a set of rules. This involves finding all the applicable rules and determining the best order for their application. To this end, we build an index that accelerates both rules lookup and rules application.

Index Construction. We build an index, which is a compact Trie [17]. To reduce rule lookup time, this index will be used for the tasks of rules evaluation and application.

Formally, given a set of rules R, with their conditions driven from a set of tags G, such that $|R| = n$ and $|G| = m$. Each rule r in R, is a tuple (c, a, x), where c is the condition, a is the action and x is the rule context.

We construct a compact Trie T on R which has the following characteristics:

- A root node and n leaves (since each rule condition c ends at a leaf node which contains its corresponding action a and context x).
- For each internal node, its descendants have the same prefix (the same condition).
- Two branches leaving the same node can't start with the same prefix.

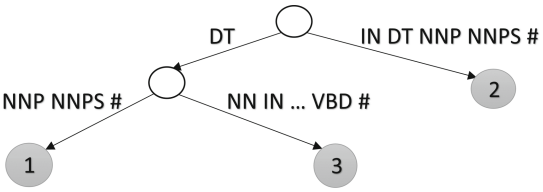


Fig. 5. Syntactic rules indexing via a compact Trie

Figure 5 shows a compact Trie constructed over the tag-sequences of the previous set of rules from Table 1. The leaf nodes are numbered according to their corresponding rules and the labels are printed on the edges. A special end tag # is added to ensure that each sequence terminates at the level of a leaf node. The reordering and context for each rule are kept in the leaf node that corresponds to it.

Efficient Search for Applicable Rules. The task of finding all the applicable rules for a given sentence is very challenging given the number of rules and the variations in their corresponding part-of-speech tags. Algorithm 1 presents an easy and efficient way to identify all the applicable rules for a given sentence using a compact Trie representation for all the reordering rules. In-order to find all the applicable rules for a sentence S, Algorithm 1 finds all the applicable rules for each suffix (each position) in the tagged source sentence S_t; this is done by traversing the Trie starting from the root node and following the path led by each suffix in s_t. The rules found for each suffix are accumulated in *candidate_rules* and returned when the algorithm terminates. All the applicable rules for S can be found in $O(k^2)$ time, where k is the length of S.

Algorithm 1. Algorithm *FindAll* that finds all the applicable rules for a given sentence

Input : T: a Trie constructed over a set of rules R.
$S_t = t_1, t_2, t_k$: the part-of-speech tags of the English sentence S where k is the length of S.
Output: *candidate_rules*: a list that contains all the applicable rules for S.

```
1 Function FindAll(R, T, S_t):
2     foreach suffix s_t in S_t starting at postion i do
3         rules_i = finds all applicable rules for s_t in T;
4         foreach rule r in rules_i do
5             candidate_rules.add ((r, i))
6         end
7     end
8 return candidate_rules;
```

Having the set of applicable rules for a given sentence, we need to determine the best one among them. This is done by sorting the obtained candidate rules according to their condition part to ensure that the rules concerning long-distance reorderings are applied first.

Rule Quality Evaluation. As mentioned in Sect. 4.3, since the majority of rules are not useful for reordering purposes, a good method for rule quality estimation is needed. Algorithm 2 scores the rules which can be applied to a given sentence using the CS metric.

Algorithm 2. Algorithm *UpdateSent* that updates the evaluation score for each syntactic rule

Input : R: a set of rules.
T: a Trie constructed over R.
$S_t = t_1, t_2, t_k$: S part-of-speech tags for the sentence S.
S_a: alignment points for the sentence S and its target translation.
close: a close list.
Output: Updates the scores for each applicable rule in T for the sentence S.

```
1  Function UpdateSent(R, T, S_t, S_a, close):
2      original_CS = findCS(S_a)
3      while i < max_iterations do
4          candidate_rules = FindAll(R, T, S_t) that are not present in close;
5          r_best = find the best rule in candidate_rules;
6          insert r_best in close;
7          S'_a = reorder S_a using r_best;
8          new_CS = findCS(S'_a);
9          r_best.usage+=1;
10         if new_CS < original_CS then
11             r_best.positive+=1;
12         end
13         else if new_CS > original_CS then
14             r_best.negative+=1;
15         end
16         else
17             r_best.neutral+=1;
18         end
19         i = i + 1;
20     end
```

Algorithm 2 starts by identifying the best applicable rule for a given sentence S as described in Sect. 4.3. A close list is then used to prevent the rules from being reused in the same position. The best rule is then applied to reorder the word-aligned sentence, and the number of crossing alignments is then estimated using the CS metric. The score of the applied rule is then updated based on the

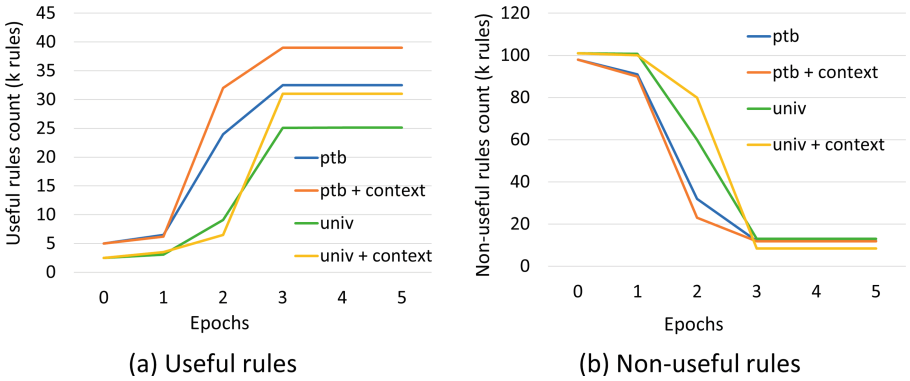


Fig. 6. Rules count variation when applying the process of rules filtering

CS difference[1]. This process is repeated for several iterations as indicated by the *max_iterations* variable.

After scoring all the rules, we estimate the usefulness of a given rule r by taking the ratio of the number of time the rule gave a positive impact on the reordering task and the total number of its application:

$$usefulness(r) = \frac{positive(r)}{usage(r)} \tag{5}$$

In case the rule usefulness surpasses a certain threshold, it will be considered useful and selected for reordering. Applying this equation on the whole corpus will select a subset of useful reordering rules (since not all the rules can be applied in all possible contexts). This process of rules usefulness estimation is repeated for several iterations until the subset of the useful rules stabilizes, which indicates that a convergence point has been reached.

Figure 6 shows the variation of the number of selected rules when estimating the rules' usefulness.

The number of useful rules (Fig. 6(a)) increases with the number of epochs and at the same time, the count of non-useful rules (Fig. 6(b)) decreases until a convergence is achieved.

5 Experiments

To test our approach, we have used the English-Arabic parallel corpus provided by the IWSLT2016[2] evaluation campaign which offers a complete testing framework which includes: training, development, and evaluation data. Our results have been obtained on the IWSLT 2010, 2011, 2012, 2013 and 2014 test sets.

[1] *FindCS* is a simple method that finds the number of crossing alignments (CS) for a given aligned sentence.

[2] http://workshop2016.iwslt.org/59.php.

Tagging is done using the Stanford English Log-linear Part-Of-Speech Tagger [18]. The Univ part-of-speech tags are obtained by converting the PTB tagset using a simple tag mapping method[3].

5.1 Preprocessing

For the Arabic language, our preprocessing includes: diacritic sign removal, Arabic character normalization and word segmentation by means of the AMIRA toolkit [19] using the default tokenization scheme in which conjunctions, prepositions, determinants, suffixes and future markers are all individually separated. For the English side, only word tokenization is performed using the Python NLTK toolkit[4]. We have also added a number <*nbr*> and a link classes <*url*> to all numbers and links found in the parallel corpus respectively. Sentence length has been limited to 40 words; "bad" sentence pairs, i.e. whose length difference exceeds a certain threshold were also removed.

Table 3 shows some statistics about the resulting data from the preprocessing step.

Table 3. Statistics about the training corpus

	English	Arabic
Sentences	110 549	110 549
Words	1692394	1910968
Unique words	26574	37539

5.2 Evaluation of Translation Quality

We have investigated the use of two tagsets and the presence/absence of part-of-speech contexts. This leads to four systems:

1. Reordering with Univ tagset without context.
2. Reordering with PTB tagset without context.
3. Reordering with Univ tagset with context.
4. Reordering with PTB tagset with context.

All our systems have been tested using the Moses PSMT framework [20]. We have used a 6-gram language model instead of the default tri-gram model to ensure a better language modeling for the segmented Arabic language. The rest of the parameters are kept unchanged. Our test results have been reported using the Blue Score Metric [21].

[3] The conversion table can be found in the following link http://universaldependencies.org/tagset-conversion/en-penn-uposf.html.

[4] http://www.nltk.org/.

Table 4. Blue score results for the PSMT baseline and the MSE-bidirectional reordering model

Test set	PSMT-Baseline	PSMT-MSE-Bi
IWSLT2010	17.24	17.33 (+0.09)
IWSLT2011	17.28	17.54 (+0.26)
IWSLT2012	19.30	19.48 (+0.18)
IWSLT2013	18.67	18.64 (−0.03)
IWSLT2014	16.16	16.82 (+0.66)

Table 4 shows the Blue Score results obtained by the Moses baseline with and without its default MSE-bidirectional reordering model [22]. The values in parentheses indicate the gain in Blue score with respect to the PSMT baseline system. A slight increase in Blue Score is obtained when the default Moses reordering model was turned on.

Table 5. Blue scores using the PTB and the Univ part-of-speech tags without including the context

Test set	Base-UNIV	Base-PTB
IWSLT2010	17.03 (−0.21)	17.52 (+0.28)
IWSLT2011	17.62 (+0.34)	18.18 (+0.90)
IWSLT2012	19.80 (+0.50)	19.95 (+0.65)
IWSLT2013	18.73 (+0.06)	19.11 (+0.44)
IWSLT2014	17.22 (+1.06)	17.51 (+1.35)

Table 5 shows the reordering results obtained when using the PTB tags and the Univ tags without including the context. The obtained results when using the PTB was slightly better than the one obtained with the Univ tags. The maximum gain in Blue Score was 1.35 point compared to the Baseline PSMT system.

Table 6 shows the reordering results obtained using the PTB tags and the Univ tags when the context is included. The obtained results for the two tagsets were very similar with a maximum increase of about 1.5 in the Blue Score over the PSMT baseline. These results prove the importance of using the context to enhance the accuracy of the syntactic rules. Indeed, the more specific the rules, the better. Another thing to note is the effect of the POS-tag fineness: we can see that the use of PTB tags yields better results than with Univ tags, especially when no context is used. This suggests that more tag fineness will lead to a more accurate reordering.

Table 6. Blue Scores using the PTB tags and the Univ tags when including the context

Test set	CONTEXT-UNIV	CONTEXT-PTB
IWSLT2010	17.72 (+0.48)	17.71 (+0.47)
IWSLT2011	18.31 (+1.03)	18.34 (+1.06)
IWSLT2012	19.90 (+0.6)	20.09 (+0.79)
IWSLT2013	19.13 (+0.46)	19.24 (+0.57)
IWSLT2014	17.61 (+1.45)	17.58 (+1.42)

5.3 Evaluation of Alignment Ambiguity

We have also used the Normalize Crossing Links Score (NCS) [6] to measure the quality of the different investigated reordering systems. The NCS metric formula is the following:

$$NCS = \frac{C}{S} \tag{6}$$

where C is the number of crossing links in the aligned corpus and S is the number of words in the source text of the corpus.

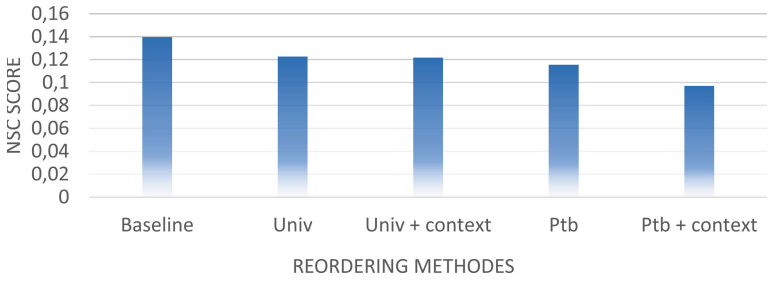


Fig. 7. The NCS scores for the different reordering methods

For this formula the smallest the NCS score, the better. An ideal score will be zero, which means that the corpus is completely monotonic[5]. The NCS scores are shown in Fig. 7 for the different reordering methods. We recall that the smallest the score, the better. The results indicate that using the PTB tagset with the contextual information produce less ambiguous alignments, hence better translation results.

[5] We mean by a monotonic corpus, a corpus in which the alignment does not contain any crossing links.

6 Conclusion

We have introduced a general method for word preordering in which reordering rules are extracted from a parallel corpus using only word alignments and basic part-of-speech tagging. Rule quality is estimated using the CS metric, which allows the selection of only the best applicable rules. Our proposal has been evaluated in terms of translation quality using the Blue Score, and the change in alignment ambiguity has been investigated using the NCS metric. We have found out that using the PTB tags yield a more noticeable improvement over the baseline PSMT system; this suggests that the higher the tag fineness the better the effect of the part-of-speech preordering methods.

As a future work, we plan to explore a similar approach using tree structures (dependency and constituency trees). We also plan to examine the coupling of both preordering and post-ordering strategies in the same framework and check whether that yields to further improvements in the overall translation performance.

References

1. Brown, P.F., Cocke, J., Della-Pietra, S.A., Della-Pietra, V.J., Jelinek, F., Lafferty, J.D., Mercer, R.L., Rossin, P.: A statistical approach to machine translation. Computat. Linguist. **16**(2), 76–85 (1990)
2. Zens, R., Och, F.J., Ney, H.: Phrase-based statistical machine translation. In: Jarke, M., Lakemeyer, G., Koehler, J. (eds.) KI 2002. LNCS (LNAI), vol. 2479, pp. 18–32. Springer, Heidelberg (2002). https://doi.org/10.1007/3-540-45751-8_2
3. Och, F.J., Ney, H.: Discriminative training and maximum entropy models for statistical machine translation. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics. Association for Computational Linguistics, pp. 295–302 (2002)
4. Xia, F., McCord, M.: Improving a statistical MT system with automatically learned rewrite patterns. In: Proceedings of the 20th International Conference on Computational Linguistics. Association for Computational Linguistics, p. 508 (2004)
5. Habash, N.: Syntactic preprocessing for statistical machine translation. In: Proceedings of the 11th MT Summit, p. 10 (2007)
6. Genzel, D.: Automatically learning source-side reordering rules for large scale machine translation. In: Proceedings of the 23rd International Conference on Computational Linguistics, pp. 376–384. Association for Computational Linguistics (2010)
7. Yang, N., Li, M., Zhang, D., Yu, N.: A ranking-based approach to word reordering for statistical machine translation. In: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers, vol. 1, pp. 912–920. Association for Computational Linguistics (2012)
8. Sudoh, K., Nagata, M.: Chinese-to-Japanese patent machine translation based on syntactic pre-ordering for WAT 2016. In: Proceedings of the 3rd Workshop on Asian Translation (WAT2016), pp. 211–215 (2016)
9. Jehl, L., Gispert, A., Hopkins, M., Byrne, W.: Source-side preordering for translation using logistic regression and depth-first branch-and-bound search (2014)

10. Fuji, M., Utiyama, M., Sumita, E., Matsumoto, Y.: Global pre-ordering for improving sublanguage translation. In: WAT 2016, p. 84 (2016)
11. Zhang, Y., Zens, R., Ney, H.: Chunk-level reordering of source language sentences with automatically learned rules for statistical machine translation. In: Proceedings of the NAACL-HLT 2007/AMTA Workshop on Syntax and Structure in Statistical Translation, pp. 1–8. Association for Computational Linguistics (2007)
12. Elming, J.: Syntactic reordering integrated with phrase-based SMT. In: Proceedings of the Second Workshop on Syntax and Structure in Statistical Translation, pp. 46–54. Association for Computational Linguistics (2008)
13. Koehn, P.: Statistical Machine Translation, 1st edn. Cambridge University Press, New York (2010)
14. Och, F.J.: Minimum error rate training in statistical machine translation. In: Proceedings of the 41st Annual Meeting on Association for Computational Linguistics, vol. 1, pp. 160–167. Association for Computational Linguistics (2003)
15. Marcus, M.P., Marcinkiewicz, M.A., Santorini, B.: Building a large annotated corpus of English: the penn treebank. Comput. Linguist. **19**(2), 313–330 (1993)
16. Petrov, S., Das, D., McDonald, R.: A universal part-of-speech tagset. arXiv preprint. arXiv:1104.2086 (2011)
17. De La Briandais, R.: File searching using variable length keys. In: Papers presented at the March 3–5, 1959, Western Joint Computer Conference, pp. 295–298. ACM (1959)
18. Toutanova, K., Klein, D., Manning, C.D., Singer, Y.: Feature-rich part-of-speech tagging with a cyclic dependency network. In: Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology, vol. 1, pp. 173–180. Association for Computational Linguistics (2003)
19. Diab, M.: Second generation AMIRA tools for Arabic processing: fast and robust tokenization, POS tagging, and base phrase chunking. In: 2nd International Conference on Arabic Language Resources and Tools, vol. 110 (2009)
20. Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., et al.: Moses: open source toolkit for statistical machine translation. In: Proceedings of the 45th Annual Meeting of the ACL on Interactive Poster and Demonstration Sessions, pp. 177–180. Association for Computational Linguistics (2007)
21. Papineni, K., Roukos, S., Ward, T., Zhu, W.J.: BLEU: a method for automatic evaluation of machine translation. In: Proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pp. 311–318. Association for Computational Linguistics (2002)
22. Koehn, P., Axelrod, A., Birch, A., Callison-Burch, C., Osborne, M., Talbot, D., White, M.: Edinburgh system description for the 2005 IWSLT speech translation evaluation. In: IWSLT, pp. 68–75 (2005)

Speech Recognition and Synthesis

Towards a High-Quality Lemma-Based Text to Speech System for the Arabic Language

Oumaima Zine(✉), Abdelouafi Meziane, and Mohamed Boudchiche

Department of Mathematics and Computer Science, Mohammed First University, Oujda, Morocco
zine.oumaima@gmail.com, abdelouafi_meziane@yahoo.fr, moha.boudchiche@gmail.com

Abstract. Recent numbers put the Arabic language at around 250 million native speakers, making it the fifth spoken language regarding the number of speakers. Therefore, it has gained the interest of researchers in speech technologies in particular speech recognition and speech synthesis. Indeed, many researchers are still investigating in Arabic Text To Speech to deliver an intelligible and close to natural Text To Speech systems. Nevertheless, the most of the available free and semi-free Arabic Text To Speech systems are still away from the natural sounding as human voice does, and the generation of smooth voice is still involved. The primary intention of this work is to increase the quality of the produced speech resulting from the sub-segment based approach proposed in our previous work. To this end, a lemma-based approach for concatenative TTS synthesis is adopted and presented in this paper. In this context, a study of Arabic lemmas frequency was conducted to identify the highly frequent lemmas that often occur in written and spoken Classical and Modern Standard Arabic (MSA). This study reports an analysis of roughly 65 million words fully vocalized obtained from Tashkila corpus, Nemlar, and Al Jazeera. These latter cover modern and classical Arabic languages. As a result, an Arabic lemmatized frequency list was generated. The top 1,000 frequent lemmas were found to provide approximately 79% coverage of the Arabic words. Thus, the former were used as the basic acoustic units of our Text to Speech System. Finally, we demonstrate that this approach affords an improvement in the intelligibility and naturalness of a Text To Speech system with an overall rate 4.5 out of 5.

Keywords: Text to speech · Arabic language · Lemma frequency · Speech corpus Speech synthesis · Unit selection · Concatenative synthesis · Sub-segments

1 Introduction

Speech is obviously the main key of communication and interaction between human beings. Therefore, early attempts at speech technologies and especially speech synthesis were made and achieved successfully during the 18th century. Since then, many computer operating systems have included speech synthesizers and were first developed to aid the visually impaired by offering a computer-generated voice that would read the text to the user with good intelligibility.

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 53–66, 2018.
https://doi.org/10.1007/978-3-319-73500-9_4

Recently, the need for text-to-speech synthesis has increased significantly, and it is becoming inevitable. The naturalness of the produced speech is, as well, highly required. To this end, recent advances in text-to-speech (TTS) synthesis research led to more intelligible and natural sounding synthetic speech than a decade ago. However, Arabic text-to-speech research is still in its early stages, and the sound quality remains a major problem. This is due to the challenges of the Arabic language concerning structure and co-articulation [1].

The two primary text-to-speech technologies trends are divided into concatenative synthesis and statistical approaches [2]. In this work, our focus will be on the concatenative approach.

In this paper, a unit selection approach based on the Arabic lemmas is presented and discussed, with the ultimate goal of improving the quality of our text-to-speech system when using the sub-segments based approach introduced in the work published earlier [3]. We are expanding our acoustic corpus to include units of different size, shorter units as sub-segments and longer ones as lemmas.

However, in the case of lemmas, the full coverage is impractical. As the unit size increases, full coverage becomes harder to achieve [4]. It, therefore, seems rational to prioritize using lemmas that are frequently occurring in the Arabic language. In this context, the process and the results of an Arabic lemma frequency study are described in this paper. Finally, we introduce the improvements achieved by using longer units such as lemmas.

This paper is organized as follows. Section 2 presents an overview of the existing Text-To-Speech approaches. In Sect. 3, we discuss our basic unit choice in addition to the process of combining sub-segments and lemma-sized units. Section 4 describes the main steps for generating the lemma frequency count. Section 5 presents the method of construction of our lemma database. Section 6 describes our experiments and shows the results. Finally, Sect. 7 concludes the paper.

2 Overview of Text to Speech

Several approaches are used to synthesis speech from text. A full review of this is described in [5]. However, of particular concern to us is the concatenative approach. There are basically two ways of achieving the concatenative synthesis; on the one hand, a non-consuming approach that is based on fixed inventory where all the segment are of the same length (e.g. diphones). On the other hand, the flexible approach which is the unit selection, where variable length speech samples are stored in the form of sentences, intonational phrases, words, syllables, diphones or phonemes. Afterward, the system has to make the decision to the best match [6]. This latter approach is considered state of the art for most of the commercial TTS systems [7] and applied to many languages (English, Chinese, French, Indian, etc.).

An approach for unit selection was proposed in [8] to improve the Google's unit selection synthesizer. The approach was based on a combination of diphones and phrase-based units to increase the quality of the limited domain applications. A pre-selection algorithm based on an HMM module was used to guide the selection of diphone-sized

candidate units. Additionally, a novel approach was proposed for a runtime unit selection that aims to reduce the latency and achieve rapid selection of a subset of the units that have the highest probability of being adequate for the required context.

Similarly, the earlier synthesizer [4] prioritized the unit selection procedure, by integrating a unit selection TTS in computational environments with limited resources, such as mobile devices, without affecting the speech quality. Moreover, efficient techniques were presented to cope with the issues arising in embedded environments such as the acoustic inventory compression and runtime load minimization.

Regarding the Arabic language, works on unit selection speech synthesis are still very rare. In [9] an Arabic TTS system based on unit selection was developed mainly for North-African users. The proposed system was based on syllables as the basic acoustic units. Furthermore, an algorithm was developed to minimize the selection cost, taking into account a set of phonological, linguistic and contextual features.

On the other hand, many researchers have been carried out to synthesis speech using a fixed inventory of acoustic segments whatever their level (i.e. phonemes, diphones, syllables, or words). As an example, [10] presented a modified MARY TTS for Arabic language using diphone-based concatenative approach. Similarly, Alsharif [11] developed a Holy Quran-based Arabic TTS (HQB-ATTS) based on concatenating allophones/syllables. The developed approach was based on some Tajweed rules. A prosodic syllabification was proposed. An inventory of 2,700 syllables was created and used to synthesize the speech.

Likewise, El Shafei et al. [12] introduced a high-quality Arabic text to speech system based on the concatenation of diphone/sub-syllables to construct the spoken utterances. The chosen speech units cover the classical Arabic where the co-articulation is minimal. They also proposed an extension of the set of the acoustic units to incorporate the common co-articulation effects of the Modern Standard Arabic.

Another alternative was to use longer units such as words and sentences. Campbell [13] introduced conversational text to speech synthesizer based on concatenating frequent phrases and words. Several prosodic variants of identical words are also provided to achieve natural intonation. Moreover, single phone-sized sounds were also used to ensure that any possible sequence of sounds can be generated.

A novel approach was introduced in [3]. The former describes our concatenative method of generating speech from Arabic texts. We proposed a set of sub-segments of variable length, where the consonant is considered as the nucleus of the acoustic unit. This latter consists of Half vowel – Consonant – Half vowel adapted to the different positions in the word (initial, medial and final). In this work, we proved that placing the boundary in the consonant nucleus may affect the smoothness of the produced speech due to the aperiodicity of the consonant signal. Instead, the vowel sound was found to be more suitable for appropriate concatenation due to the periodicity of the vowel and the long steady-state portion of this latter. Additionally, we described a process of optimization based on some phonological rules of the Arabic language. As a result, 72% of units, which do not occur in the Arabic language, were excluded after the optimization process.

In the other hand, this paper aims to enhance the quality and the naturalness of the produced speech when using the former approach, by performing a unit selection

synthesis in a large corpus containing larger units such as lemmas and shorter ones as sub-segments.

Apart from this, the access to fully available text to speech systems is essential. However only a few of them are released and available for noncommercial usage [14, 15]. Table 1 shows an overview of the existing and noncommercial text-to-speech synthesizers and their main features.

Table 1. Overview of some popular TTS systems

System	Supported languages	Synthesis approach
Google TTS	35 languages: English, French, Spanish, German, polish, Turkish…	HMM-driven unit selection
ACAPELA	34 languages: Arabic, French, English…	Unit selection
MBROLA, le Mons Belgian	9 languages: English, French, Spanish…	Diphone-based
WaveNet (Google DeepMind)	English, Chinese	Deep neural network (DNN)
ArabTalk	Arabic	Artificial neural networks (ANN), HMM based synthesis
Sakhr TTS	Arabic, English	Unit selection, diphone-based
Mary TTS 5.2	10 languages: English, French, German, Turkish…	Unit selection, HMM based synthesis

3 Unit Choice

Choosing the correct unit length is the most important aspects in concatenation synthesis. As stated in the overview, current synthesis solutions are mostly based on diphones, syllables, sub-syllables or even triphones. Diphones are defined to extend the nucleus of the first phone to the nucleus of the subsequent one. The main advantage of using diphones is the relatively limited size of their inventory; therefore, they are to some extent affordable for low resources TTS [16]. However, an audible spectral discontinuity may result from using such small units. The use of longer units such as triphones may be suitable for producing natural speech. In contrast, the full coverage of this acoustic unit can be tough assignment [4].

Nonetheless, we cannot cope with the problem of spectral discontinuity in concatenation synthesis, unless longer speech units are adopted, which otherwise would be difficult to handle since it would imply later signal corrections at the concatenation points, and some smoothing techniques that may produce unnatural sounding. So using larger units, and dealing with the coverage problem will be the ideal solution.

Therefore, we are currently investigating the synthesis by lemmas, where the lemma-sized chunks are treated as favored candidate units in addition to the sub-segments proposed in our previous work [3].

In the Arabic language, lemma refers to the masculine singular form of the verb in the past tense (the lemma of the verb "فَأَخَذْتُهُمْ" /fa>axa*atohumo/ is "أَخَذَ" />axa*a/), and

masculine singular for nouns (the lemma of the noun "بِمَدَارِسِهِمْ" /bimadaArisihimo/ is "مَدْرَسَة" /madorasap/), stripped off of its prefixes and suffixes. In the case of particles, their lemma is the particle without clitics (the lemma of the particle "فَفِيهِمْ" /fafiyhimo/ is "فِي" /fiy/).

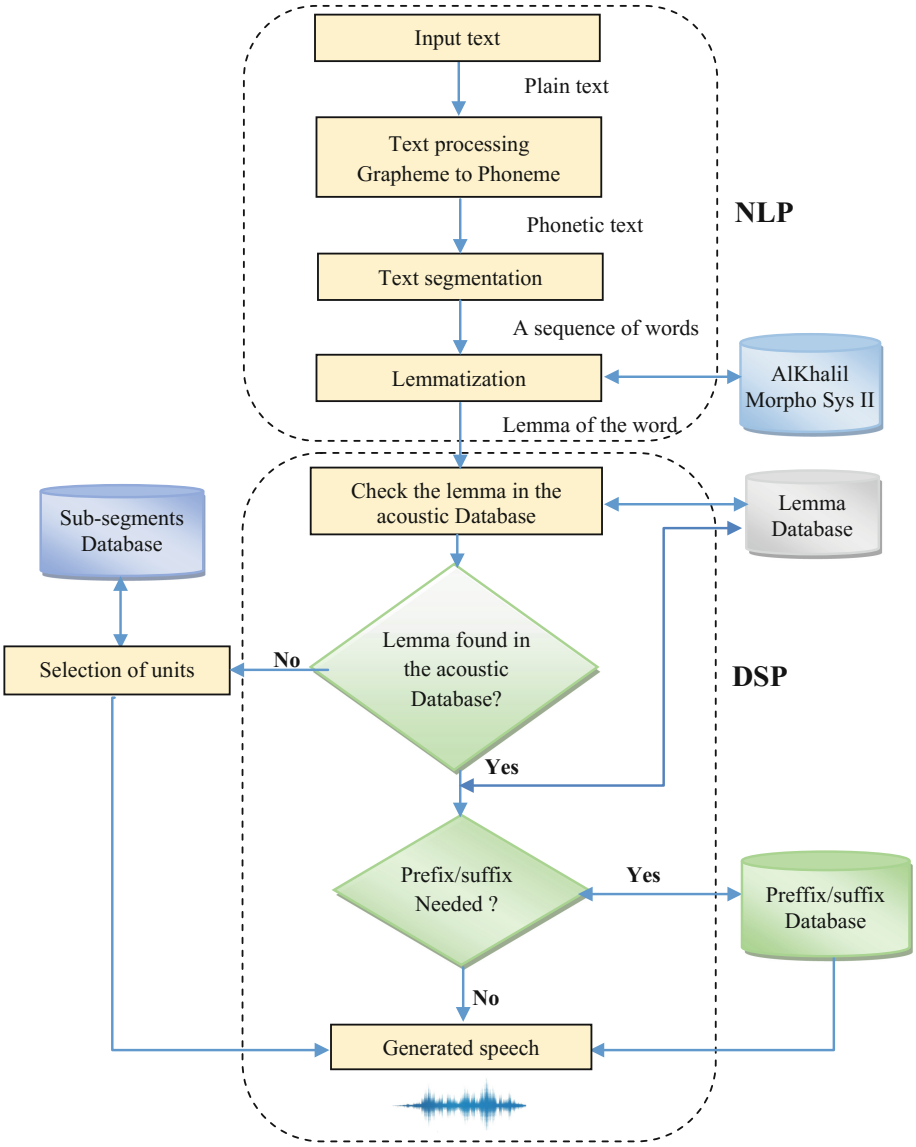


Fig. 1. Overall proposed Arabic TTS system

Our synthesis approach is based on a large inventory of non-uniform units favoring the longer ones which are lemmas and backing off to relatively shorter ones when

needed. The input text is split into words. The lemmatizer extracts the appropriate lemma of each word. Afterward, the system selects the adequate lemma from the speech database. Prefixes and suffixes are then added if needed. Otherwise, if the lemma of the word was not found in the acoustic database, sub-segments based selection module will determine the best candidate units from the sub-segments database and tries to generate this word. Figure 1 describes the overall process of the proposed approach.

The next section tackles the problem of coverage of the Arabic lemmas by identifying the highly frequent ones that often occur in classical and MSA.

4 Lemma Frequency Materials and Methods

As the unit size increases, full coverage becomes harder to achieve especially being limited to only a few hours of speech (around 4 h and 20 min of recording). Therefore, we concluded that the way to use longer units as lemma is to target the most frequently existing lemmas in the Arabic language.

However, the most of the currently available word frequency dictionaries are seriously outdated as stated in [17] and no study was done regarding the frequent lemmas in the Arabic language.

To that end, we describe in this section the process of obtaining the lemma frequency list.

4.1 Lexical Resources

Currently, only a few available Arabic resources are vocalized. For this reason, finding vocalized corpora was considered as a difficult task to accomplish [18].

Our lemma list frequency was compiled from a collection of Arabic vocalized text corpora totaling 64,829,945 words, which cover modern and classical Arabic language.

Below is the list of the Arabic text corpora used in our study:

- **Aljazeera** learning Arabic service is a new service that aims to learn Arabic as a foreign language. The site provides vocalized Arabic texts, exercises, short stories and courses about the Arabic language. About 24,000 unique words from the classical Arabic are contained from Al-Jazeera website [19].
- **Shamela library**[1] is an Islamic e-library [20] that includes hundreds of books in classical Arabic covering several domains, including Hadith, Fiqh, history, etc.
- **Nemlar** is a written fully vocalized corpus [21] produced by RDI, Egypt within the NEMLAR[2] project, it consists of about 500,000 words of Arabic language from 13 different categories including political news, Islamic texts, phrases of common words, Scientific press, dictionary entries explanation, etc. the corpus is provided with the following tags: stem, clitic, POS and the Arabic pattern corresponding to each word of the corpus. Therefore, we proceeded to its lemmatization and then the tags were added to the Nemlar corpus.

[1] http://shamela.ws/.
[2] http://www.nemlar.org.

- **Quranic corpus "Al-Mus'haf"** [22] is a corpus covering Quranic Arabic texts and providing several morphological information such as stem, Stem's pattern, Lemma, Lemma's pattern and the root. 17,455 Quranic words are covered. Table 2 summarizes the used corpora with the number of the provided words.

Table 2. Number of words in the used Arabic corpora

Corpora	Number of words
Shamela library	64,272,589
Nemlar	455,333
Al-Mus'haf	78,247
Al-jazeera	23,776
Total words	64,829,945

4.2 Lemmatization

As mentioned previously, the available tags in the Nemlar corpus are the stem, POS and the Arabic pattern (Al wazn). However, to be able to use this corpus in our study, we proceeded to its enriching with the lemma tag. The following steps were performed for this purpose.

The words obtained from the text corpus were processed using AlKhalil Morpho Sys 2 [23] in order to designate one or more possible lemma tags to each word. Afterward, a process of disambiguation that consists of identifying a single correct lemma, from the outputs of the previous step, for each word was performed manually by a specialist linguist. Finally, we assigned to each word the tag of its unique lemma, and then the tags were included in the Nemlar corpus [24]. A further step of verification of the Nemlar corpus was performed manually by a linguist.

This process of lemmatization was applied as well to the Shamela and Al-Jazeera corpus. Consequently, around 49,808 lemmas resulted from this process.

4.3 Frequency Lemma List Processing

The resulting lemmas from the lemmatization process were assessed in order to generate their frequency count and to measure their dispersion over the corpus. Therefore, we have taken into account not only the words with the form of lemma but also the words composed by the lemma and additional prefixes and suffixes.

The only condition is that the word retains the form of the lemma, without considering its last vowel, which is Sokun for the nominal lemmas and Fatha for verbal ones. For example the word "فَأَخَذُوهم" /fa>axa*uwhm/ is considered as an occurrence of the lemma "أَخَذَ" />axa*a/, similarly the particle "فَفِيهِمْ" /fafiyhimo/ is counted as an occurrence of the lemma "فِي" /fiy/. In contrast, the word "كَأَخْذِهِمْ" /ka>axo*ihimo/ is not considered as an occurrence of the lemma "أَخَذَ" />axa*a/. Table 3 presents the coverage

by the most frequent 10,000 lemmas in Arabic. Table 4 lists some of the top 1,000 frequent lemmas with their occurrences.

Table 3. Lemma coverage for different frequency bands in Arabic

Number of lemmas	Corpora coverage (%)
1,000	78.86
2,000	86.55
3,000	89.99
4,000	91.94
5,000	93.16
6,000	93.97
7,000	94.55
8,000	94.97
9,000	95.29
10,000	95.54

Table 4. Sample of results obtained from the top frequent 1,000 and their occurrences

Lemma	Frequency (%)	Word list
فِي	3.63	وفِيهَا - لَفِيهَا - وَفِيكَ- فِيكُمَا-أَوَفِيكُمْ...
مِنْ	2.30	وَمِنْك - وَمِنْ – أَفَمِنْ – مِنْها – وَلَمِنْ...
قَالَ	2.12	وقَالَ - فَقَالَتِ – فقالوا...
كَانَ	1.47	أَكَانَتَا – أَكَانَ – فكَانَتِ - وَكَانَ...
الله	1.23	وباللهِ – بالله – اللهُمَّ – فالله...

It can be seen from Table 3 that the most frequent 1,000 lemmas in Arabic can provide 78.86% coverage of a standard written text. In other words, storing the most frequent 1,000 lemmas in our speech database means covering about 87.86% of the words from the input text, and hence better naturalness in the generated speech will be achieved. Apparently, high coverage of about 95.54% of the Arabic texts can be reached with the top 10,000 lemmas; however, the problem of coverage of this number of lemmas in our speech database will be increased.

Therefore, we concluded that the most frequent 1,000 are sufficient to enhance the quality of our lemma-based TTS system. For greater clarity of the results, the top ten high-frequency lemmas are represented in Fig. 2.

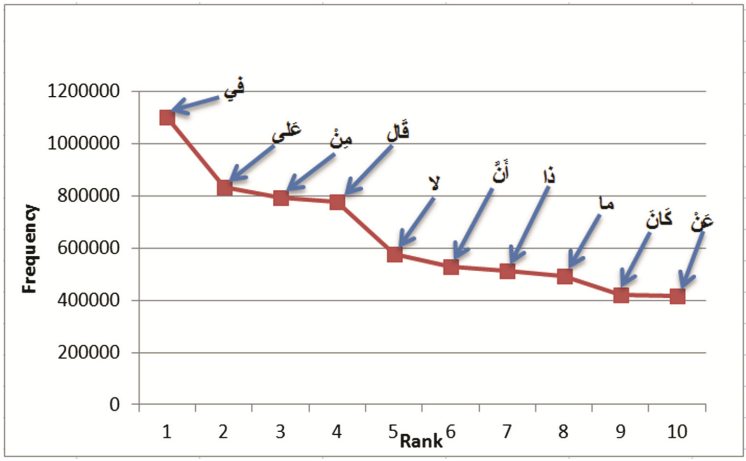


Fig. 2. Distribution of the ten most frequent Arabic lemmas

After the frequent lemmas list has been obtained, we checked to what extent the top 1,000 frequent are covered by our speech corpora.

To this end, we processed an automatic full vocalization of the text transcriptions using Al Jazeera vocalization tool [19], after that we have calculated the intersections between our speech corpus and the top 1,000 lemmas.

As a result, 53.28% of the top 1,000 lemmas were found to be covered by our speech corpus. Each lemma is available in different.

5 Lemma Database Construction

The most important step in designing a TTS system is the preparation of the speech database. Generally this process involves three main steps: preparation and collection of texts to be recorded, the corpus recording and finally, the segmentation and annotation. In this work, the process of the speech database construction was limited to two stages: database preparation and the segmentation.

5.1 Database Preparation

The recording of large inventory requires a strict recording procedure to assure the uniformity of the database and long hours of recording sessions. Due to the high cost of the recording process, we decided to use a pre recoded audio book from Masmoo3 Website [25]. This latter was recorded by a native professional speaker with natural prosody and using phonetically balanced sentences. Almost all the recorded sentences were statements. The audio book contains 1148 sentences, summing up to 4 h 20 min of speech. The recordings were split up into 98 files with the average file length of 2.5 min. Each file was provided with its corresponding orthographic transcription file in .doc format.

Afterward, we performed manual text normalization to the entire transcription file, by expanding numbers, signs, and abbreviations in the textual form. Furthermore, a full vocalization was performed automatically using Al Jazeera vocalization tool [19]. A manual spell checking was carried out in order to ensure the correct matching between the recordings and text spellings.

5.2 Segmentation

The final step in the speech database construction is the segmentation of the speech into the target utterances. In our case, we aim to use a mixed inventory of both lemmas and sub-segments, taking advantage of their respective strengths and compensating for their respective weakness.

The segmentation of the corpus was performed in two ways:

- **Manual lemma-level segmentation:** First, the long segments which are lemmas were located in the speech database and then segmented from the boundary of the first consonant of the lemma to the half of the last consonant of the same lemma. The last vowel of the lemma is truncated in order to make the lemma unit flexible and adapted to the different inflected forms. For example, both words "فَقَلوا" /faqalwA/ and "فَقَالَتْ" /faqaAlato/ are generated from the lemma "قَال" /qaAl/ without its last vowel. This process of lemma-level segmentation was handmade by the software Praat [26] in order to achieve a high accuracy of the boundaries selection.
- **Automatic sub-segments level segmentation:** The process of segmenting and alignment of the sub-segments utterances is being performed automatically by an HMM system.

The results of this segmentation provide a mixed speech database of both the most frequent lemmas in the Arabic language and sub-segments.

6 Experiment and Results

The evaluation of the proposed Arabic TTS system was performed in term of intelligibility and naturalness. Both tests were conducted with five Arabic participants.

- **Intelligibility Assessment test:** The Diagnostic Rhyme Test was applied on the word level and sentence level. For the word perception test, a questionnaire was specially prepared to perform the listening test containing twenty pairs of words that differ only in a single consonant. From each pair, we played one word at a time and then the participants were asked to mark on the answering sheet which word of each pair of the words they think is correct. The average of word perception rate is 100% since all the words were successfully perceived by the listeners. The sentence perception test was conducted with the same subjects, where a set of ten Arabic sentences with different length and complexity were played twice. The participants were asked to write down the sentence they hear exactly. All the sentences were correctly perceived except one, which is due to the short duration of some words composing the sentences. Table 5 presents the success rate of the intelligibility tests.

Table 5. The intelligibility assessment results

Sentence-level test (DRT)	Word-level test
90%	100%

- **Naturalness Assessment test:** For the second test, the Mean Opinion Score (MOS) test was applied to assess the quality of the produced speech. The primary intention of this evaluation was to compare the quality of sub-segments based TTS with the result of lemma-sized and sub-segments merge and to ensure that the former approach will increase the naturalness of the produced speech. The naturalness evaluation was conducted in two stages; first, a set of 10 sentences were synthesized using the sub-segments database. The participants were asked to give their opinion regarding the quality from a scale of 1 to five. In the second stage, the same sentences were synthesized but this time using the proposed approach based on both lemmas and sub-segments. Again the participants hear the sentences and were asked to give their judgment score.

To ensure the validity of the obtained results, tests were conducted in a comparable approach with other available Arabic TTS systems. The Demo version of Acapela was used for this purpose, as well as Euler/Mbrola TTS system. The following charts illustrate the results of the MOS evaluation comparing our proposed approach against Acapela and Euler systems (Figs. 3 and 4).

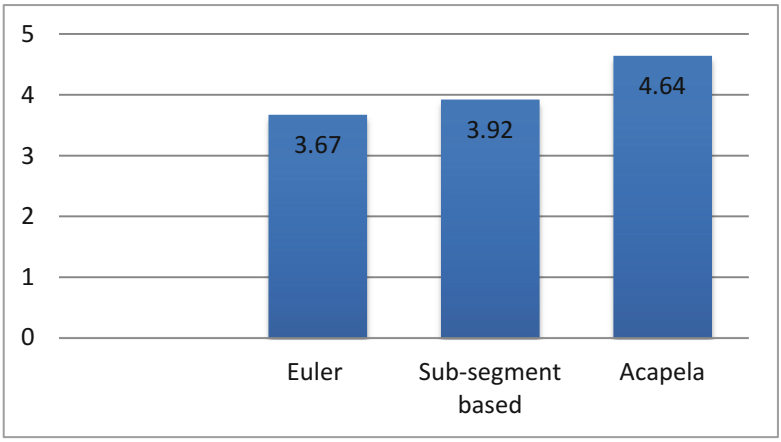


Fig. 3. Naturalness average scores (MOS) for Euler and Acapela compared to our sub-segments based system

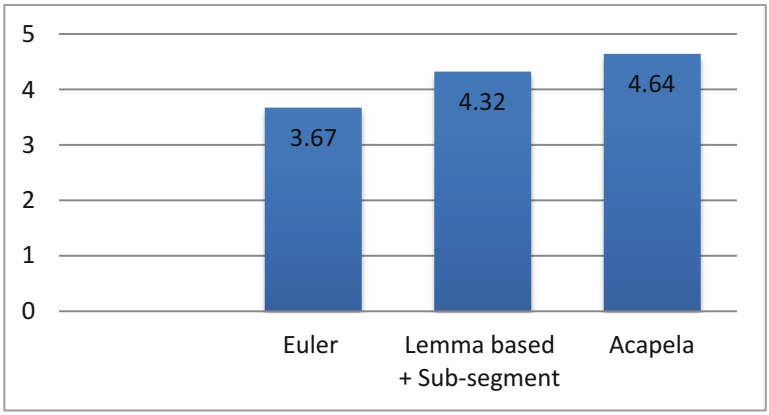


Fig. 4. Naturalness average scores (MOS) for Euler and Acapela compared to our lemma-based approach merged with the sub-segment approach

As we can see from the results, merging the lemmas with sub-segments had a significant positive impact. Furthermore, the results of the comparison of our system against the Acapela and Euler/Mbrola showed a significant reduction of the gap between the best commercial system Acapela and the available free solution Euler/Mbrola.

7 Conclusion and Future Work

In this paper, we have presented a unit selection approach that combines long units as lemmas with sub-segments. Besides, to cope with the large number of Arabic lemmas, we have described the materials and methods used for generating the lemma frequency list, as well as the process of construction of the lemma database. Finally, we presented the results of merging sub-segments and lemma units which have shown significant improvement in both intelligibility and naturalness of the produced speech.

Our next step is to increase the number of the stored lemmas from 1000 to 5000 or more of frequent lemmas in order to achieve better coverage of the Arabic texts. To meet this goal, a much larger set of speech materials is needed.

Apart from this, although our corpus was designed to cover several candidates to the same lemma, the prosodic context is currently not considered in our selection criteria, this is due to the limited number of the candidates of each lemma. We plan to cope with this problem by using the augmented speech database.

Acknowledgment. The authors gratefully acknowledge and thank Masmoo3 Team for providing us with the Arabic audio files used to build our speech corpus.

References

1. Chabchoub, A., Alahmadi, S., Cherif, A., Barkouti, W.: Di-Diphone Arabic speech synthesis concatenation. Int. J. Comput. Technol. **3**, 218–222 (2012)
2. Zen, H., Tokuda, K., Black, A.W.: Statistical parametric speech synthesis. Speech Commun. **51**, 1039–1064 (2009). https://doi.org/10.1016/j.specom.2009.04.004
3. Zine, O., Meziane, M.: Novel approach for quality enhancement of Arabic Text To Speech synthesis. In: Presented at 3rd International Conference on Advanced Technologies for Signal and Image Processing, ATSIP 2017 (2017). https://doi.org/10.1109/ATSIP.2017.8075550
4. Bozkurt, B., Öztürk, Ö., Dutoit, T.: Text design for TTS speech corpus building using a modified greedy selection. In: INTERSPEECH (2003)
5. Khan, R.A., Chitode, J.S.: Concatenative speech synthesis: a review. Int. J. of Comput. Appl. **136**(3), 1–6 (2016). https://doi.org/10.5120/ijca2016907992
6. Hande, S.S.: A review of concatenative text to speech synthesis. Int. J. Latest Technol. Eng. Manag. Appl. Sci. IJLTEMAS **3**(9), 12–15 (2014)
7. Hamacher, V., Chalupper, J., Eggers, J., Fischer, E., Kornagel, U., Puder, H., Rass, U.: Signal processing in high-end hearing aids: state of the art, challenges, and future trends. EURASIP J. Appl. Sig. Process. **2005**, 2915–2929 (2005)
8. Gonzalvo, X., Tazari, S., Chan, C., Becker, M., Gutkin, A., Silen, H.: Recent advances in Google real-time HMM-driven unit selection synthesizer. Presented at the September 8 (2016)
9. Abdelmalek, R., Mnasri, Z.: High quality Arabic text-to-speech synthesis using unit selection. In: 2016 13th International Multi-Conference on Systems, Signals and Devices (SSD), pp. 1–5. IEEE (2016)
10. Rashad, M.Z., El-Bakry, H.M., Isma'il, I.R.: Diphone speech synthesis system for Arabic using MARY TTS. Int. J. Comput. Sci. Inf. Technol. **2**, 18–26 (2010). https://doi.org/10.5121/ijcsit.2010.2402
11. Alsharif, B., Tahboub, R., Arafeh, L.: Arabic text to speech synthesis using quran-based natural language processing module. J. Theor. Appl. Inf. Technol. **83**, 148 (2016)
12. Husni-Al-Muhtaseb, M.E., Al-Ghamdi, M.: Techniques for high quality arabic speech synthesis. Computer Science and Engineering, King Fahd University of Petroleum and Minerals (2003)
13. Campbell, N.: Conversational speech synthesis and the need for some laughter. IEEE Trans. Audio Speech Lang. Process. **14**, 1171–1178 (2006). https://doi.org/10.1109/TASL.2006.876131
14. Dutoit, T., Pagel, V., Pierret, N., Bataille, F., van der Vrecken, O.: The MBROLA project: towards a set of high quality speech synthesizers free of use for non commercial purposes. In: Proceedings of the Fourth International Conference on Spoken Language, ICSLP 1996, vol. 3, pp. 1393–1396 (1996)
15. MaryTTS – Overview. http://mary.dfki.de/documentation/overview.html
16. Karabetsos, S., Tsiakoulis, P., Chalamandaris, A., Raptis, S.: Embedded unit selection text-to-speech synthesis for mobile devices. IEEE Trans. Consum. Electron. **55**, 613–621 (2009)
17. Buckwalter, T., Parkinson, D.: A Frequency Dictionary of Arabic: Core Vocabulary for Learners. Routledge, London (2014)
18. Zaghouani, W., Bouamor, H., Hawwari, A., Diab, M., Obeid, O., Ghoneim, M., Alqahtani, S., Oflazer, K.: Guidelines and framework for a large scale Arabic diacritized corpus. In: The Tenth International Conference on Language Resources and Evaluation (LREC 2016), pp. 3637–3643 (2016)

19. Aljazeera Network, Aljazeera Learning Arabic Service 2016. http://learning.aljazeera.net/arabic. Accessed 10 Aug 2017
20. Belinkov, Y., Magidow, A., Romanov, M., Shmidman, A., Koppel, M.: Shamela: a large-scale historical arabic corpus. arXiv Preprint arXiv:161208989 (2016)
21. Yaseen, B.: Language technology for Arabic. NEMLAR, Center for Sprog-teknologi, Univ. of Copenhagen, Copenhagen (2005)
22. Zeroual, I., Lakhouaja, A.: A new Quranic Corpus rich in morphosyntactical information. Int. J. Speech Technol. **19**, 339–346 (2016). https://doi.org/10.1007/s10772-016-9335-7
23. Boudchiche, M., Mazroui, A., Ould Abdallahi Ould Bebah, M., Lakhouaja, A., Boudlal, A.: AlKhalil Morpho Sys 2: a robust Arabic morpho-syntactic analyzer. J. King Saud Univ. Comput. Inf. Sci. **29**(2), 141–146 (2017). https://doi.org/10.1016/j.jksuci.2016.05.002
24. Boudchiche, M., Mazroui, A.: Approche hybride pour le développement d'un lemmatiseur pour la langue arabe. Presented at the 13th African Conference on Research in Computer Science and Applied Mathematics, Hammamet, Tunisia (2016)
25. Masmoo3 - Arabic Audio Books. http://www.masmoo3.com/. Accessed 10 Aug 2017
26. Boersma, P., Weenink, D.: Praat: doing phonetics by computer [Computer program]. Version 6.0.29. http://www.praat.org/. Accessed 24 May 2017

Towards a Speech Recognizer for Multiple Languages Using Arabic Acoustic Model Application to Amazigh Language

Ali Sadiqui[1,2(✉)] and Ahmed Zinedine[1,2(✉)]

[1] OFPPT, ISTA Meknès, Meknès, Morocco
sadiqui2000@yahoo.fr, ahmedzinedine@yahoo.com
[2] Faculté des Sciences Dhar El Mehraz, Atlas, B.P. 1796, Fès, Morocco

Abstract. The construction of acoustic models of a language, used in automatic speech recognition (ASR) systems, is a developed technology achievable without great difficulty when a large amount of speech and written corpus is available. However, these technological resources are not available in a large part of languages called "Less Resourced Languages". An alternative solution is to take advantage of the phonetic structures shared between the different languages to build an acoustic model for the target language.

In this paper, we will return to an experiment in this direction. Indeed, we used an acoustic model of the Arabic language to create one for the Amazigh language. The originality of our work comes from the will to address this language which has become an official language in Morocco, and which has not enough resources for the automatic speech recognition. In addition, both languages share several phonemes and certain characteristics. The realized system has reached a recognition rate of about 73% by word. The potential and the effectiveness of the proposed approach is demonstrated by experiments and comparison with other approaches.

Keywords: Automatic speech recognition · Acoustic model · Arabic Amazigh · CMU sphinx

1 Introduction

A speech recognition system (ASR) in general, refers to a system able to convert a signal produced by a speaker into a sequence of words corresponding to the underlying linguistic message. This type of tools is widely used in various fields such as dictation applications, assistive software for people with disabilities, voice control machines, etc.

The general principle of an ASR is to extract the information content of a speech signal to convert it to text. This processing strongly developed in parallel with the evolution of the means and techniques. It passed from an example-based recognition to a model-based one.

Almost all the automatic current speech recognition systems are based on statistical modeling. They can be broken down into modules as shown in Fig. 1.

Indeed, once the sound is emitted from the speaker, it passes through acoustic analysis which represent the signal of speech in a more suitable form for the process of

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 67–78, 2018.
https://doi.org/10.1007/978-3-319-73500-9_5

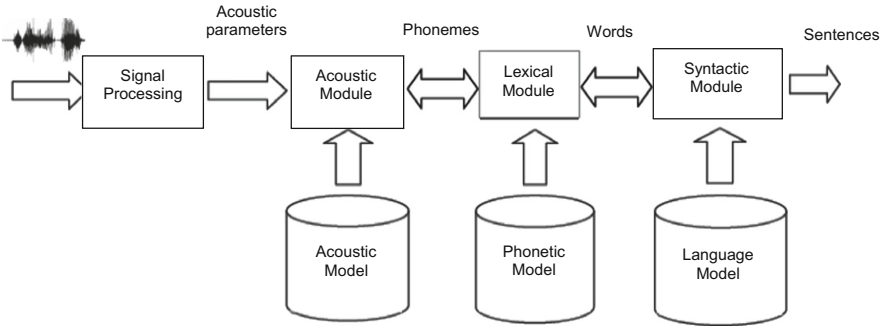


Fig. 1. Automatic recognition of speech by statistical modeling [1].

recognition. The following representations: (Mel Frequency Cepstral Coefcients), LPCC (Linear Predictive Cepstral Coefcients) or PLP (Perceptual Linear Predictive analysis are generally the most employed in this area [2–4]. These settings are, then sent to the acoustic recognizer. This latter detects the most likely produced acoustic unit. This unit can be a word or derivatives of words such as phonemes or syllables.

Then, the acoustic module is based on the lexical module to return words pre-configured and saved in the dictionary of the ASR. In case the acoustic unit is a phoneme or a syllable, one speaks of a pronunciation dictionary. It allows to associate each word in the dictionary with a sequence of acoustic units according to its pronunciation.

In the case of automatic recognition of large vocabulary continuous speech, the system interacts with a syntactic module to incorporate syntactic or semantic constraints. These constraints are usually created from the language models.

The modern creation of the ASR process has recently become a fairly easy task and has developed considerably with the evolution of the used tools. However, this treatment is subject to the constraints of technology resources to build the already explained components. Indeed, the availability of labeled voice data (voice corpus) in particular becomes for certain languages a costly and not accessible task. One of the possible solutions is to adapt an already made acoustic model to create one for the target language. This technique bears the name of: cross lingual acoustic modeling [4–6].

This approach aims, deeply, to study the possibility of creating multi-language acoustic templates [7–9]. The idea is to characterize all possible phonemes to cover all languages. Given that several phonemes are shared by multiple languages, the number of possibilities becomes limited.

In this work, and after developing a multi speaker continuous speech ASR for the modern Arabic language, we study the possibility of using the acoustic model already produced to create an ASR of the Amazigh language. This study is based firstly on the fact that this language has a shortage of technical resources for the creation of its own ASR and secondly on the fact that the Arabic and Amazigh languages share certain characteristics In particular the fact that they share certain phonemes, several words and the use of the phoneme “Shadda”. Therefore, our goal seems promising.

In the first section of this article, we begin by introducing the technique of portability to theoretically explain this approach and the used methods. In the second section, we will present our performed Arabic ASR as well as a phonetic study of the Amazigh language. Finally, we present the achieved system and then we analyze the obtained results before concluding.

2 Multilingual International Phonetics Recognition

With the emergence of the automatic multilingual speech recognition, several solutions based on multilingual acoustic models, are proposed. They are based, in general, on creating a table of phonemic correspondences (phone mapping table) between a source language (monolingual case) or multiple languages (multilingual case) and the target language. For this, there are two methods: manual knowledge-based methods and automatic data-driven methods [10–12].

Manual methods include searching closest source/target phoneme couples in the API table (Fig. 2). This approach requires acoustic and phonetic knowledge of both languages (source and target) [13].

CONSONANTS (PULMONIC)

	Bilabial	Labiodental	Dental	Alveolar	Postalveolar	Retroflex	Palatal	Velar	Uvular	Pharyngeal	Glottal
Plosive	p b			t d		ʈ ɖ	c ɟ	k g	q ɢ		ʔ
Nasal	m	ɱ		n		ɳ	ɲ	ŋ	N		
Trill	B			r					R		
Tap or Flap		ⱱ		ɾ		ɽ					
Fricative	ɸ β	f v	θ ð	s z	ʃ ʒ	ʂ ʐ	ç ʝ	x ɣ	χ ʁ	ħ ʕ	h ɦ
Lateral fricative				ɬ ɮ							
Approximant		ʋ		ɹ		ɻ	j	ɰ			
Lateral approximant				l		ɭ	ʎ	ʟ			

Where symbols appear in pairs, the one to the right represents a voiced consonant. Shaded areas denote articulations judged impossible.

Fig. 2. Notation API for consonants [15].

Automatic methods consist in having a labeled vocal corpus, in limited quantity in some cases, in target language, then look for the closest source/target phoneme couples following a function of distance between phonemes or following the matrix of confusion of source/target phonemes.

With this in mind, the CMU Sphinx project [14] offers the possibility to rely on already existing acoustic models for use in another language. These acoustic models proposed by CMU Sphinx, have been during years of experience, carefully optimized for best performance of recognition and are suitable for almost all applications.

In addition, the work of CMU project GlobalPhone [15, 16] is to design an automatic multilingual speech recognition system based on phonemes. From the inventory of the international phonetic alphabet, they defined a set of phonemes named Global Phoneme Set. Figure 3 is an example of all of the global phonemes obtained with 5 languages (Croatian, Japanese, Korean, Spanish and Turkish) in notation Worldbet. We notice the presence of 78 phonemes 14 of which are shared between 5 languages, and half of this set belongs to only one language. Therefore the total number of acoustic units to model is reduced to 78 multilingual phonemes against 170 monolingual phonemes in the case of the 5 GlobalPhone languages.

Phonèmes [WorldBet]	KO	SP	CR	TU	JA	Σ
n, m, s, l, tS, p, b, t, d, g, k, i, e, o	x	x	x	x	x	14
f, j, z		x	x	x	x	
r, u	x	x	x	x	x	6
dZ	x		x	x		
a	x	x	x			
S			x	x	x	
h	x			x	x	4
4	x	x			x	
n̄, x, L		x	x			
A				x	x	
N	x	x				
V, Z			x	x		10
y, 7	x			x		
ts			x		x	
p', t', k', dZ', s', oE, oa, 4i, uE, E, /\, i/\, u/\, iu, ie,	x					17
io, ia	x					
D, G, T, V, r(, ai, au, ei, eu, oi, a+, e+, i+, o+, u+		x				15
palatal c, palatal d			x			2
ix, soft				x		2
?, Nq, V[, A:, e:, i:, o:, 4:					x	8
Monolingue Σ = 170	40	40	30	29	31	
Multilingue						78

Fig. 3. The phonemes of the training database of GlobalPhone.

However, this approach presents some challenges. In addition to the choice of the total size of these units, the definition of units depending on the context, among other things, is another constraint. Indeed, the use of acoustic data from several source languages creates the possibility to characterize several contexts of phonemes that may

not exist in the target language, which may cause degradation in the quality of the acoustic model to create.

3 Presentation of the Used Arabic Acoustic Model

The acoustic model made for the Arabic language is based on University Carnegie Mellon Sphinx tools. Indeed, we used as units a context phoneme. Each context phoneme is modeled by a hidden Markov chain in 3 states with multi-Gaussian densities of observation of number 8 (Tables 1 and 2).

Table 1. List of used Arabic phonemes and their notation.

Phonemes	Notation		Phonemes	Notation
ء	E		ق	Q
ب	B		ك	K
ت	T		ل	L
ث	Th		م	M
ج	J		ن	n
ح	H_		هـ	h
خ	X		و	w
د	D		ي	y
ذ	V		الجيم المصرية (g)	G_
ر	R		P -اسبانيا>	p
ز	Z		v -نوڤبر>	V_
س	S		الفتحة -> بَ	a
ش	sh		المد->با	A_
ص	S_		الضمة -> بُ	u
ض	D_		المد -> بو	U_
ط	T		الكسرة -> بِ	i
ظ	Z_		المد -> بي	I_
ع	c		الشدة	اعادة الحرف
غ	g		التنوين	a n / u n / i n
ف	f			

Table 2. Example of Latin notation of Arabic words and their phonetic decomposition.

Arabic word	Latin notation	Phonetic decomposition
وسيمون	wasI_mU_n	w a s I_ m U_ n
غالبا	gA_liban	g A_ l i b a n
سجود	Sujud	s u j U_ d
أحرز	euH_riza	e u H_ r i z a
رقبة	Raqaba	r a q a b a
قوة	Quwwa	q u w w a

The total length of the used corpus was 13 h 30 min, 20% of which has been used for the performance test and the rest for the creation of the acoustic model. The used corpus is recovered, in large part from [17, 18]. The model has resulted in a multi-speaker system of continuous speech with a rate of word recognition of about 81% (Fig. 4).

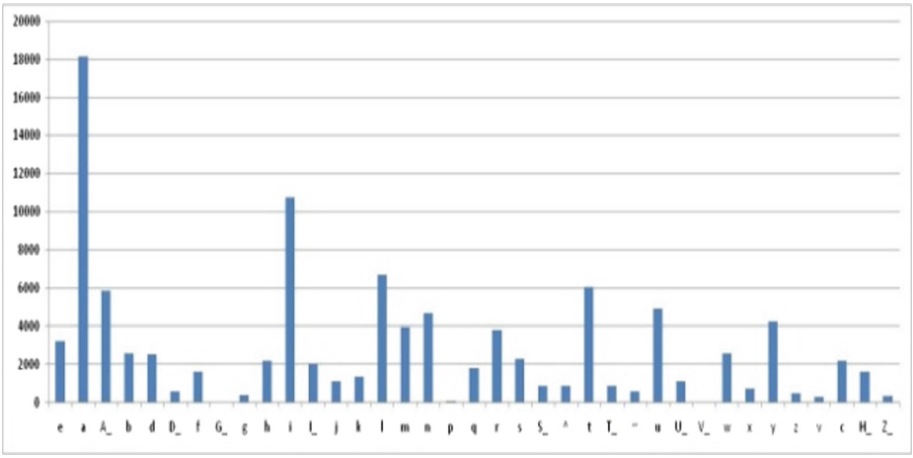


Fig. 4. Distribution of Arabic phonemes in the used corpus.

4 Presentation of the Amazigh Language

The Amazigh language is a branch of the Afro-asiatic languages (Hamito Se-mitic) [19–21]. Today, it covers the northern part of Africa, and also extends to include the Canary Islands, Niger and Mali. In Morocco, the Amazigh language is present in the form of several dialects.

Recently this language has seen a considerable evolution. In fact, in 2011, it became an official language in Morocco, and several institutes ensure its development and its dissemination, including the Royal Institute for the Amazigh Culture (IRCAM) [22] that has been created since 2003.

Over the last 10 years of its creation, IRCAM has published more than 150 books related to the Amazigh culture and language. However, in terms of processing of natural language (ALP) this language, like many non-European, languages, is still suffering from scarcity of resources and language processing tools.

5 The Graphic System Tifinaghe-IRCAM

Since 2003, Tifinaghe-IRCAM became the official graphic system to write the Amazigh in Morocco. This system contains:

- 27 consonants: ⴱ, ⴳ, ⴳⵯ, ⴷ, ⴹ, ⴼ, ⴽ, ⴽⵯ, ⵀ, ⵃ, ⵄ, ⵅ, ⵇ, ⵉ, ⵊ, ⵍ, ⵎ, ⵏ, ⵔ, ⵚ, ⵜ, ⵟ, ⵡ, ⵣ, ⵥ;
- 2 semi-vowels: ⵢ, ⵯ;
- 4 vowels: ⴰ, ⵄ, ⵉ and ⵓ.

In addition, no particular punctuation is known for Tifinaghe. IRCAM has recommended the use of the international symbols of punctuation (Fig. 5).

Fig. 5. The alphabet tifinaghe-IRCAM

6 Our Contribution

6.1 Presentation of the Corpus

Our Amazigh corpus consists of 610 files and a dictionary of 603 words. This corpus is retrieved from an educational CD of the Amazigh language.

The audio files were recorded in a studio by several speakers. They were invited to read one or more prearranged Amazigh words. The signal acquisition was performed with 16-bit precision and a sampling frequency of 16 kHz (Fig. 6).

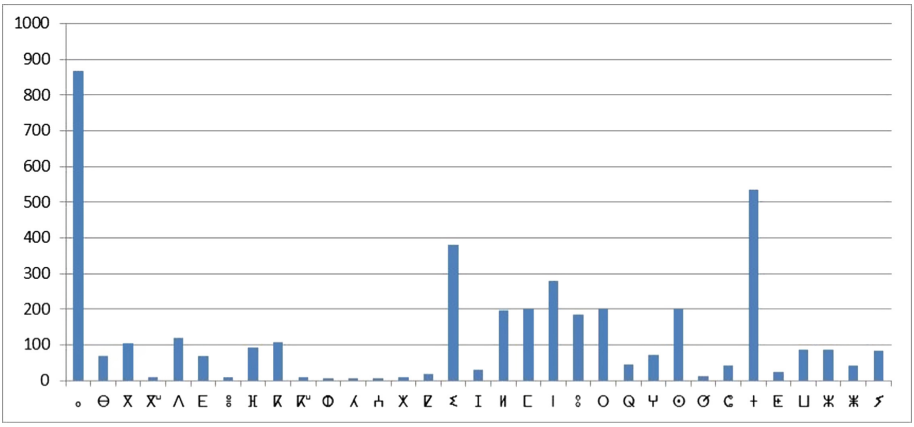


Fig. 6. Distribution of Amazigh phonemes in the studied corpus.

The corpus is spread over 22 directories. Each directory groups the words of the same category. The chosen textual data cover different areas of daily life: nature, everyday life, etc. (Table 3).

Table 3. The decomposition of the used corpus.

	Series #1	Series #2
The number of files	488	122
Average length of a file	1–3 s	1–3 s
Type of use	Training	Test

6.2 Creation of Pairs of Source and Target Phonemes

A first reflection enabled us to develop an array of phonetic correspondence linking each phoneme of the Amazigh language with its closest equivalent in the Arabic language.

Note that, aware that the phonemes ⵅ, ⴽ, ⵇ, ⵣ and ⵖ don't have their correspondents in the Arabic language, we tried as a first step, to test an approximate match of such phonemes as a result of the approach of manual correspondence already explained. The table of the source/target pairs is presented in Table 4.

A test of decoding has been done according to this manual mapping. It has led to a recognition rate by words in the range of 51% (Table 5).

Table 4. The phonetic matching table used in the Arabic acoustic model.

Amazigh phoneme	Arabic phoneme		Amazigh phoneme	Arabic phoneme
ⵄ	الفتحة -> بَ		ⵊ	ج
ⵒ	ب		ⵍ	ل
ⵅ	الجيم المصرية		ⵎ	م
ⵅ$^{\omega}$	**الجيم المصرية**		ⵏ	ن
ⵑ	د		ⵓ	***الضمة -> بُ***
ⵟ	ض		ⵔ	ر
ⵓ	/لضمة -> بُ		ⵕ	ر
ⵖ	ف		ⵖ	غ
ⵇ	ك		ⵙ	س
ⵇ$^{\omega}$	**ك**		ⵚ	ص
ⵀ	هـ		ⵛ	ش
ⵃ	ح		ⵜ	ت
ⵄ	ع		ⵟ	ط
ⵅ	خ		ⵡ	و
ⵣ	ق		ⵣ	ز
ⵉ	الكسرة -> بِ		ⵥ	**ظ**
الشدة	اعادة الحرف		ⵢ	ي

Table 5. Amazigh notating of words and phonetic decomposition.

Amazigh word	Phonetic decomposition in Latin notation
ⴰⵅⴰⵎⴰ	E A G A M A
ⵜⴰⵅⴰⵏⵜ	T A G A N T
ⵜⵓⵔⵜⵉⵜ	T U R T I T
ⵜⴰⵖⵓⵇⵜ	T A F U K T
ⵜⴰⵙⵍⵉⵜ	T A S L I T
ⵓⵏⴷⴰⵔ	E U N D_A R
ⴰⵢⵢⵓⵔ	E A Y Y U R
ⵜⴰⵣⵉⵔⵉ	T A Z I R I
ⵉⵜⵔⴰⵏ	E I T R A N
ⵜⴰⴷ2ⴷ2ⴰⵏⴳⴰ	T A D2 D2 A N G A

This result confirmed the validity of the choice and encouraged us to proceed to the second step.

6.3 Adaptation of the Arab Acoustic Model by the Method of Cross-Ling

There are two well-known methods of adaptation that are in the community of automatic recognition of the word: MLLR (Maximum Likelihood Linear Regression) and MAP (Maximum A Posteriori).

MLLR [23] is a technique of adaption based on a linear transformation of the parameters of acoustic models.

The MAP adaptation [24] allows a re-assessment of the HMM parameters observed in the data of adaptation. For a small amount of data the MLLR method is supposed to be better. But for a large amount of data MAP showed its superiority on MLLR [25].

We used in our work, given the limited amount of the corpus at our disposal, the MLLR adaptation technique to adapt the Arabic acoustic model to target language.

The procedure of using the tool CMU Sphinx to perform this step is explained in detail in [26].

We used 116 words (about 20%) of the Amazigh corpus as a corpus of adaptation and the rest for the performance test. We kept the same phoneme coupling (source/target) presented in Table 4.

The test resulted in a by- word recognition rate of 73%.

The final system was able to decode both Arabic and Amazigh words (Table 6).

Table 6. Presentation of the obtained results.

Language	Total words tested	Total decoded words using Arabic ASR only	Total words decoded using Arabic ASR with adaptation	Recognition rate
Arabic	6137	4972	4897	81% | 79%
Amazigh	116	60	84	51% | 73%

7 Conclusion

The system that we have achieved has shown very satisfactory results with a high recognition rate. We consider, however, that it is a preliminary work that will allow us to fulfill a more evolved recognition system for the Amazigh language. It also showed us that the Arabic language, with its wealth of phonemes and characteristics, could certainly be a very powerful source language for the creation of the ASR for the less resourced languages especially those that have been for centuries influenced by the Arabic language and civilization (some of the Western European languages and Asian countries as examples).

References

1. Lê, V.B.: Reconnaissance automatique de la parole pour des langues peu dotées. thèse de doctorat, Joseph Fourier - Grenoble1 (2006)
2. Rabiner, L.-R., Schafer, R.-W.: Digital Processing of Speech Signals. Prentice-Hall, Englewood Cliffs (1978)
3. Jurafsky, D., Martin, J.H.: Speech and Language Processing, 2nd edn. Prentice Hall Inc, Englewood Cliffs (2008). Chapter 9 to end of Sect. 9.3
4. Boite, R., Bourlard, H., Dutoit, T., Hancq, J., Leich, H.: Traitement de la parole. Presses Polytechniques et Universitaires Romandes, Collection Electricité, Lausanne, Switzerland (2000)
5. Schultz, T., Waibel, A.: Language independent and language adaptive acoustic modeling for speech recognition. Speech Commun. **35**, 31–51 (2001)
6. Lin, H., Deng, L., Droppo, J., Yu, D., Acero, A.: Learning methods in multilingual speech recognition. In: Proceedings of the NIPS, Vancouver, BC, Canada (2008)
7. Byrne, W., et al.: Towards language independent acoustic modeling. In: Proceedings of the ICASSP (2000)
8. Van Doremalen, J., Cucchiarini, C., Strik, H.: Optimizing automatic speech recognition for low-proficient non native speakers. EURASIP J. Audio Speech Music Process. **2010**, 1–13 (2010)
9. Heigold, G., Vanhoucke, V., Senior, A.W., Nguyen, P., Ranzato, M., Devin, M., Dean, J.: Multilingual acoustic models using distributed deep neural networks. In: Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, pp. 8619–8623 (2013)
10. Garcia, E., Mengusoglu, E., Janke, E.: Multilingual acoustic models for speech recognition in low-resource devices. In: Proceedings of the ICASSP (2007)
11. De Wachter, M., Demuynck, K., van Compernolle, D., Wambaq, P.: Data driven example based continuous speech recognition. In: Proceedings of Eurospeech, Geneva, Switzerland, pp. 1133–1136 (2003)
12. Schultz, T., Kirchhoff, K. (eds.): Multilingual Speech Processing. Academic Press, Amsterdam (2006)
13. International Phonetic Association: Handbook of the International Phonetic Association: A Guide to the Use of the International Phonetic Alphabet, pp. 1–204 (1999)
14. Open source speech recognition toolkit CMU sphinx. https://sourceforge.net/projects/cmusphinx/files/Acoustic%20and%20Language%20Models/
15. Schultz, T.: GlobalPhone: A multilingual speech and text database developed at karlsruhe university. In: ICSLP 2002, Denver, CO, USA, Septembre 2002
16. The GlobalPhone Project: http://www.cs.cmu.edu/~tanja/GlobalPhone
17. Ali Sadiqui, Nouredine Chenfour, Réalisation d'un système de reconnaissance automatique de la parole arabe basée sur CMU Sphinx, article publié sur «Annals. Computer Science Series» Tome 8, Avril 2010
18. Arabic Speech Corpus. http://en.arabicspeechcorpus.com/
19. Greenberg J.: The Languages of Africa. The Hague (1966)
20. Ouakrim, O.: Fonética y fonología del Bereber, Survey at the University of Autònoma de Barcelona (1995)
21. El Barkani, B.: Le choix de la graphie tifinaghe pour enseigner, apprendre l'amazighe au Maroc: conditions, reprrésentation et pratiques. Linguistique. Université Jean Monnet - Saint-Etienne. Français (2010)
22. The Royal Institute of Amazigh Culture. http://www.ircam.ma

23. Leggetter, C.-J., Woodland, P.-C.: Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models. Comput. Speech Lang. **9**(2), 171–185 (1995)
24. Gauvain, J.-L., Lee, C.-H.: Maximum a posteriori estimation for multi-variate gaussian mixture observations of markov chains. IEEE Trans. Speech Audio Process. **2**(2), 291–298 (1994)
25. Wang, Z., Schultz, T.: Non-native spontaneous speech recognition through polyphone decision tress specialization. In: Eurospeech 2003, pp. 1449–1452, Geneva, Switzerland, September 2003
26. Open source speech recognition toolkit CMU sphinx. http://cmusphinx.sourceforge.net/wiki/tutorialadapt

A Game with a Purpose for Automatic Detection of Children's Speech Disabilities Using Limited Speech Resources

Reem Elhady, Mohamed Elmahdy(✉), Injy Hamed, and Slim Abdennadher

Computer Science Department, Faculty of Media Engineering and Technology, German University in Cairo, Cairo, Egypt
reem.elhady@student.guc.edu.eg,
{mohamed.elmahdy,injy.hamed,slim.abdennadher}@guc.edu.eg

Abstract. Speech therapists and researchers are becoming more concerned with the use of computer-based systems in the therapy of speech disorders. In this paper, we propose a computer-based game with a purpose (GWAP) for speech therapy of Egyptian speaking children suffering from Dyslalia. Our aim is to detect if a certain phoneme is pronounced correctly. An Egyptian Arabic speech corpus has been collected. A baseline acoustic model was trained using the Egyptian corpus. In order to benefit from existing large amounts of Modern Standard Arabic (MSA) resources, MSA acoustic models were adapted with the collected Egyptian corpus. An independent testing set that covers common speech disorders has been collected for Egyptian speakers. Results show that adapted acoustic models give better recognition accuracy which could be relied on in the game and that children show more interest in playing the game than in visiting the therapist. A noticeable progress in children Dyslalia appeared with the proposed system.

Keywords: Dyslalia · GWAP · Arabic · Speech recognition
Speech disorders

1 Introduction

Speech and language disorders can affect a person's ability to talk, understand, read, write and express himself/herself. In this study, we are concerned with Dyslalia speech disorder. Dyslalia is an articulatory disorder in which children, or adults, do not pronounce the sounds clearly, sounds are changed or distorted or they replace one sound for another, e.g. a person may have a lisp use of the /T/ instead of the /s/ sound[1]. It is a result of having the sound pronounced from an incorrect part of the vocal tract. It may be also due to delayed speech, hearing impairment or learning disability. Mental retardation can also cause Dyslalia.

In the process of therapy, speech therapists use a variety of strategies including oral motor therapy, articulation therapy, and language intervention activities.

[1] Throughout the paper, SAMPA notation is used for phonetic transcriptions [16].

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 79–89, 2018.
https://doi.org/10.1007/978-3-319-73500-9_6

In the step of oral motor therapy, the therapist uses a variety of oral exercises, including facial massage and various tongue, lip, and jaw exercises, to strengthen the muscles of the mouth. The therapists physically show the child how to make certain sounds, such as the /r/ sound, and may demonstrate how to move the tongue to produce specific sounds. During the language intervention activities, the therapist interacts with a child by playing and talking. They may use pictures, books, objects, or ongoing events to stimulate language development. The therapist may also model correct pronunciation and use repetition exercises to build speech and language skills.

Our proposed system plays the role of language intervention activities. Therapists not only have to provide a variety of materials for different tasks for each therapy session, but they also have to keep a record of the child's performance during the tasks. Moreover, the therapist usually has problems to manage the recording details of these sessions for further analysis and to prepare appropriate materials that address the required treatment process.

Several studies mention the importance of computer-based systems that aim at supporting such therapies [4,14,15]. All of these studies have focused on Latin languages like English and Romanian.

There were specific criteria chosen in the proposed game design [6] for best effect: attractiveness, curiosity, immediate and accurate feedback, the issue of control, challenge feeling, and automatic system adaptation to user's performance. The experiments were done on the most common problems in the Arabic Egyptian language which are the replacement of the (س) /s/ phoneme to (ش) /S/, (ث) /T/, (د) /d/, or (خ) /x/; and the replacement of (ر) /r/ phoneme to (ل) /l/, (ي) /i:/, or (غ) /G/ [11,13].

2 Speech Engine

In order to have a game that improves child Dyslalia, there should be first a speech engine that can accurately and automatically detect the problem in the child pronunciation, and then the game shall take its role. So, we have worked separately on speech engine, and after we had successfully ensured recognition accuracy of the engine, we have merged it to an attractive game to test the effect of the game therapeutically. A major problem in Arabic speech recognition is the existence of quite many different Arabic dialects. Every country has its own dialect and sometimes there exist different dialects within the same country. There are many speech data resources for MSA, but unfortunately, the available resources for dialectal Arabic are very limited. That is why there are only limited researches done in the area of dialectal Arabic speech recognition. In this paper, we are proposing a cross-lingual acoustic modeling approach for dialectal Arabic, where we can benefit from existing MSA speech resources, in order to improve dialectal Egyptian Arabic recognition rate.

2.1 Speech Corpora

MSA Corpus. The existing MSA speech resource we used is the Nemlar news broadcast speech corpus. It was chosen in training MSA acoustic models [1]. The corpus consists of 33 h of MSA news broadcast speech. The broadcasts were recorded from different radio stations. All files were recorded in linear PCM format, 16 kHz, and 16 bit. The total number of speakers is 259 and the lexicon size is 62K distinct words with a phoneme set of 34 phonemes. This corpus was mainly selected because the transcriptions are fully vowelized and manually reviewed, and hence there were accurate phonetic transcriptions. The Nemlar corpus excluded speech segments with music or noise in the background. Cross-talks and segments with truncated words were excluded as well.

Egyptian Arabic Corpus. In order to adapt MSA and make it Egyptian dialect dependent and hence improve the recognition rate, we have used a previously collected Egyptian corpus [2]; a database of most frequently used words and utterances. The database includes utterances from different speech domains like greetings, time and dates, words spelling, restaurants, train reservation, Egyptian proverbs, etc. The diversity of speech domains ensures good coverage of acoustic features. A lexicon of 700 words was used with accurate phonetic transcription using the dictionaries [3,5]. The total number of speakers is 22 native Egyptian speakers with tri-phones coverage of 15K distinct tri-phones. Every speaker was prompted to read 50 utterances chosen randomly from the database. All recordings were performed in linear PCM, 16 kHz, and 16 bits. The Egyptian corpus was used as a training set to train the Egyptian baseline acoustic model and in adapting existing MSA acoustic model.

Experimental Testing Corpus. Since the main objective is to test if the implemented engine detects the defected phonemes or not, subjects for this experiment were adults and children. This was intended to make sure that the engine works generic on all people suffering from Dyslalia not only children. We had ten subjects, six adults; three males and three females, and four children; three girls and one boy, aged from 7 to 10 years old. Subjects were males and females who responded to a general request for participation in an Automatic Speech Recognition experiment. All subjects were normal people without any speech problem diagnosed. There were no any age or gender restrictions. The reason why the chosen subjects were not suffering from dyslalia, is the difficulty of exactly identifying the wrong phoneme pronunciation since the patient may have a problem in more than one phoneme in the word. Our strategy was to test specific phoneme each time, so that would have given us wrong indication. The tool being used for speech collection is Audacity; a free audio editor and recorder. The recording was in a closed room free from any noise. Subjects used a Microsoft LifeChat LX-3000 microphone while recording for better and clear audio files. Files are recorded in PCM, mono channel, and sampling frequency of 16 kHz.

Table 1. Sample from the subjects' experiment for the Phoneme (س) with all possible Dyslalia replacements

Correct	سماعه	بسكويت
س/s/	/sma?:h/	/bskawi:t/
ش/S/	/Sma?:h/	/bSkawi:t/
ث/T/	/Tma?:h/	/bTkawi:t/
د/d/	/dma?:h/	/bdkawi:t/
خ/x/	/xma?:h/	/bxkawi:t/

Each subject was asked to record specific words that consist of the (س) /s/ and (ر) /r/ phonemes, where the phoneme is placed in the beginning, middle and ending of the word. The subjects were asked to record each of these words several times, but with a replacement of the correct phoneme to its Dyslalia replacements. These were the data that have been used for testing recognition accuracy. Each of the collected recorded words was associated with the actual phonetic transcription and the expected correct phonetic transcription. This allows us to identify whether the word is pronounced correctly, or the Dyslalia replacement in case of wrong pronunciation. An example from the collected testing set is shown in Table 1 where the two words سماعه and بسكويت are pronounced correctly with the /s/ phoneme (/sma?:h/ and /bskawi:t/ respectively), and all common Dyslalia replacements with phonemes /S/, /T/, /d/, and /x/.

2.2 Adaptation and Results

The whole amount of the MSA corpus was used to train the MSA acoustic model with a typical number of tied-states and Gaussians of 3,000 and 8 respectively. The MSA acoustic model has been adapted in order to make it dialect-dependent and hence improve the recognition rate. The MSA acoustic model was adapted using the Egyptian training set along with the normalized transcriptions. CMU Sphinx has been used in this work [8]. Below are the three adaptation techniques that were evaluated. In all the adaptation techniques, we compared word recognition accuracy, phoneme recognition accuracy, and their normalization recognition accuracy. This comparison is calculated using the process of force-alignment which takes an existing transcript and finds out which, among the many pronunciations for the words, or each phoneme in the word occurring in the transcript, are the correct pronunciations. For the phoneme, the output is written into a file with .phsegdir file name extension in sphinx3_align and it contains each phone start and end positions in terms of frames on time scale (100 frames per second) along with the log likelihood acoustic spectral match score. For the whole word, the output is written into a file with .wdsegdir file name extension in sphinx3_align and it contains also the word start and end positions

Table 2. Sample for phoneme confusion matrix confidence score.

Audio-Text	سماعه /sma?:h/	شماعه /Sma?:h/	ثماعه /Tma?:h/	دماعه /dma?:h/	خماعه /xma?:h/
سماعه /sma?:h/	−2269102	−2514996	−2327558	−2453191	−2482886
شماعه /Sma?:h/	−2427326	−2319749	−4686261	−2399874	−2325771
ثماعه /Tma?:h/	−2262885	−2322751	−2222441	−2263326	−2266508
دماعه /dma?:h/	−2310795	−2370847	−2233933	−2231769	−2278456
خماعه /xma?:h/	−2247120	−2301651	−2480661	−2224442	−2110647

in terms of frames on time scale along with large negative acoustic spectral match score. For the normalization, the match score of the targeted phoneme is divided by the match score of the whole word. We have used Confusion Matrix (CM) method for Word, Phoneme and their normalization as well in testing the results. The idea in the three scenarios is that we get the final match score; e.g. the Phoneme CM, for each word tested, we get the score of the phoneme we want to test with its correct audio reference and start to compare it with other known defects of this phoneme. The lowest score in a row is the best match, and hence the replaced phoneme is detected. Table 2 shows a sample of phoneme CM confidence scores where the word is detected as correctly pronounced if the largest log likelihood value is on the diagonal.

Maximum A-Posteriori (MAP) Adaptation. As shown in Tables 3 and 4, the MAP adapted model resulted in 86.2% recognition accuracy in case of (س) /s/ phoneme-based CM acoustic modeling, and 83.1% recognition accuracy in case of (ر) /r/ phoneme-based CM acoustic modeling, which are actually worse than the baseline with 3% and 4% absolute. This result was almost predictable since MAP adaptation requires large data set for adaptation which was not the case in this experiment.

Maximum Likelihood Linear Regression (MLLR) Adaptation. As shown in Tables 3 and 4, MLLR adaptation was found to give better results when adapting all acoustic model parameters: Gaussian means, variances, mixture

Table 3. Recognition accuracy for the different adaptation techniques for (س) /s/ phoneme CM results of the subjects' collected data.

Technique	Phoneme	Word	Normalized
Baseline	89.2%	86.1%	90.4%
MAP	86.2%	82.1%	88.1%
MLLR	92.3%	86.1%	92.3%
MAP + MLLR	93.4%	89.6%	92.4%

Table 4. Recognition accuracy for the different adaptation Techniques for (ر) /r/ Phoneme CM results of the subjects' collected data.

Technique	Phoneme	Word	Normalized
Baseline	87.1%	86.3%	86.2%
MAP	83.1%	82.2%	83.2%
MLLR	91.2%	89.1%	89.4%
MAP + MLLR	95.6%	93.1%	94.2%

weights, and transition weights. In the case of phoneme-based CM acoustic modeling, the adapted MSA model performed recognition accuracy of 92.3% in case of (س) /s/, and 91.2% in case of (ر) /r/ which are actually better than the baseline with 3% and 4% absolute.

Combined MAP and MLLR Adaptation. The combination of MAP and MLLR resulted in the best recognition accuracy As shown in Tables 3 and 4. In the case of phoneme-based CM acoustic modeling, the adapted MSA model performed recognition accuracy of 93.4% in case of (س) /s/ as shown in Table 3, and 95.6% in case of (ر) /r/ as shown in Table 4 which are actually 4% and 8% absolute increase compared to the baseline.

3 "Kalemni Aktar"

"Kalemni Aktar" ("Talk to me more") is our proposed web GWAP that is used mainly to help Dyslalia children improve their Dyslalia. It also works as a tool for therapists to monitor their patients' progress.

3.1 Game Design

"Kalemni Aktar" is a web game application; it is either a one player game or a two player game for the sake of competition. There are three main interfaces; Player, Physician, Admin. The game is divided into three rounds. In the player interface, once the player is logged in and the game starts, he/she chooses a theme to continue the game with by selecting between the following: Zoo, Kitchen, or Around the world. In the 1st round of the game, the player is directed to all the Arabic Phonemes to pick a phoneme out of the Arabic 28 phonemes. A round of random three words with the selected phoneme on the beginning, middle, and end appear to the player in bubbles. The player must click on the 3 bubbles but in any order he/she wants, to explode with an image, text, and a stored audio of the target word. For each round, there is a timer of 180 s set. This is done to increase the challenge for the players. Within this interval of time, it is possible to have 5 trials for each word. For each trial, the system detects if the

Fig. 1. *Kalemni Aktar* ("Talk to me more") GWAP graphical user interface - wrong pronunciation. (Color figure online)

phoneme in the word is pronounced correctly or not. If it is correct, the system automatically moves to the next word in the round, else the system recognizes the said word as a feedback to the player to help him in the other 4 trials.

In Fig. 1, the player is playing in the first round where the selected phoneme is (س) /s/. The phoneme is in the beginning of the word. The player should have said سمكه /smkh/ (word appearing in green color), but the player said ثمكه /Tmkh/ (word appearing in red color) instead. The player still has more 4 trials.

In Fig. 2, the phoneme is in the middle of the word. The player said أسد /?asd/ correctly from the 2nd trial, so she got 8 points, and the bubble of the 3rd word appears to the player to move further.

Since one of the incentives in GWAPs that engage and motivate players is the score [9]. The score for each round is calculated as follows:

- If the word is pronounced correctly from the first trial, the player gets 10 Points.
- If the word is pronounced correctly from the second trial, the player gets only 8 points.
- If the word is pronounced correctly from the third trial, the player gets only 6 points.
- If the word is pronounced correctly from the fourth trial, the player gets only 4 points.
- If the word is pronounced correctly from the fifth trial, the player gets only 2 points.
- Finally, if after the 5 trials the player still pronounces the word incorrect, he/she does not achieve any point.

Fig. 2. *Kalemni Aktar* ("Talk to me more") GWAP graphical user interface - correct pronunciation.

After each round, a feedback appears to the player with the achieved score. This is repeated till all phonemes are pronounced. At the end of this level, a summarized report appears to the player with the feedback of the detected phoneme problems of all phoneme rounds. The 2nd round is the automatic system adaptation to the performance in the 1st level. It consists of some exercises with random words on those detected phoneme issues. The 3rd round is a tongue twister, a proficiency level stressing on certain phoneme which evaluates several words at the same time. In the physician Interface, speech therapists have access to their patients' profiles to monitor the progress of the cases they have; so that they can assist in the sessions. For the Admin Interface, it is a basic interface for managing game content.

By going through the game, all proven important game design factors from attractiveness, curiosity, immediate and accurate feedback, the issue of control, challenge feeling, and automatic system adaptation to user performance were covered [12].

3.2 Subjects and Experiment

Subjects for this game were twenty children; ten boys, and ten girls aged from seven years old to ten years old. They are patients in Ain Shams Specialized Hospital. The subjects were chosen suffering from same level of Dyslalia specifically from the phonemes (س) /s/ and (ر) /r/. The twenty children were not able to pronounce the targeted phoneme correctly.

Subjects were divided into two groups; each group consists of five boys, and five girls. The 1st group was having basic normal sessions with the speech therapist. The 2nd group was introduced to the implemented game in the session.

The sessions were held for thirty to sixty minutes twice a week. The duration of the experiment was two months. The experiment had two main targets. The 1st target was to detect the accuracy of the game to see whether it could be really relied on for detecting the defected phonemes with its wrong replacement or not. The 2nd target was to compare the progress of the child speech when playing the game than when having normal sessions with the speech therapist.

Throughout the two months experiments, the subjects of the two groups were monitored and interviewed about their feedback, level of interest and motivation.

3.3 Experimental Results

All subjects were introduced to a new speech therapist, who hasn't been involved earlier in the experiment, to test the children's level of dyslalia in the targeted phonemes after the two months therapy. This was to make sure that results will be double blinded. After gathering all test results for the two groups, results showed that both groups were been able to develop the (س) /s/ and (ر) /r/ phonemes better compared to the beginning of the experiment; however the 2nd group determine faster progress in speech than the 1st group does.

For the phoneme (س) /s/, nine out of the ten children of the second group were able to pronounce the (س) /s/ correctly, however one child still pronounces (ث) /T/ instead. In comparison to the first group where only seven out of the ten children were able to pronounce it correctly as shown in Fig. 3. For the phoneme (ر) /r/, nine out of the ten children of the second group were able to pronounce the (ر) /r/ correctly, however one child pronounced (ي) /?:/ instead. In comparison to the first group where only six out of the ten were able to pronounce it correctly as shown in Fig. 4.

The children in the second group reported very good feedback, they were interested in using the application and switching between different themes, and

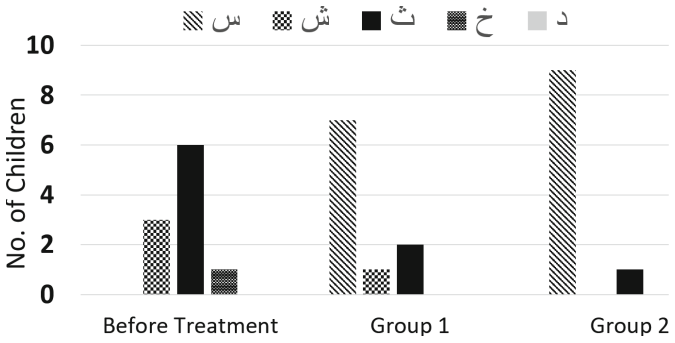


Fig. 3. The progress after two months for phoneme (س) /s/

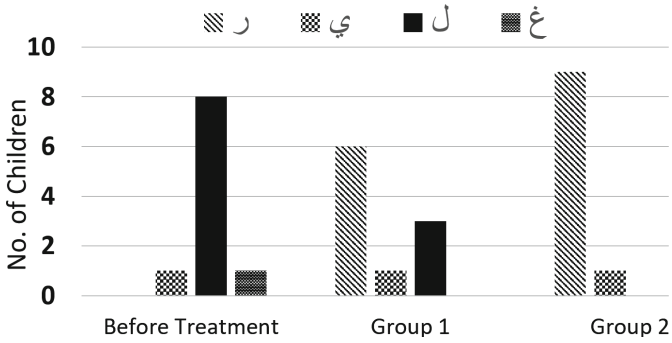


Fig. 4. The progress after two months for phoneme (ر) /r/

phonemes. They found the interfaces friendly with suitable colors, and the characters of the game attracted their attention. Some mentioned that they were very curious to explore the different levels, and to achieve the highest score possible. While the children in the 1st group reported quite negative feedback compared to the others, some mentioned that they got bored in the therapy, some wanted to return home, and some got unmotivated throughout the session.

The use of such computer-based methods during various phases of the speech therapy determines a new psychological and pedagogical situation by creating a special interesting learning environment, and by facilitating a new, superior method for correcting and developing speech.

4 Conclusion and Future Work

We have presented a speech recognition based system for Dyslalia children called Kalemni Aktar. The aim is to provide assistance to Dyslalia children to improve their speech. Results showed that Kalemni Aktar reached its goal of providing a suitable and useful environment for the Dyslalia child to develop his/her speech. It has been shown that MAP + MLLR combined adaptation technique has best recognition results with accuracy reached 93.4% for (س) /s/ phoneme and 95.6% for (ر) /r/ phoneme.

"Kalemni Aktar" turned out to be a real help in the therapeutic activity, by providing various exercises that children can do both in the sessions and at home. Computer games are a powerful tool for motivating children to practice speech motor skills. The existence of computer based methods cannot replace the therapist's role, but it only helps them in the therapy and helps the children having more exercises at home to develop their speech. The experiments showed high attention and concentration levels of children who practiced, as well as improvement in performance in terms of the game scores. The visual environments used in the prototype game proved to be easy for children to relate to, however more variety is needed to sustain curiosity.

For future work, we recommend upgrading the application by including all other Dyslalia types. The game may also include detection of other speech problems as stuttering to be fully integrated software for all speech disorders. It is also important to evaluate the application by involving more Dyslalia children in the testing phase, this will help to adapt the application according to their assessed needs. This Game can be of great importance to some nonprofit organizations in the sake of improving the society.

References

1. Yaseen, M., Attia, M., Maegaard, B., Choukri, K., Paulsson, N., Haamid, S., Krauwer, S., Bendahman, C., Fersøe, H., Rashwan, M., Haddad, B.: Building annotated written and spoken Arabic LR's in NEMLAR project. In: Proceedings of LREC, pp. 533–538 (2006)
2. Elmahdy, M., Gruhn, R., Minker, W., Abdennadher, S.: Cross-lingual acoustic modeling for dialectal Arabic speech recognition. In: Proceedings of INTERSPEECH, pp. 873–876 (2010)
3. Stevens, V., Salib, M.: A Pocket Dictionary of the Spoken Arabic of Cairo. The American University in Cairo Press, Second printing (2005)
4. Cagatay, M., Ege, P., Tokdemir, G., Cagiltay, N.E.: A serious game for speech disorder children therapy. In: Proceedings of the 7th IEEE International Symposium on Health Informatics and Bioinformatics (HIBIT), pp. 18–23 (2012)
5. Hinds, M., Badawi, E.: A Dictionary of Egyptian Arabic. Librairie du Liban (2009)
6. Koster, R.: Theory of Fun for Game Design. O'Reilly Media Inc., Sebastopol (2013)
7. de Carvalho Souza, A.M., dos Santos, S.R.: Handcopter game: a video-tracking based serious game for the treatment of patients suffering from body paralysis caused by a stroke. In: Proceedings of the 14th IEEE Symposium on In Virtual and Augmented Reality (SVR), pp. 201–209 (2012)
8. Elmahdy, M., Gruhn, R., Minker, W.: Novel Techniques for Dialectal Arabic Speech Recognition. Springer, New York (2012). https://doi.org/10.1007/978-1-4614-1906-8
9. Von Ahn, L.: Games with a purpose. Computer **39**(6), 92–94 (2006)
10. Schipor, O.A., Pentiuc, S.G., Schipor, M.D.: Improving computer based speech therapy using a fuzzy expert system. Comput. Inform. **29**(2), 303–318 (2012)
11. Danubianu, M., Pentiuc, S.G., Andrei, O., Marian, S., Ioan, N., Doina, U., Schipor, M.: TERAPERS-intelligent solution for personalized therapy of speech disorders (2009)
12. Umanski, D., Kosters, W.A., Verbeek, F.J., Schiller, N.O.: Integrating computer games in speech therapy for children who stutter. In: Proceedings of First Workshop Child, Computer and Interaction (WOCCI), pp. 17–21 (2008)
13. Kotby, N., Barakah, M.: Patterns of dyslalia in Egypt. Folia Phoniatrica et Logopaedica **31**(2), 125–128 (1979)
14. Murray, T.G., Parker, V.: Integration of computer-based technology into speech-language therapy. Education. Tech. **31**, 53–59 (2004)
15. Tobolcea, I., Danubianu, M.: Computer-based programs in speech therapy of Dyslalia and Dyslexia-Dysgraphia. Broad Res. Artif. Intell. Neurosci. (BRAIN) **1**(2), 52–63 (2010)
16. Wells, J.C.: SAMPA for Arabic, www.phon.ucl.ac.uk/home/sampa/arabic.htm. Accessed 01 May 2017

Building a Rich Arabic Speech and Language Corpus Based on the Holy Quran

Ali Meftah[1], Yasser Seddiq[1,2(✉)], Yousef Alotaibi[1], and Sid-Ahmed Selouani[3]

[1] College of Computer and Information Sciences, King Saud University, Riyadh, Saudi Arabia
{ameftah, yaalotaibi}@ksu.edu.sa
[2] King Abdulaziz City for Science and Technology (KACST), Riyadh, Saudi Arabia
yseddiq@kacst.edu.sa
[3] LARIHS Lab, Université de Moncton, Campus de Shippagan, Shippagan, Canada
selouani@umcs.ca

Abstract. This paper pursues the goal of creating a reliable speech corpus based on The Holy Quran (THQ) audio recordings. Achieving that goal involves major steps to be done and essential requirements to be considered. With the availability of tremendous amount of recordings nowadays, it is of a fundamental importance to select the ones that feature both high audio quality and perfect reciter performance. Also, since the targeted beneficiaries from the corpus are the digital speech processing research community, it is also very essential to maintain an efficient, a familiar and a convenient way of presenting the audio corpus and other language material, such as the language model. Audio recordings of THQ are selected from four sources having a high standard regarding the reciters' performance. A significant effort is made in phonetical transcription of the audio content such that the written transcript maps perfectly to the uttered phonemes. Furthermore, the corpus dictionary, which is usually required in many fields such as machine learning and datamining, is also created. The first release of the corpus consists of recorded recitations and the necessary metadata of three chapters of THQ of different lengths recited by four reference reciters. Those chapters are selected for this phase based on statistical analysis of the lengths of all chapters and the frequency of occurrence of the Arabic phonemes across all chapters of THQ.

Keywords: The Holy Quran · Speech corpus · Arabic speech processing

1 Introduction

Speech corpora form the solid foundation for any research on data mining and/or speech processing. Speech corpora are language specific and researchers who target a certain language should consider selecting the proper corpus of that language very seriously. They should also give high priority to investigating the available corpora for that language in order to assess accessibility, richness, correctness, and quality of those corpora. Creating new corpora and enhancing the existing ones are both valuable

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 90–101, 2018.
https://doi.org/10.1007/978-3-319-73500-9_7

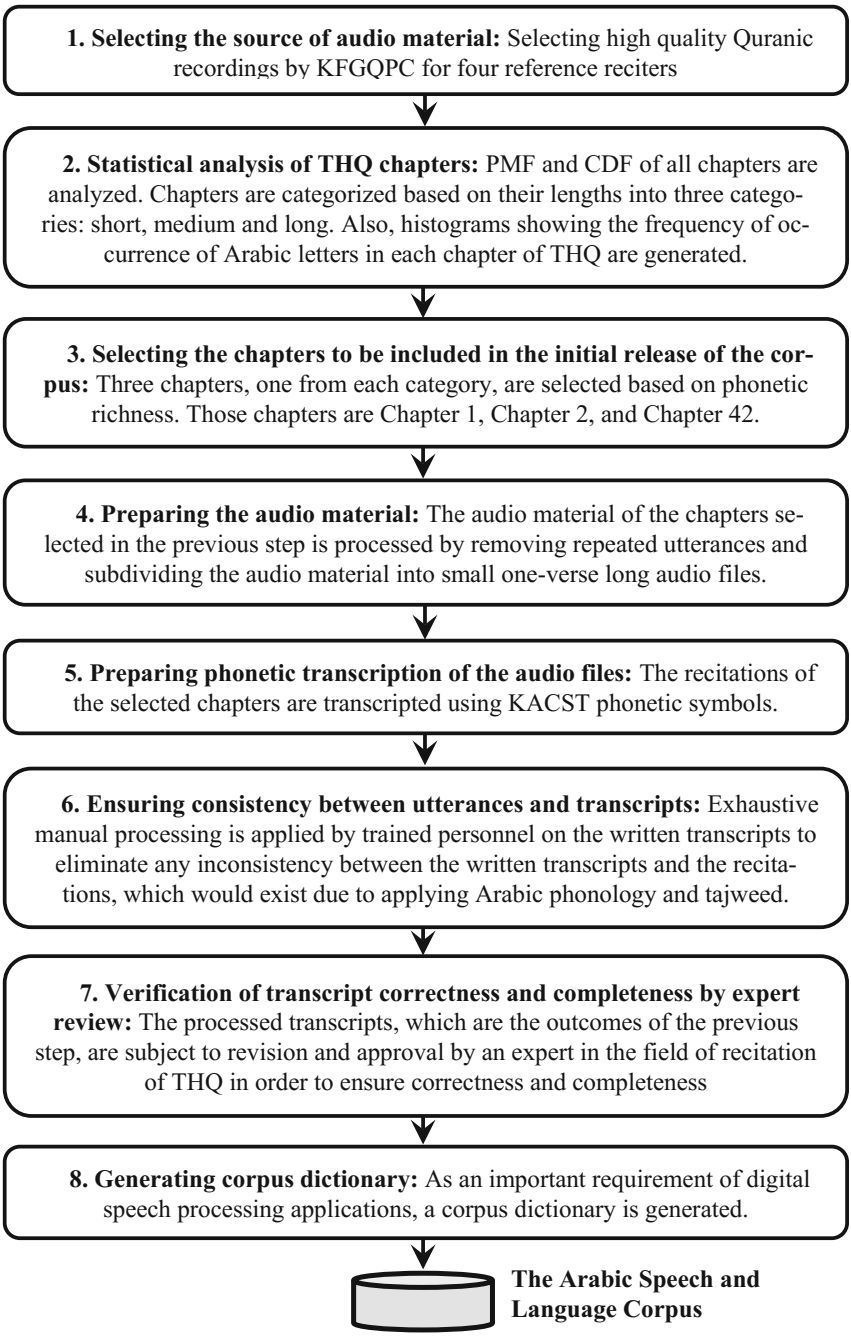


Fig. 1. The process of creating the Arabic Speech and Language Corpus.

contributions that researchers would make to their languages. Building a spoken corpus requires a good audio material, and an efficient approach that allows users to extract a comprehensive language model. The content of a good corpus should also be representative of the phonology and phonetics of a given language. Numerous Arabic speech corpora are available for the research community. For instance, King Abdulaziz City for Science and Technology (KACST) Arabic Phonetic Database [1], The Saudi Accented Arabic Voice Bank [2], The BBN/AUB Corpus of the Levantine dialect [3], and the West Point Corpus of native and non-native speakers [4], just to name a few. However, these corpora were designed for very narrow and specific application. Therefore, the research community in Arabic speech processing field is still looking forward to having access to more comprehensive corpora that would enable researchers to conduct exhaustive studies dedicated to Arabic language.

This work aims at contributing to the enrichment of the Arabic linguistic resources that could be used in various fields of Arabic speech and language processing. This paper presents an Arabic speech corpus based on the recorded recitations of The Holy Quran (THQ) and describes the process and the criteria that we follow to select the most suitable recitations amongst the tremendous amount of recordings that are publicly available nowadays. The process of creating the corpus is illustrated in the chart in Fig. 1. The paper presents detailed description of those steps. The first stage of the corpus creation consists of providing a representative subset of THQ audio and language model and other resources contents. That subset is selected on a statistical basis to ensure audio material adequacy. The outcomes of this first stage of the project are reported.

2 Creating an Arabic Speech Corpus Based on THQ

All audio recordings that we have access to are to be qualified for suitability for the corpus. Two quality criteria are considered: reciter performance and precise written scripts. Therefore, choices are made from the THQ recordings produced by the King Fahd Glorious Quran Printing Complex (KFGQPC) [5], which is an official Saudi government authority responsible for producing authenticated prints and recordings of THQ. Not only those recordings are made under controlled environment to maintain highest acoustic quality, but also the reciters are well selected to ensure correctness of pronunciation performance. Having access to such material is of great value in paving the way for the subsequent activities towards creating the corpus. Four sources from KFGQPC by considering the following reciters: Abdullah Ali Basfar (R01), Mohammed Ayub (R02), Ali Alhuthaifi (R03), and Ibrahim Alakhdhar (R04) are selected in addition to text form reflecting the pronounced text. This audio material is of high quality and deemed very suitable to serve the purpose of creating a corpus.

The four sources are analyzed by collecting statistics about the frequency of occurrence of the Arabic letters. Each audio source should match the histogram illustrated in Fig. 2 that is based on a written script of THQ. Arabic letters and phonemes are transcribed using KACST symbols [6] henceforth as listed in Table 1.

While the corpus ultimatly targets all content of THQ, at this stage, only a part of THQ is covered. Since chapters are not equal in length, we chose to include sample chapters of various lengths for the initial release of the corpus. The distribution of THQ

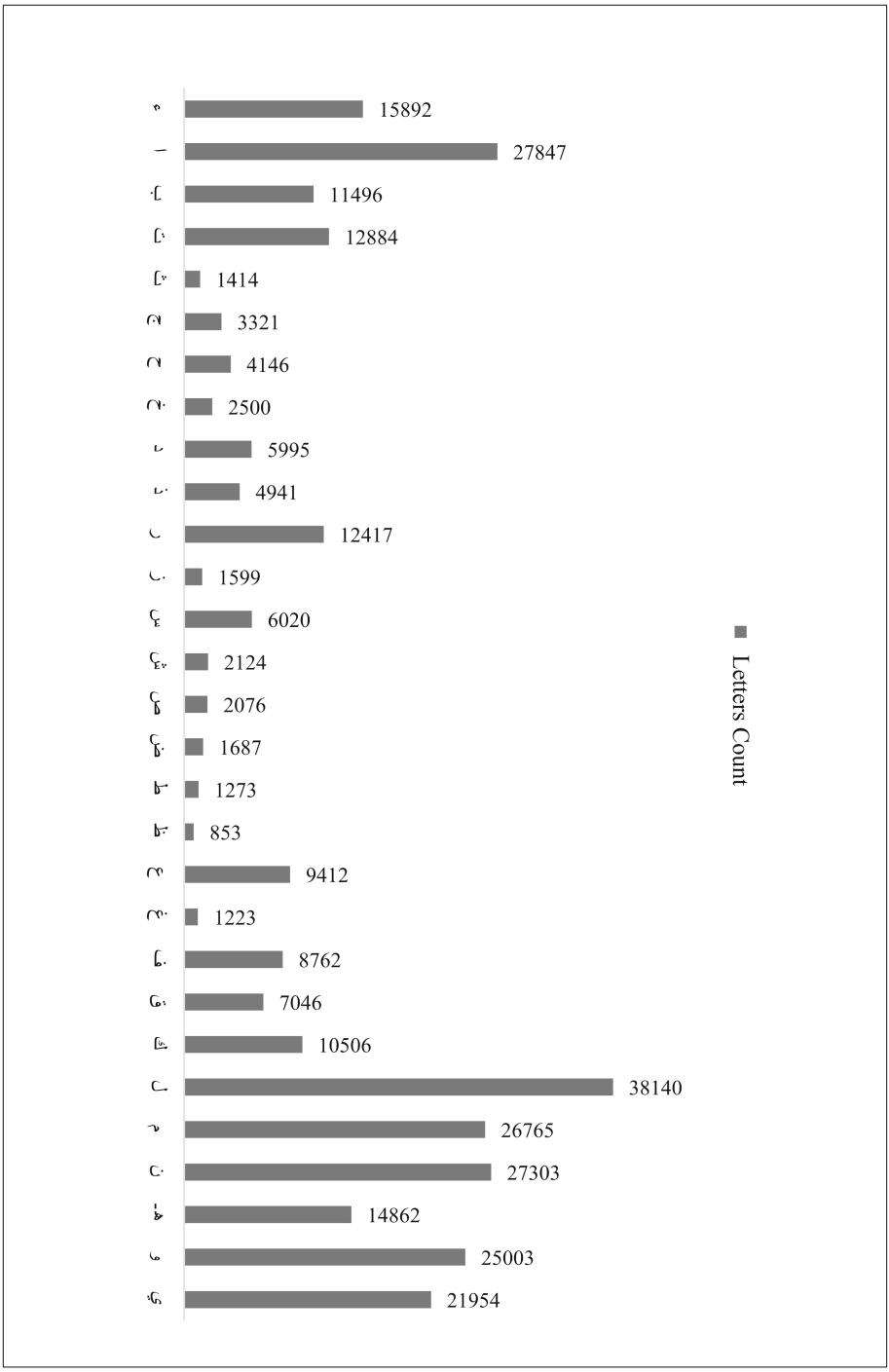
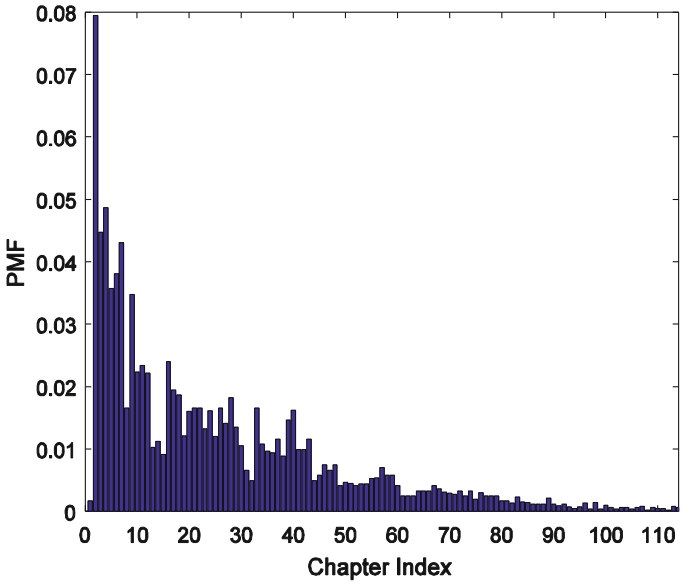


Fig. 2. Statistics of the Arabic letters (written) across all chapters of THQ

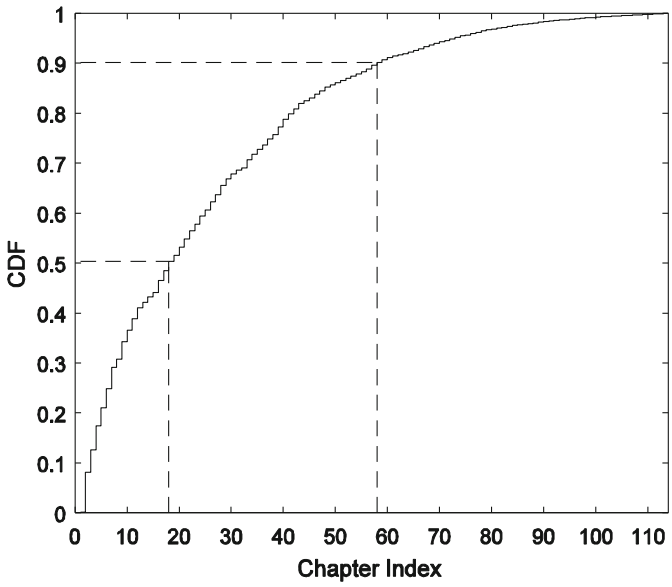
Table 1. KACST phonetic symbols

No.	Arabic Writing	KACST Symbols	IPA Symbols
1	ء	hz10	ʔ
2	ب	bs10	b
3	ت	ts10	t
4	ث	vs10	θ
5	ج	jb10	ʤ
6	ح	hb10	ħ
7	خ	xs10	χ
8	د	ds10	d
9	ذ	vb10	ð
10	ر	rs10	r
11	ز	zs10	z
12	س	ss10	s
13	ش	js10	ʃ
14	ص	sb10	$s^{?}$
15	ض	db10	$d^{?}$
16	ط	tb10	$t^{?}$
17	ظ	zb10	$ð^{?}$
18	ع	cs10	ʕ
19	غ	gs10	ɣ
20	ف	fs10	f
21	ق	qs10	q
22	ك	ks10	k
23	ل	ls10	l
24	لا	lb10	l̪
25	م	ms10	m
26	ن	ns10	n
27	ه	hs10	h
28	و	ws10	w
29	ي	ys10	j
30	ـَ	as10	a
31	ـُ	us10	u
32	ـِ	is10	i
33	ـا	as20	aa
34	ـو	us20	uu
35	ـي	is20	ii

content over all 114 chapters is analyzed by means of the probability mass function (PMF) given in Fig. 3(a) and the cumulative distribution function (CDF) given in Fig. 3(b). According to the CDF, half of THQ content covers only 18 chapters (16% of the chapters). We consider those as long chapters. It is worth mentioning that while this



(a)



(b)

Fig. 3. THQ content with respect to the 114 chapters illustrated using (a) PMF and (b) CDF.

group consists of long chapters, there is an exception of Chapter 1 that is deemed as one of the shortest chapters of THQ. Thus, this chapter will be treated as a member of the group of short chapters to be clarified next. A second group consisting of the last 56 chapters plus Chapter 1 (50% of the chapters) contributes to only 10% of THQ content implying that those are short chapters. In between, there are Chapters 19 to 58 (40 chapters forming 35% of the total chapters) contribute to 40% of THQ, which indicates a medium-length chapters. We decide to select one chapter from each group (long, medium and short) for the initial release of the THQ corpus that we are creating.

From Fig. 2, the least frequent letter in THQ is zb10 (ظ) that occurs 853 times followed by gs10 (غ) and tb10 (ط) that occur 1223 and 1273 times, respectively. Thus, we pay high attention to those least occurring letters when selecting the chapter that we should start with. The occurrence of those letters across the 25 chapters in focus is investigated and summarized in Fig. 4. The selection is made on a chapter that has balanced yet high frequency of occurrence of those three letter, which is Chapter 42 (Alshoura). Beside this chapter, the longest chapter (Chapter 2: Albaqarah) and one of the short chapters (Chapter 1: Alfatehah) are considered.

After selecting the audio material chapters, they are partitioned into reasonably short audio files. Each audio file should contain one complete verse. The maximum length of an audio file is chosen to be the period needed to recite three lines of the written script of THQ based on the KFGQPC print. However, there are verses that are long and some of them could span one page. Such verses are further partitioned such that each part does not exceed the specified maximum length.

The audio files are also processed to ensure consistency of content produced by each reciter. That is, because reciters are allowed to repeat some parts of text whenever appropriate, those repetitions result in inconsistence phoneme histograms across the four sources. Therefore, audio material is traced for repeated text that is eliminated whenever found.

The audio file names are assigned according to this following code: *D06N01SxxxAxxxASxRxxTxx*, where the name is decoded as follows:

- ***D06***: indicates the corpus serial number.
- ***N01***: indicates that the current recitations follow the narration of Assem AlKoofi. This is one of ten different narrations of THQ named after the scholars Assem AlKoofi, Ibn Katheer almakki, Nafea AlMadni, Abu Jaafer AlMadni, Abu Amro AlBassry, Hamzah AlKoofi, Ibn Amer AlShami, AlKessaei AlKoofi, Yaqoob AlBassry, and Khalaf bin Hesham [7].
- ***Sxxx***: is the chapter number.
- ***Axxx:*** is the verse number.
- ***ASx:*** indicates the partition number of a verse in case if it is partitioned due to exceeding maximum length as explained earlier. In case of a one complete verse, this code is set to AS0.
- ***Rxx:*** is the reciter index, where x ranges between 1 and 4.
- ***Txx:*** is the trial number.

For example filename D06N1S042A016AS0R02T01 indicates that the Ayaa (phrase) number 16 from the chapter 42 (Alshoura) read it by the reciter number 2 (Mohammad Ayyub) in trial 1 used the narrator Hafss.

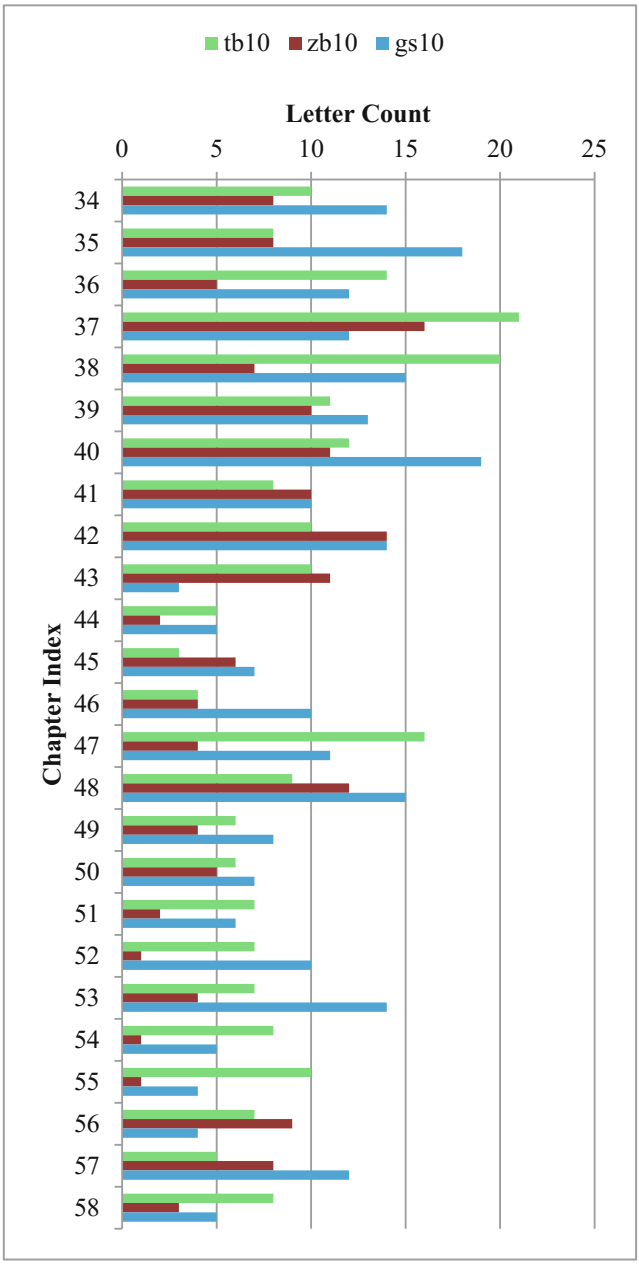


Fig. 4. Histogram of the three least frequent letters in THQ

3 Corpus Metadata

The corpus must contain text-format metadata describing the transcription of the audio material. In this section, we describe the work of preparing the metadata of the corpus. The process starts by using an electronic copy the script of THQ in text format that is published by KFGQPC that is written according to the Othmanic orthography. Indeed, in Arabic, spoken text does not map perfectly to the corresponding written text since there are letters that appear in written text but are not uttered and vice versa. In the case of THQ, there are also the *tajweed* rules that also inforce reciters to alter some written letters. That inconsistency between what is written and what is spoken must be addressed in the corpus because it primary targets speech processing. Therefore, the text metadata of the corpus must adhere perfectly to the uttered speech. It is very important to emphasis on the fact that this way of writing Arabic text is not correct from rules of writing perspective, but in the case of speech corpora this is acceptable because this transcription will be exclusively read by computers not by humans. Moreover, that transcription is written in phonetic alphabets not in ordinary alphabets. In this work, we use the KACST phonetic symbols that are illustrated in Table 1.

A famous inconsistency in Arabic speech is the effect when uttering /hz10 ls10/(ال), which means "the", followed by one of the Solar Letters (also called Sun Letters) [8]. In such case the phoneme /ls10/(ل) is not uttered. For instance, the word (و السماء) meaning "and the sky" is transcripted without the /ls10/(ل) as follows: /ws10 as10 ss20 ms10 as20 hz10 as10/in KACST symbols, which is equivalent to /wassama:ʔ/in IPA. Another case is converting a written (ب) to an uttered /ms10/(م) whenever the former is preceded by (ن). This effect is called *Eqlaab* in Tajweed terminology. Beside the aforementioned two effects, there are many other effects in Arabic and Tajweed such as Tanween, Ghunnah, Edgham and Ekhfaa. All these effects are considered in the process of text transcription of the current corpus.

The transcription process described above is done by a qualified personnel. All the subsequent stages and material are based on the outcomes of this stage. Hence, error-free transcripts must be delivered by end of this fundamental stage. The correctness of the text transcripts is assured by passing an expert review. Finally, the text-format transcripts are read to be added to the corpus material.

4 Corpus Dictionary

An important part of the corpus metadata is the dictionary of the corpus, which is a lookup table listing all unique vocabulary used in the corpus. Each word in the dictionary is transcribed using phonetic symbols and presented in format that can be recognized by ASR systems. In this corpus, the dictionary data is organized in three columns: the original word written in Arabic, the word transcribed using English alphabets, and phonetic transcription of the word using KACST phonetic symbols. A sample of the dictionary is illustrated in Table 2.

Table 2. Sample entries of THQ corpus dictionary

THQ Words		
English	Arabic	Pronunciation
yajtabiii	يَجْتَبِيّ	ys10 as10 jb10 ts10 as10 bs10 is61 sp
yajtanibuwna	يَجْتَنِبُونَ	ys10 as10 jb10 ts10 as10 ns10 is10 bs10 us21 ns10 as10 sp
yajmau	يَجْمَعُ	ys10 as10 jb10 ms10 as10 cs10 us10 sp
yaxtim	يَخْتِمْ	ys10 as10 xs10 ts10 is10 ms10 sp
yaxluqu	يَخْلُقُ	ys10 as10 xs10 ls10 us10 qs10 us10 sp
yashaA	يَشَآ	ys10 as10 js10 as10 hz10 sp
yashaAi	يَشَآءِ	ys10 as10 js10 as10 hz10 is10 sp
yashaaaAu	يَشَآءُ	ys10 as10 js10 as61 hz10 us10 sp
yashaaaAuwna	يَشَآءُونَ	ys10 as10 js10 as61 hz10 us21 ns10 as10 sp
yaGfiruwna	يَغْفِرُونَ	ys10 as10 gs10 fs10 is10 rs10 us21 ns10 as10 sp

The outcome of this work is the first release of the THQ corpus. It contains 346 phrases, with a total of 7,033 spoken words and 44,359 phonemes. Chapter 1 includes seven phrases that contain 29 spoken words, Chapter 42 contains 53 phrases with 860 spoken words, and Chapter 2, which is the longest chapter of THQ, contains 286 phrases containing 6,144 spoken words.

5 Conclusions and Perspective

A phonetically rich Arabic speech corpus was created. Its content is based on the audio material of THQ. Recitations of four reputed reciters were carefully selected. The first release of the corpus consists of a representative subset of THQ audio content and other language resources. Namely, Chapters 1, 2 and 42 were selected for this phase on the basis of a statistical analysis of the CDF of the lengths of all chapters and the occurrence frequency of the Arabic phonemes across all chapters of THQ.

A significant effort was made to achieve perfect phonetical transcription eliminating all inconsistencies between the written script and the uttered phonemes. The existence of such inconsistency is normal due to reciters adherence to Arabic phonology and the tajweed rules. A very important step towards achieving our goal was ensuring that the corpus transcription material passed expert revision before confirming its correctness and validity for digital speech processing applications. Moreover, in order to make the corpus useful for machine learning, data mining applications and ASR, a corpus dictionary was also created. The dictionary entries consist of all unique words in the covered material.

The outcome of this work is a speech corpus consisting of 7,033 spoken words equivalent to 44,359 phonemes. Chapter 1 contains 29 spoken words, Chapter 42 contains 860 spoken words, and Chapter 2, which is the longest chapter of THQ, contains 6,144 spoken words.

This work emphasizes on the process that was applied in creating the current corpus. The steps illustrated in Fig. 1 are of great importance to ensure the soundness and completeness of speech corpora that are based on THQ.

The ultimate goal of this project is to include all chapters of THQ in the corpus. The future work will involve the rest of THQ in a similar procedure that is applied in this phase before introducing the complete corpus.

Acknowledgment. This project was funded by the National Plan for Science, Technology and Innovation (MAARIFAH), King Abdulaziz City for Science and Technology, Kingdom of Saudi Arabia, Award Number (11-INF1968-02).

References

1. Alghamdi, M.: KACST Arabic phonetic database. In: The 15th International Congress of Phonetics Science; 3–9 August 2003; Barcelona, Spain, pp. 3109–3112. Universitat Autònoma de Barcelona, Barcelona (2003)
2. Alghamdi, M., Alhargan, F., Alkanhal, M., Alkhairy, A., Eldesouki, M., Alenazi, A.: Saudi accented Arabic voice bank. J. King Saud Univ. **20**, 45–64 (2008)

3. BBN Technologies (with American University of Beirut a subcontractor), et al.: BBN/AUB DARPA Babylon Levantine Arabic Speech and Transcripts LDC2005S08, 1st edn. Linguistic Data Consortium, Philadelphia (2005)
4. LaRocca, S., Chouairi, R.: West Point Arabic Speech LDC2002S02, 1st edn. Linguistic Data Consortium, Philadelphia (2002)
5. King Fahd Glorious Quran Printing Complex. http://www.qurancomplex.org/
6. Alghamdi, M., Mohamed El Hadj, Y.O., Alkanhal, M.: A manual system to segment and transcribe Arabic speech. In: IEEE International Conference on Signal Processing and Communications (ICSPC), Dubai, United Arab Emirates, pp. 233–236 (2007)
7. AlQahtany, M.O., Alotaibi, Y.A., Selouani, S.-A.: Analyzing the seventh vowel of classical Arabic. In: 2009 International Conference on Natural Language Processing and Knowledge Engineering, NLP-KE 2009, pp. 1–7 (2009)
8. Definition of sun letter, Merriam-Webster Dictionary. https://www.merriam-webster.com/dictionary/sun%20letter. Accessed 24 May 2017

Text Categorization, Clustering and Summarization

Combining Words and Concepts for Automatic Arabic Text Classification

Alaa Alahmadi, Arash Joorabchi, and Abdulhussain E. Mahdi(✉)

Electronic and Computer Engineering Department, University of Limerick, Limerick, Ireland
{alaa.alahmadi,arash.joorabchi,hussain.mahdi}@ul.ie

Abstract. The paper examines combining words and concepts for text representation for Arabic Automatic Text Classification (ATC) and its impact on the accuracy of the classification, when used with various stemming methods and classifiers. An experimental Arabic ATC system was developed and the effects of its main components on the classification accuracy are assessed. Firstly, variants of the standard Bag-of-Words model with different stemming methods are examined and compared. Arabic Wikipedia and WordNet were examined and compared for providing concepts for effective Bag-of-Concepts representation. Based on this, Wikipedia was then utilized to provide concepts, and different strategies for combining words and concepts, including two new in-house developed approaches, were examined for effective Arabic text representation in terms of their impact on the overall classification accuracy. Our experimental results show that text representation is a key element in the performance of Arabic ATC, and combining words and concepts to represent Arabic text enhances the classification accuracy as compared to using words or concepts alone.

Keywords: Arabic text classification · Text representation models · Bag of words
Bag of concepts · Wikipedia · WordNet

1 Introduction

Automatic Text Classification (ATC) is an essential process for efficient organization of digital text. With the rapid growth of Arabic digital text, ATC has become one of the important tasks in Arabic text mining. The goal of ATC is to assign one or more predefined categories to a given textual document. The process involves three main components: text pre-processing, text representation and the classifier which is built using a generic Machine Learning (ML) algorithm. The classification begins by pre-processing the textual content of all the documents in the dataset in order to extract a set of well-defined features. These features are then passed to the text representation component where each document is represented as a set of features in a Vector Space Model (VSM) [1]. A document is often seen as a set of feature points in space of features, and different representation strategies are used to place these features in a VSM model. Finally, the VSM is fed to an ML based classification algorithm.

At its basic level, a text representation model expresses a piece of text or document in a compact representation of its textual content. Text representation models are

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 105–119, 2018.
https://doi.org/10.1007/978-3-319-73500-9_8

commonly built using words as features, where text in a document is represented by the words it is composed of and the document is classified to a category based on the proportion of words that it has in common with other documents from the same category. In this case, the resulting representation is known as Bag-of-Words (BOW) model [2]. In addition to its simplicity, the BOW has proven its effectiveness particularly in English text classification [3], as well as many other languages. However, despite its efficiency, the BOW model has a number of limitations:

- The BOW model treats synonymous words as independent features. For example, "classification" and "categorization" are considered as two independent words with no semantic association. As a result, documents that discuss similar topics and contain synonymous words could be considered as unrelated.
- Words can have different meanings depending on their surrounding context, i.e., polysemy. The BOW model cannot recognize the meaning of a single word in different contexts even if the meaning is totally different. Take the word "bright" for example, depending on its context "bright" could mean shining or clever. Using the BOW as a representation model the word "bright" would be treated as a single feature irrespective of its intended meaning in different contexts.
- The BOW model representation breaks terms into their constituent words, e.g., it breaks "text classification" into the words "text" and "classification". As a result, the order of the words is lost and the unique meanings of the terms disappear. In addition, the BOW model tends to destruct the semantic relations between words and terms as it treats them as stand-alone units with. The semantic relation does not cover only synonymy and polysemy, it also reflects the relationship between words. For example, "text classification" is related to "text mining".

To address above limitations, a feature known as a concept has been introduced in text mining, giving raise to the Bag of Concepts (BOC) text representation model [4]. A concept is a unit of knowledge which provides a unique meaning. Synonymous words are mapped to the same concept which provides that unique meaning they share. Words with multiple meanings are mapped to different concepts based on the surrounding text. In order to use the BOC representation model in an ATC system, a knowledge base such as WordNet, Open Directory Project (ODP), or Wikipedia is needed to provide concepts. In recent years, concepts have been used to represent text for English ATC [5–11]. However, using concepts alone to represent text does not result in a significant classification improvement as confirmed by [4]. Therefore, a number of researchers have experimented with combining words and concepts to represent text based mainly on the following strategies:

- Adding Concepts (AC): this strategy involves forming a combined text representation model by adding concepts identified in the document to its BOW model as extra features using different knowledge bases. In particular, AC has been used for English ATC with Wikipedia [6, 9–11], WordNet [12], and ODP [5]. Furthermore, it has also been proposed for text clustering with WordNet [4] and Wikipedia [13].
- Replacing Terms with Concepts (RTC): this strategy is similar to AC, but words and terms in the document for which a concept has been identified are removed from the combined text representation. Only words which do not have corresponding concepts

are added to the text representation model. The strategy was first proposed for English text clustering by Hotho et al. [14], who also showed that this strategy yielded less accurate clustering results compared to AC.

- Adding Concepts and Categories (ACC): in this strategy, which was proposed by Wang et al. [10, 11], the parent categories of the concepts are added to the combined representation model along with concepts and words.

Since its introduction, the AC strategy has been used by a number of researchers who demonstrated its improving impact on the performance of English ATC. For example, Gabrilovich et al. [5] used ODP as a knowledge base to provide concepts for text representation. They used a feature generating technique which searches for new features that describe the target document better than the ones contained in the training documents. The feature generator constructs new features from the ODP categories and adds them to the BOW model using AC strategy. Experimental results showed improved classification performance in comparison with the BOW model. However, the ODP has a number of drawbacks. Its categories are not equally covered, some categories are repeated in different branches of the categories hierarchy tree, and sometimes some are more influenced by the views of the editors in charge. Gabrilovich et al. [6] subsequently showed that using Wikipedia as a knowledge base instead of the ODP improved their classification results further.

Wang et al. [10] used Wikipedia synonyms, associated concepts and hyponyms (parent categories for the concept), by adding them to the BOW model. The study showed that adding synonyms to the BOW is not useful for ATC, whereas adding the top 5, 10, 15, 20, and 25 associated concepts improved the classification accuracy. In addition, the authors compared enriching the document representation with hyponyms of candidate concepts extracted from the first five levels of their relational hierarchy as provided by Wikipedia. Their results showed that adding hyponyms extracted from first three levels achieved better classification accuracy compared to adding hyponyms extracted from levels 1 to 5 of the hierarchy.

In this paper, we examine combining words and concepts for text representation for Arabic ATC and how this impacts the accuracy of the classification when used with various stemming methods and classifiers, compared to using words or concepts alone. To achieve this, an experimental Arabic ATC system has been developed and the effect of each main component on the classification accuracy is assessed. First, variations of the BOW model resulting from the application of different stemming methods at the pre-processing stage are examined and compared. Then two knowledge bases, namely Wikipedia and WordNet, are used to provide concepts to represent Arabic text using a BOC model. A comparison between these knowledge bases is conducted and the one yielding the best accuracy is used to provide concepts to build a combined text representation model using the AC, the RTC and the ACC strategies, as well as to develop two new combined model strategies. These combined models are then used in our Arabic ATC system and the classification accuracy achieved by each is evaluated and compared to the use of the BOW or BOC alone. The paper is organised as follows: Sect. 2 reviews related work on Arabic ATC. Section 3 then describes the construction of our Arabic ATC system with a specific focus on Arabic text representation. The experimental setup and the datasets used in this work are described in Sect. 4. Section 5 presents and

discusses our experimental results, and Sect. 6 concludes the work and highlights our main findings.

2 Arabic ATC - Related Work

Compared to English ATC, the field of Arabic ATC is underdeveloped. Text representation for Arabic ATC is therefore a relatively new field and, hence, limited work has been published in this field. To-date, most reported works focus on comparing different stemming methods, investigating the impact of pre-processing, applying different classification algorithms and evaluating their effects on the classification of Arabic text, as described below.

Researchers such as Harrag et al. [15] compared different stemming methods in ATC for Arabic text. The authors compared three stemming methods, namely Light Stemming (LS) [16], Root Extraction (RE) [16] and dictionary-lookup method [17]. The RE method works by removing the suffixes and prefixes attached to a given word and word pattern matching is used to extract the root of the word. The LS method only involves removing a small set of prefixes and suffixes. The LS does not deal with infixes or recognize patterns to find roots. The dictionary-lookup method uses dataset statistics to generate possible roots for a given word and estimates the probability of deriving the word from each of the possible roots. Harrag et al. [15] used the stemming methods to reduce the feature space of the VSM for two different classifiers, the Artificial Neural Networks (ANN) and the Support Vector Machine (SVM). To evaluate their ATC system's performance, an in-house Arabic dataset containing 453 documents distributed over 14 categories was used. Reported results showed that ANN yielded a better performance than the SVM. Furthermore, the dictionary-lookup stemming method performed better with ANN whereas the LS method performed better with an SVM classifier.

Al-Shammari et al. [18] proposed the local stemming method and compared it with the LS [19] and RE. This method selects the shortest form of a word among syntactically related words in a text. To evaluate the classification performance with different stemming methods, Al-Shammari used a dataset that was constructed by merging the Saudi News Papers (SNP) dataset and the Saudi Press Agency (SPA) dataset as collected by Al-Harbi et al. [20]. Only 2,966 documents were selected which belonged to three categories: "cultural", "social", and "general". In the classification experiments, a 10-fold cross-validation was used with the Naive Bayes (NB), SVM and k-Nearest Neighbours (k-NN) ML algorithms. The experiment results showed that the local stemming method significantly improved text classification accuracy, in comparison to other stemming methods, and worked better with the SVM classifier.

Other researchers compared the accuracy of using different classification algorithms in Arabic ATC systems. For example, Mesleh et al. [21] used the original words without using any stemming methods to build a BOW model. He used the Chi-squared (χ^2) as the Feature Selection (FS) technique to reduce the size of the feature space. The dataset used was collected from online Arabic newspaper archives and contained 1,445 documents that vary in length and fall into nine categories. The results showed that using an

SVM classifier yielded a better classification performance compared to using the k-NN and NB classifiers. It yielded a macro-average F1 score of 88.11% when evaluated using the in-house complied Arabic dataset. Mesleh's dataset was also used by Kanaan et al. [22]. They applied the LS stemming method proposed by [23] to build a BOW model for Arabic ATC. A comparison was performed between the k-NN, Rocchio, and NB as classifiers with different weighting methods such as the Term Frequency (TF), the Term Frequency Inverse Document Frequency (TFIDF) and the Weighted Inverse Document Frequency (WIDF). Their results showed that the WIDF scheme yielded the best performance when used in conjunction with the k-NN, while TFIDF yielded the best performance when used in conjunction with Rocchio. Among the above three classifiers, the NB classifier was reported to be the best, yielding a macro-averaged F1 score of 84.53%.

Al-Harbi et al. [20] compared the SVM and C5.0 classification algorithms in Arabic ATC. The original text was used without any stemming to build the BOW model, and the Chi-squared was used as the FS technique to reduce the size of the feature space. The C5.0 provided a better performance than the SVM. On the other hand, Alsaleem et al. [24] compared the SVM and NB classification algorithms using the BOW model to classify the SNP dataset collected by [20], and his experiment results showed that the SVM algorithm outperformed the NB algorithm.

In terms of text representation models, most reported works in Arabic ATC have used the BOW model. For example, Khreisat et al. [25] used an N-gram frequency statistical technique to compare two similarity measures, the Manhattan distance and the Dice's coefficient. The authors used a dataset collected from four online Jordanian Arabic newspaper archives. Tri-grams were used to represent each document after pre-processing by removing punctuation marks, diacritics, non-letters and stop words. The Khoja RE stemming method [26] was applied to the remaining words. The chosen similarity measures were used with 40% of the dataset utilised for training and the rest for testing. Reported results showed that the best accuracy was obtained using the Dice coefficient in conjunction with the tri-gram frequency method.

Others have used statistical phrases to represent Arabic text for ATC. For example, Al-Shalabi et al. [27] compared the use of the BOW and phrases text representation models. The k-NN algorithm was used to classify documents from the dataset created by [21]. All Arabic documents were pre-processed by removing stop words, non-letters, and punctuation marks. Two independent text representation models were then built, namely a BOW model and a bag of phrases model with each phrase composed of two words. To train the classifier, 60% of the dataset was used for training and the rest for testing. The results showed that using phrases for text representation in conjunction with the k-NN classifier outperformed the BOW model and yielded an average accuracy score of 73.57% as compared to a score of 66.88% for the BOW model.

Elberrichi et al. [28] used the Arabic WordNet (AWN) to identify concepts appearing within the documents. A comparison between the use of different text representation models, utilizing words, N-grams and words and concepts combined using the RTC strategy, was conducted on the Arabic text dataset collected by [21]. The combined model, used in conjunction with the Chi-squared as the FS technique and a *k*-NN classifier, was reported to achieve higher performance results compared to other

representations. Yousif et al. [29] developed two new representation models based on lexical and semantic relations extracted from the Arabic WordNet, namely the List of Pertinent Synsets (LoPS) and the List of Pertinent Words (LoPW). The LoPS is the list of concepts (synsets) that have relations with the documents' original terms, whereas LoPW is the list of words that have relations with the documents' original terms. The authors compared the developed representation models with the standard BOW and BOC representation models. In the classification experiments, a 10-fold cross-validation was used with an NB classifier to classify documents from the Arabic BBC dataset [30]. The experiment results showed that both developed representation models improved the accuracy of ATC as compared to the standard BOW and BOC models. In addition, it was found that the LoPW model outperforms the LoPS model.

3 Text Representation for Arabic ATC

In order to investigate and assess the effect of commonly used text representation models on the classification accuracy of Arabic ATC, an experimental ATC system has been built. As described in Sect. 1, the first component of this ATC system performs the pre-processing of the text, where the text is converted to a well-defined set of features. To achieve this, first the text is tokenised by breaking it up into individual and meaningful units known as tokens. Each token is separated from others by a particular character or symbol. In written Arabic, words are separated by a space and each word is considered as a single meaningful unit. Hence, the tokenisation involves keeping these words and removing all other remaining punctuation marks, digits and numbers as they are considered noise. Then, common words such as pronouns, prepositions and articles are removed from the text. These words are called "stop words" and occur frequently and, therefore, have no discriminatory significance. This is then followed by replacing all words with their roots or stemmed forms, where morphological information is used to merge various word forms, such as plurals and verb conjugations, into their distinct roots. In this work, we have focused on the Root Extraction (RE) and the Light Stemming (LS) as the two most commonly used stemming methods for Arabic text as indicated in our Related Work section.

Next, all rare words are identified and removed based on their frequency of appearance in the whole dataset, using a threshold of four that has been chosen by experimentation. This process also reduces the complexity of the text representation model and improves the training time of the classifier. Next, the remaining words are passed to the text representation component of the ATC system as valid features, which make the dimensions of the resulting VSM. Depending on the text representation strategy used, different types of features are employed to represent text. For example, in the case of the BOW model, the features are simply the remaining words as per above. Variations of the BOW model are built based on the stemming method of the text pre-processing component of the system. Hence, three BOW models have been investigated; a BOW-RE, BOW-LS and a BOW-Original where no stemming method is applied.

In the case of the BOC model, different types of BOC are built depending on the knowledge base used to provide the concepts. In this work both Wikipedia and WordNet

are employed as knowledge bases and the resulting BOC models are compared to find the best knowledge base. To identify concepts using Arabic WordNet, an open-source toolkit called Arabic WordNet (AWN) is used. For each word in a text, the AWN browser searches its database and returns an ordered list of synonyms and the first synonym in the list (i.e., the most commonly used sense) is used as the concept for the word. To build the Wikipedia-based BOC representation model, the Arabic Wikipedia XML dump files (http://dumps.wikimedia.org/arwiki/), consisting of 273,709 articles, is used in this work. An open source toolkit known as Wikipedia-Miner [31] has been used to process the dump files and create a database that contains a summarized version of the Arabic Wikipedia's content and structure.

When both words and concepts are used together as features, we get a combined text representation model which can be built using different strategies, such as the AC, the RTC or the ACC as described in Sect. 1. In our ATC system, we have applied and compared different strategies to build combined text representation models. These include the AC, RTC and AC, as well as two in-house developed strategies, which are our attempt at developing new combined text representation models with a relatively reduced vector size as described in the following section.

3.1 New Combined Text Representation Strategies

In this section, we describe two in-house developed strategies for building a combined text representation model, namely Adding Unmapped Concepts (AUC), and using Concepts for Terms which do not appear in the Document (CTD).

Adding Unmapped Concepts (AUC). This strategy first involves creating a "Concept-Words Map" for the whole dataset. To achieve that, we map each concept identified in the dataset to the corresponding word(s) that share the unique meaning provided by the concept. By doing this, the algorithm will resolve synonyms and capture different words which refer to the same concept in the documents of the training subset. These words are considered as alternative labels for the concept in the "Concept-Words Map". The concept's label provided by the Knowledge Base (KB) is considered as the preferred label. In order to build a combined model to represent a given document using the AUC strategy, the following tasks have to be performed. Firstly, a BOW vector has to be created for the document and all words that are considered as features in the BOW model are added to the AUC model's vector. In this way, words which have a significant frequency value in the BOW model and carry an important value for the classification task are kept in the AUC model. Then, for each concept in the BOC model of the document, the "Concept-Words Map" is checked for alternative labels of the concept. If one of the alternative labels appears in the text, the preferred label for the concept is added to the AUC model. Otherwise, nothing is added regarding that concept and we move to the next one. The difference between the AUC strategy and AC strategy is that only concepts mapped to different alternative labels are added to the combined representation model.

Using Concepts for Terms not appearing in the Document (CTD). This strategy first involves creating a "Concept-Words Map" for the whole dataset. To achieve that, we map each concept. All previously proposed combined model strategies use both concepts and words with their corresponding weights. As a result, the numbers of features in resulting model are larger than the number of features in a corresponding BOW or BOC model. The CTD strategy does not change the original dimensions of the BOW model. It is similar to the BOW model in two ways; the size of the VSM and the type of features that are used to represent a document, where each dimension in the VSM corresponds to a word from the BOW dictionary. The strategy works like the BOW model for words that appear in the document, where their corresponding dimensions in the VSM will have the value of the words' TFIDF weights. The only difference is related to those words that do not appear in the document but have a corresponding concept in the document's BOC model. For these words, the weight of their related concepts in the document will be used as corresponding dimensions in the VSM. Hence, the CTD works as follows. First, a "Word-Concepts Map" is created for the whole dataset. To achieve this, each word that has been identified as a concept(s) is mapped to its corresponding concept(s). By doing this, the algorithm resolves synonyms and capture different words which refer to the same concept in all the documents of the training subset. In addition, words that have multiple meaning are mapped to different concepts. To represent a document using the CTD strategy and build the combined model, first all words which are considered as features in the BOW model dictionary are checked for their appearance in the document. In the case that a given word appears in the document, the word's TFIDF weight in the document is used in the CTD model. If the word does not appear in the document, the "Word-Concepts Map" is checked to see if that word has a corresponding concept. If it does, the concept is retrieved from the "Word-Concepts Map" and checked for its existence in the document's BOC model. If the concept exists, the word's corresponding dimension in the CTD model is assigned to the weight of the concept. If the corresponding concept does not exist, the word will not be represented in the CTD combined model.

4 Datasets and Experimental Setup

In order to provide a baseline for an objective assessment and comparison of the performance of our Arabic ATC system and proposed text representations, we have used three of the most frequently employed datasets in all cited similar work. The documents in these datasets, which have been collected from on-line web sites, are all written in Modern Standard Arabic (MSA). The datasets used are:

- Arabic 1445 dataset: this dataset was created by Mesleh et al. [21] from online Arabic newspaper archives containing news articles from Al-Jazeera, Al-Nahar, Al-hayat, Al-Ahram, and Al-Dostor. The dataset contains 1,445 documents that vary in length and fall into nine categories: computer, economics, education, engineering, law, medicine, politics, religion, and sports.
- Saudi News Papers (SNP) dataset: this dataset consists of 5,121 Arabic documents of different length which belong to seven categories: culture, economics, general,

information technology, politics, social, and sport. It has been collected by Al-Harbi et al. [20] and consists of articles and news stories from Saudi newspapers.
– Al-khaleej dataset: this dataset is a collection of 5,690 Arabic news documents gathered from the archive of the online newspaper Al-khaleej by Abbas et al. [32]. It consists of four categories: international news, local news, sport, and economy.

We conducted all the experiments using WEKA [33], which is a popular open source toolkit for ML. We first converted the textual documents into the format required by WEKA, i.e., ARFF format (Attribute-Relation File Format). We then used the data to train four separate classification algorithms, namely Support Vector Machines (SVM), Naive Bayes (NB), Decision Trees (DT), and Random Forest (RF). This was then followed by a 10-fold cross validation to evaluate the performance of the classifiers using the standard information retrieval measures of Precision, Recall, and F1.

5 Results and Discussion

In this section, results of our various experiments are presented and assessed, in terms of achieved F1 classification performance of the ATC system, with regards to:

– The effect of the pre-processing component on the performance the BOW model.
– The role of the characteristics of the knowledge base used to build the BOC.
– How the different BOW-BOC combined text representation models compare.

In the first stage of our experiments and evaluation, we focused on the effect of stemming on the overall classification accuracy. We conducted a comparison between the performance of our ATC system when the BOW is used with no stemming (i.e., Original-BOW) to that when stemming is applied in the pre-processing component using the LS and RE methods. Accordingly, three different BOW model representations have been built, BOW-LS, BOW-RE and the Original-BOW, for each dataset. Each of these BOW representation models has been used with four classification algorithms: SVM, NB, DT and RF. Our results here show that, in terms of classification algorithms, the SVM achieves the highest classification accuracy, as can be seen in Tables 1, 2 and 3. In addition, the SVM shows higher classification accuracy when used with the Original-BOW model for the cases of the large datasets as illustrated in Tables 2 and 3. This can be attributed to the SVM's ability to deal successfully with high-dimensional data [34].The DT algorithm seems to perform relatively well with the Original-BOW only for two datasets which have a large number of categories as illustrated in Tables 1 and 2. The BOW-LS representation seems to achieve the best average classification accuracy, particularly when applied to the two largest datasets, i.e., the SNP and Al-khaleej, as illustrated in Tables 2 and 3. The BOW-RE representation model on the other hand shows a good average performance with the Arabic 1455 dataset as can be seen in Table 1.

Table 1. F_1 scores for the three versions of BOW model for the Arabic1445 dataset with different classifiers.

Classifier	Original-BOW	BOW- based LS	BOW-based RE
SVM	0.90	0.92	0.91
NB	0.82	0.85	**0.88**
DT	**0.80**	0.79	0.77
RF	0.75	0.74	0.75
Average	0.81	0.82	**0.83**

Table 2. F_1 scores for the three versions of BOW model for the SNP dataset with different classifiers.

Classifier	Original-BOW	BOW- based LS	BOW-based RE
SVM	**0.81**	0.80	0.77
NB	0.63	**0.67**	0.66
DT	0.67	0.66	0.60
RF	**0.60**	0.59	0.57
Average	**0.67**	**0.68**	0.65

Table 3. F_1 scores for the three versions of BOW model for the Al-khaleej dataset with different classifiers.

Classifier	Original-BOW	BOW- based LS	BOW-based RE
SVM	**0.96**	0.94	0.92
NB	0.80	0.82	0.84
DT	0.86	0.87	0.83
RF	0.83	0.83	0.81
Average	0.86	**0.87**	0.85

The second stage of our experiments and evaluation focused on investigating the use of concepts as representation features for Arabic ATC. Two knowledge bases, namely WordNet and Wikipedia, were used to build the BOC representations model for our ATC system. Tables 4, 5 and 6 show achieved F1 scores when the ATC system uses a WordNet-based BOC and a Wikipedia-based BOC with different classifiers. All classifiers achieved higher accuracy when Wikipedia was used as a BOC knowledge base for all the datasets, as compared to WordNet. We believe this can be attributed to the fact that WordNet provides a BOC model with fewer concepts for representing Arabic text. Arabic WordNet has only 9,228 concepts compared to Arabic Wikipedia which has 273,709 concepts. Another reason for this is the ambiguity of the text as the documents in all the datasets are written in MSA and contain no diacritics. Arabic WordNet returns a ranked list of possible concepts for a word in the text, and the first ranked concept is the most commonly used, whereas Wikipedia selects concepts based on the

surrounding text. In addition, WordNet mostly provides information about individual words rather than general conceptual knowledge [35].

Table 4. F_1 scores for the BOC model for the Arabic1445 dataset using Wikipedia and WordNet with different classifiers.

Classifier	WordNet-based BOC	Wikipedia-based BOC
SVM	0.87	**0.89**
NB	0.83	**0.89**
DT	0.70	**0.79**
RF	0.67	**0.84**
Average	0.76	**0.85**

Table 5. F_1 scores for the BOC model for the SNP dataset using Wikipedia and WordNet with different classifiers.

Classifier	WordNet-based BOC	Wikipedia-based BOC
SVM	0.72	**0.75**
NB	0.57	**0.71**
DT	0.56	**0.65**
RF	0.51	**0.66**
Average	0.59	**0.69**

Table 6. F_1 scores for the BOC model for the Al-khaleej dataset using Wikipedia and WordNet with different classifiers.

Classifier	WordNet-based BOC	Wikipedia-based BOC
SVM	0.89	**0.92**
NB	0.79	**0.83**
DT	0.79	**0.85**
RF	0.77	**0.87**
Average	0.81	**0.87**

In the final stage of evaluation, we experimented with different strategies to build combined text representation models and compared corresponding resulting classification accuracy. We have evaluated the use of five different strategies, namely the AC, RTC, ACC, AUC and CTD. For each dataset, five combined representation models were built using Wikipedia concepts and words stemmed using the LS method. Our experimental results are presented in Tables 7, 8 and 9 in terms of obtained F1 scores. Regarding the classification algorithms, our results show that the SVM yield highest performance with all combined models. The NB algorithm comes next in terms of its accuracy, followed by the DT and the RF. The results also show that the ACC combined model achieves the best classification accuracy for two datasets as illustrated in Tables 7 and 8. The AC strategy seems to be the second best in terms of the classification

accuracy for two datasets, again as per Tables 7 and 8. The CTD strategy, on the other hand, seems to yield the best performance when used in conjunction with the NB algorithm for the two largest datasets as illustrated in Tables 8 and 9.

Table 7. F_1 scores for the different combined models for the Arabic1445 dataset with different classifiers.

Classifier	AC	RTC	ACC	AUC	CTD
SVM	**0.920**	0.912	**0.918**	0.917	0.919
NB	**0.872**	0.848	**0.873**	0.866	0.859
DT	**0.825**	0.807	0.815	**0.827**	0.792
RF	0.782	0.753	**0.805**	**0.783**	0.766
Average	**0.850**	0.830	**0.852**	**0.848**	0.834

Table 8. F_1 scores for the different combined models for the SNP dataset with different classifiers.

Classifier	AC	RTC	ACC	AUC	CTD
SVM	0.809	0.793	**0.824**	0.805	**0.824**
NB	0.651	0.626	**0.687**	0.650	**0.694**
DT	0.660	0.647	**0.676**	0.656	**0.668**
RF	0.598	0.564	0.614	0.581	0.593
Average	0.679	0.657	**0.700**	0.673	**0.694**

Table 9. F_1 scores for the different combined models for the Al-khaleej dataset with different classifiers.

Classifier	AC	RTC	ACC	AUC	CTD
SVM	0.809	0.793	**0.824**	0.805	**0.824**
NB	0.651	0.626	**0.687**	0.650	**0.694**
DT	0.660	0.647	**0.676**	0.656	**0.668**
RF	0.598	0.564	0.614	0.581	0.593
Average	0.679	0.657	**0.700**	0.673	**0.694**

From Table 10 it can be concluded that using the LS stemming method in the preprocessing stage provides better classification performance than employing the RE method, which is in line with the findings of other researchers [15, 36, 37]. We believe this is due to the fact that the RE method is harsher on words in comparison to the LS method. Using RE, two words with different meanings could be stemmed to the same root, which leads to misclassification. For example, the words "العالميه" (global) and "العلميه" (scientific) would share the same root, (علم) "science" if the RE stemming method is used. Furthermore, Table 10 shows that using Wikipedia concepts to build a BOC model for Arabic ATC yields better classification accuracy than using a BOW representation. One of the reasons for this, we believe, is the broad categories of the documents of the Arabic datasets used in this work; all datasets contained documents from

different newspapers and did not focus on specific topics. The other reason is the complex nature of the Arabic language and the poor morphological tools available, which make Wikipedia concepts better features for representing text compared to words.

Table 10. Average F_1 scores for all text representation models and for all datasets.

Original-BOW	BOW-based LS	BOW-based RE	WordNet-based BOC	Wikipedia-based BOC	AC	RTC	ACC	AUC
0.786	0.791	0.776	0.723	**0.802**	**0.804**	0.786	**0.812**	0.800

Our results have also shown that all the combined models we experimented with achieved higher classification accuracy than the BOW representation model, with the best performance achieved by the ACC strategy followed by the AC and the CTD. On the other hand, all combined models outperformed the BOC model. Finally, the results in Tables 1, 2, 3, 4, 5, 6, 7, 8 and 9 suggest that the RF algorithm provides relatively higher classification accuracy, compared to other classifiers, when used with the BOC representation model for all datasets.

6 Conclusion

In this work we have built an ATC system for Arabic text and evaluated its performance using three different datasets, with the goal of identifying key elements of text representation that influence the classification accuracy. The evaluation involved using a number of different text representation models in association with different machine learning techniques. While work reported in the literature has mainly concentrated on the BOW based representation models, our study focuses on comparing the classification performance of the BOW and BOC models with those of various combinations of both. Furthermore, two knowledge bases (i.e., Wikipedia & WordNet) were examined for building a BOC model, making our study the first of its kind to utilize Wikipedia as a knowledge base for Arabic ATC.

In conclusion, each component in an ATC system plays an important role in the classification accuracy, with text pre-processing and representation being key elements as demonstrated by our experimental results. We believe the findings of this study pave the way for venues for further research. Among those is the use of Wikipedia concepts to represent Arabic text and its application for providing richer representation models for automatic classification of more specialized textual datasets.

References

1. Salton, G., Wong, A., Yang, C.-S.: A vector space model for automatic indexing. Commun. ACM **18**, 613–620 (1975)
2. McCallum, A., Nigam, K.: A comparison of event models for naive bayes text classification. In: AAAI-98 Workshop on Learning for Text Categorization, vol. 752, pp. 41–48 (1998)
3. Sebastiani, F.: Machine learning in automated text categorization. ACM Comput. Surv. (CSUR) **34**, 1–47 (2002)

4. Hotho, A., Staab, S., Stumme, G.: Wordnet improves Text Document Clustering (2003)
5. Gabrilovich, E., Markovitch, S.: Feature generation for text categorization using world knowledge. In: IJCAI, vol. 5, pp. 1048–1053 (2005)
6. Gabrilovich, E., Markovitch, S.: Overcoming the brittleness bottleneck using Wikipedia: enhancing text categorization with encyclopedic knowledge. In: AAAI, vol. 6, pp. 1301–1306 (2006)
7. Kehagias, A., Petridis, V., Kaburlasos, V.G., Fragkou, P.: A comparison of word-and sense-based text categorization using several classification algorithms. J. Intell. Inf. Syst. **21**, 227–247 (2003)
8. de Buenaga Rodríguez, M., Hidalgo, J.M.G., Agudo, B.D.: Using WordNet to complement training information in text categorization. arXiv preprint cmp-lg/9709007 (1997)
9. Scott, S., Matwin, S.: Text classification using WordNet hypernyms. In: Use of WordNet in Natural Language Processing Systems, Proceedings of the Conference, pp. 38–44 (1998)
10. Wang, P., Hu, J., Zeng, H.-J., Chen, L., Chen, Z.: Improving text classification by using encyclopedia knowledge, pp. 332–341 (2007)
11. Wang, P., Hu, J., Zeng, H.-J., Chen, Z.: Using Wikipedia knowledge to improve text classification. Knowl. Inf. Syst. **19**, 265–281 (2008)
12. Benkhalifa, M., Mouradi, A., Bouyakhf, H.: Integrating external knowledge to supplement training data in semi-supervised learning for text categorization. Inf. Retr. **4**, 91–113 (2001)
13. Hu, J., Fang, L., Cao, Y., Zeng, H.-J., Li, H., Yang, Q., Chen, Z.: Enhancing text clustering by leveraging Wikipedia semantics. In: Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 179–186. ACM (2008)
14. Hotho, A., Staab, S., Stumme, G.: Wordnet improves text document clustering, pp. 541–544 (2003)
15. Harrag, F., El-Qawasmah, E., Al-Salman, A.M.S.: Stemming as a feature reduction technique for arabic text categorization. In: 2011 10th International Symposium on Programming and Systems (ISPS), pp. 128–133. IEEE (2011)
16. Syiam, M.M., Fayed, Z.T., Habib, M.B.: An intelligent system for Arabic text categorization. Int. J. Intell. Comput. Inf. Sci. **6**, 1–19 (2006)
17. Darwish, K., Oard, D.W.: Adapting morphology for Arabic information retrieval*. In: Soudi, A., van den Bosch, A., Neumann, G. (eds.) Arabic Computational Morphology. TLTB, vol. 38, pp. 245–262. Springer, Dordrecht (2007). https://doi.org/10.1007/978-1-4020-6046-5_13
18. Al-Shammari, E.T.: Improving Arabic document categorization: introducing local stem. In: 2010 10th International Conference on Intelligent Systems Design and Applications (ISDA), pp. 385–390. IEEE (2010)
19. Larkey, L.S., Ballesteros, L., Connell, M.E.: Light stemming for Arabic information retrieval. In: Soudi, A., van den Bosch, A., Neumann, G. (eds.) Arabic Computational Morphology, vol. 38, pp. 221–243. Springer, Dordrecht (2007). https://doi.org/10.1007/978-1-4020-6046-5_12
20. Al-Harbi, S., Almuhareb, A., Al-Thubaity, A., Khorsheed, M., Al-Rajeh, A.: Automatic Arabic text classification (2008)
21. Moh'd A Mesleh, A.: Chi square feature extraction based SVMs Arabic language text categorization system. J. Comput. Sci. **3**, 430–435 (2007)
22. Kanaan, G., Al-Shalabi, R., Ghwanmeh, S., Al-Ma'adeed, H.: A comparison of text-classification techniques applied to Arabic text. J. Am. Soc. Inform. Sci. Technol. **60**, 1836–1844 (2009)

23. Larkey, L.S., Ballesteros, L., Connell, M.E.: Improving stemming for Arabic information retrieval: light stemming and co-occurrence analysis. In: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 275–282. ACM (2002)
24. Alsaleem, S.: Automated Arabic text categorization using SVM and NB. Int. Arab J. e-Technol. **2**, 124–128 (2011)
25. Khreisat, L.: A machine learning approach for Arabic text classification using N-gram frequency statistics. J. Informetr. **3**, 72–77 (2009)
26. Khoja, S., Garside, R.: Stemming arabic text. Computing Department, Lancaster University, Lancaster, UK (1999)
27. Al-Shalabi, R., Obeidat, R.: Improving KNN Arabic text classification with n-grams based document indexing. In: Proceedings of the Sixth International Conference on Informatics and Systems, Cairo, Egypt, pp. 108–112. Citeseer (2008)
28. Elberrichi, Z., Abidi, K.: Arabic text categorization: a comparative study of different representation modes. Int. Arab J. Inf. Technol. (IAJIT) **9**, 465–470 (2012)
29. Yousif, S.A., Samawi, V.W., Elkabani, I., Zantout, R.: The Effect of Combining Different Semantic Relations on Arabic Text Classification
30. Saad, M.K., Ashour, W.: Osac: open source arabic corpora. In: 6th ArchEng International Symposiums, EEECS, vol. 10 (2010)
31. Milne, D., Witten, I.H.: An open-source toolkit for mining Wikipedia. Artif. Intell. **194**, 222–239 (2013)
32. Abbas, M., Smaili, K.: Comparison of topic identification methods for arabic language. In: Proceedings of International Conference on Recent Advances in Natural Language Processing, RANLP, pp. 14–17 (2005)
33. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The WEKA data mining software: an update. ACM SIGKDD Explor. Newsl. **11**, 10–18 (2009)
34. Ben-Hur, A., Weston, J.: A user's guide to support vector machines. In: Carugo, O., Eisenhaber, F. (eds.) Data Mining Techniques for the Life Sciences. Methods in Molecular Biology, vol. 609, pp. 223–239. Humana Press, New York (2010). https://doi.org/10.1007/978-1-60327-241-4_13
35. Gabrilovich, E., Markovitch, S.: Wikipedia-based semantic interpretation for natural language processing. J. Artif. Intell. Res. **34**, 443–498 (2009)
36. Duwairi, R., Al-Refai, M.N., Khasawneh, N.: Feature reduction techniques for Arabic text categorization. J. Am. Soc. Inform. Sci. Technol. **60**, 2347–2352 (2009)
37. Saad, M.K.: The impact of text preprocessing and term weighting on Arabic text classification. The Islamic University-Gaza (2010)

Modern Standard Arabic Readability Prediction

Naoual Nassiri[1(✉)], Abdelhak Lakhouaja[1], and Violetta Cavalli-Sforza[2]

[1] Department of Computer Science, Faculty of Sciences, University Mohamed First, B-P 717, 60000 Oujda, Morocco
naoual.nassiri@gmail.com, abdel.lakh@gmail.com
[2] School of Science and Engineering, AI Akhawayn University, Ifrane, Morocco
v.cavallisforza@aui.ma

Abstract. Reading is the most critical skill for satisfactory progress in school, as well as being highly important for access to information throughout one's life. For this reason, readability is one of the main challenges when choosing academic texts for learners or for readers in general, and especially with materials containing important information, such as newspapers and medical or legal articles. Readability refers to the ability of a text to be understood by the reader. Readability level prediction is an important measure in several domains, but primarily in education. In the current paper we present our approach to readability prediction for Modern Standard Arabic. This method is based on 170 features of measuring different types of text characteristics. We have used a corpus of 230 Arabic texts, annotated with the Interagency Language Roundtable (ILR) scale, and a frequency dictionary obtained using Tashkeela corpora. The results obtained are very encouraging and better than for previously presented work.

Keywords: Readability · Modern Standard Arabic · Machine learning
Classification · Arabic readability

1 Introduction

Reading is not a natural skill; it is a whole learning process that requires using a suitable pedagogical program. Readers have the right to read texts that are adapted to their ability; negotiating texts significantly above their skill level, they may lose the overall meaning of the text when facing difficult or unknown vocabulary items or constructions. Usually, the easier a text is to read and the clearer the ideas it contains, the more it is likely to attract and retain the attention of the reader. It is therefore useful to measure text readability in terms of characteristics that are linked to clarity and ease of understanding.

Al-Khalifa and Al-Ajlan stated that "readability depends on three main factors: the reader, the text and the situation" [1]. Readability is a measure that binds a written text to a reader or a grade level. Readability is affected by the reader's ability to understand a text and is thus a crucial indicator for determining the population targeted by a given

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 120–133, 2018.
https://doi.org/10.1007/978-3-319-73500-9_9

text, having an impact on students' education on one side and on the general public (for example, newspaper readers) on the other. It can play an important role in many fields besides education, for example for disseminating information about health and legislation.

Readability depends both on the content of a text and its presentation. Readability can be influenced by the legibility of the text, which refers to characteristics such as the font with which the text is written, the colors that are used, the sharpness of the image, and other such visual features. In this paper, however, we will focus on text readability from the perspective of language-related features and ignore visual presentation features.

Readability measurement methods can be divided into two categories: traditional and modern methods. The traditional method consists of readability formulas, which categorize text automatically by calculating its readability score using a mathematical formula. There are around 200 different formulas that use language-independent features, such as the mean number of words per sentence, the mean number of characters per word, and so on. Tools for automatically calculating text scores do not exist for most of the formulas[1]. Besides, formulas are language specific (Arabic formulas, French formulas, etc.).

Research on readability measures for education began in the 1920s for English, and was later extended to other European and Asian languages. Among the most popular traditional formulas that use features independent of language-specific characteristics, we mention two for English. The New Dale and Chall formula [2] is based on a list of approximately 3000 words known in reading by at least 80% of children in Grade 5. It uses the number of words in the text not belonging to the list and the number of words per sentence. The Flesch Reading Ease formula [3], related to the "Flesch-Kincaid Grade Level", uses two features: average sentence length in words and average number of syllables per word.

Modern methods for assessing readability have been based on machine learning models. The latter classify new observations by learning the appropriate classification from a set of data where each element is already labeled with its correct class. In this kind of classification, we need a training set that is annotated with the class to which each example belongs. For reading in one's native language, class annotations can be grade levels in school curricula or broader categories grouping more than one grade level. Another well-known scale is the Interagency Language Roundtable (ILR) scale, which was developed to describe abilities to communicate in a language, particularly in the context of second language learning [4].

In the remainder of the paper we begin, in Sect. 2, by reviewing previous work on Arabic readability, describing some of the most used formulas and presenting some previous research using machine learning models. In Sect. 3, we present the ILR scale. In Sect. 4, we describe our approach and give details about the data, tools and methodology we used. In Sect. 5, we discuss and evaluate the results through a comparative study to previous research. We conclude with some thoughts on future work.

[1] https://readable.io/text/.

2 Related Work

There has been a great deal of research on readability in the English and other European—and more recently Asian—languages, but relatively little attention has been paid to the Arabic language. This section presents some formulas for measuring readability in Arabic texts, highlighting three recent studies that have used machine learning approaches on a Modern Standard Arabic corpus intended for second language learning.

2.1 Readability Formulas

Readability formulas are, in all, a process that takes as input a corpus of texts and uses characteristics of those texts to estimate the coefficients of a mathematic formula whose goal is to calculate a difficulty level for the text.

There exist a few formulas that are used to measure readability of Arabic, of which we mention the Dawood and El-Heeti formulas [3] (the first formulas to have been proposed for the Arabic language), the ARI index [5], and the newer AARI formula [6] and OSMAN [7] formulas.

Dawood Formula. The formula uses three features to measure the ease of reading: average word length in letters, average sentence length in words and average word repetition.

El-Heeti. This formula uses the average word length in characters as the only feature.

ARI Formula. The Automated Readability Index (ARI) is an index designed to measure the comprehensibility of texts. It produces an approximate representation of the required level of study to understand the text. The formula for calculating the ARI readability index is as below:

$$\mathrm{ARI} = 4.71 * \mathrm{C}/\mathrm{W} + 0.5 * \mathrm{WPS} - 21.43 \quad (1)$$

where:

C = the number of characters
W = the number of words
WPS = the average sentence length in words

AARI Formula. The Automatic Arabic Readability Index is a formula that is based on more than 1196 Arabic texts extracted from the Jordanian curriculum. Application of the method consists of 3 basic phases. The first phase normalizes a given text, the second one converts particular Arabic letters (إ, أ, and آ) to ا (alif), and the last phase extracts the features. The formula obtained is:

$$\mathrm{AARI} = (3.28 \times \mathrm{CH}) + (1.43 \times \mathrm{ACW}) + (1.24 \times \mathrm{AWS}) \quad (2)$$

where:

CH = the number of characters
ACW = the average number of characters per word
AWS = the average number of words per sentence

OSMAN Formula. OSMAN means Open Source Metric for Measuring Arabic Narratives. The creators of this formula used a parallel corpus for English and Arabic composed of about 73,000 undiacriticized texts, which were collected from the United Nations (UN) corpus. The formula calculation process used Mishkal[2] to diacriticize Arabic text. The use of diacriticized texts is problematic because Arabic texts often are not and the process of introducing diacritics, if done automatically, can introduce error, so this is one of the weak points of the formula.

$$\text{OSMAN} = 200.791 - (1.015 * \text{A/B}) - 24.181 * (\text{C/A} + \text{D/A} + \text{G/A} + \text{H/A}) \quad (3)$$

where:

A = the total number of words
B = the total number of sentences
C = the total number of hard words (words with more than 5 letters)
D = the number of syllables per word
G = the number of complex words (words with more than 4 syllables)
H = the number of "Faseeh" words (complex words with any of the following Arabic letters "ذ, ظ, ؤ, ئ, ء" or ending with "ا, ون".

2.2 Machine Learning Approaches

In this section we introduce readability as a machine learning model by briefly reviewing three recent studies. We compare their results to ours further in the paper.

Text Readability for Arabic as a Foreign Language [8]
This research studied readability of texts for learners of Arabic as a foreign language from a machine learning perspective, using 251 Modern Standard Arabic texts of a corpus collected from the Global Language Online Support System (GLOSS[3]), annotated with the ILR scale. All the corpus files were split into sentences in order to prepare MADAMIRA [9] input files. From the MADAMIRA output, which only contains information that the user has explicitly requested through the configuration settings, 35 features were extracted and then employed in the classification phase performed using the WEKA tool [10].

Automatic Readability Detection for Modern Standard Arabic [11]
This work, like the previous, treated readability as a classification problem using the "GLOSS" corpus, whose documents are ranked using the ILR standard levels. It prepared all the corpus files in an MADAMIRA input file format. From the

[2] https://sourceforge.net/projects/mishkal/.
[3] https://gloss.dliflc.edu/.

MADAMIRA output it extracted 162 features used in the classification phase, which was performed using the TiMBL[4] package.

Arability [1]
This research used a corpus manually collected from reading books of elementary, intermediate, and secondary curricula of Saudi Arabian Schools. The corpus consists of 150 texts ranked manually by three readability levels: easy, medium, and difficult. After a normalization phase, five features, previously used in readability assessment for other languages, were extracted: average sentence length, average word length, average number of syllables per word, word frequencies and the perplexity scores for a bigram language model (LM) built from the same corpus.

3 Interagency Language Roundtable Scale

The ILR scale is a description of communicative abilities in a language. This scale evaluates the language skills using a scale ranging from 0 to 5:

- ILR Level 0 - No expertise;
- ILR Level 1 - Elementary competence;
- ILR Level 2 - Limited working proficiency;
- ILR Level 3 - General occupational competence;
- ILR Level 4 - Advanced professional competence;
- ILR Level 5 - Bilingual or mother tongue speaker competence.

Levels 0+, 1+, 2+, 3+, or 4+ are used when the person's skills significantly exceed those of a given level, but are insufficient to reach the next level.

4 Dataset and Methodology

This section presents the data, tools and process used in the readability prediction system. The process consists of the following steps:

- the frequency dictionary building process;
- the analysis and features extraction phase;
- the classification operation, using the results obtained in previous steps.

4.1 Data and Tools Collection

Readability Corpus
We assembled the corpus to be used in the readability measurement from the Global Language Online Support System (GLOSS), which is a platform that offers thousands of lessons in dozens of languages. For the current work, the chosen language was

[4] https://languagemachines.github.io/timbl/.

MSA. The corpus contains 230 texts annotated according to the ILR scale. Table 1 describes the corpus distribution over 5 classes.

Table 1. 5-class text distribution

ILR level	Number of texts in the corpus
1	27
1^+	19
2	87
2^+	62
3	35

Table 2 describes the corpus distribution over 4 classes, obtained by collecting the files of level 1 and 1+ in a single class named "1_1^+".

Table 2. 4-class text distribution

ILR level	Number of texts in the corpus
1_1^+	46
2	87
2^+	62
3	35

The 3-class distribution was obtained by assembling the files of level 1 and 1+ and assembling the files of level 2+ and 3 (Table 3).

Table 3. 3-class text distribution

ILR level	Number of texts in the corpus
1_1^+	46
2	87
2^+_3	97

AraNLP Tool

We used AraNLP for the text segmentation of the GLOSS corpus, used in the classification process and the Tashkeela corpus, used in building the frequency dictionary (4.2). AraNLP is a Java library used for the processing of Arabic texts. It supports the most important steps in the processing of natural languages, such as diacritic and punctuation removal, tokenization, sentence segmentation, part-of-speech tagging, root stemming and word segmentation.

MADAMIRA Tool
We used MADAMIRA 2.1 as a Morphological Analysis and Disambiguation tool of Arabic. It has a specific input XML file format which contains a list of sentences and configuration options.

WEKA 3.6 Tool
Waikato Environment for Knowledge Analysis (WEKA) is an open source machine learning software tool. It groups a set of algorithms under one entity. It can be used directly or can be called within a Java code. WEKA contains data pre-processing, classification, regression, clustering, association rules, and visualization tools.

4.2 The Frequency Dictionary

Since we used frequency-based features in our approach, a frequency dictionary for the Arabic language was needed. Therefore we built a frequency dictionary using the process outlined in Fig. 1.

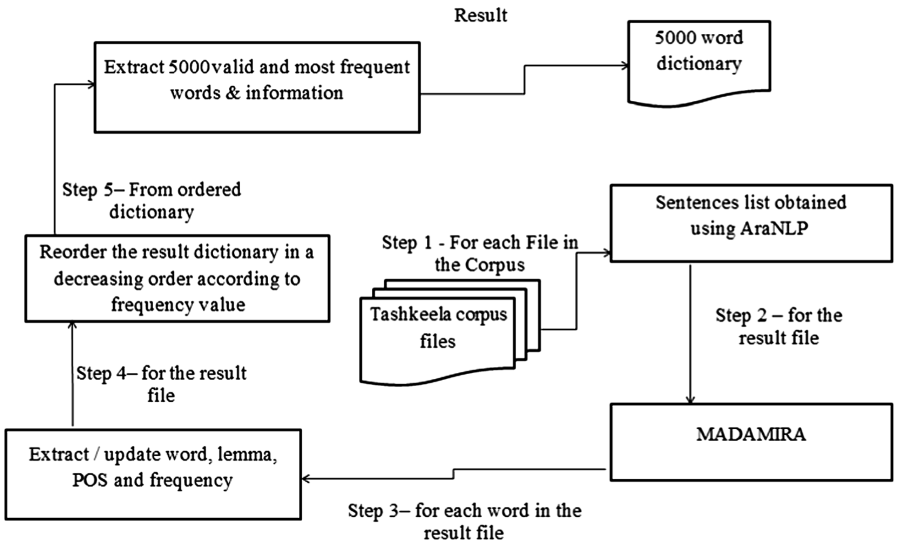


Fig. 1. Frequency dictionary building process

To build the Arabic frequency dictionary, we used the freely available Tashkeela[5] corpus [12] composed of around 70 million diacriticized Arabic words. We split each file in this corpus into sentences using AraNLP (Step 1). We prepared the resulting file as a MADAMIRA input file to get the analysis output. For each word in the MADAMIRA output file, we obtained from the most highly ranked result the lemma, the Buck Walter transliteration, the POS and the word itself. Then we converted them into a

[5] http://tashkeela.sourceforge.net.

frequency dictionary entry format by calculating the frequency of the pair (lemma, POS) and the POS frequency (that is, the number of appearances in the corpus).

The dictionary obtained contains the 5000 most frequent Arabic words, and their information. In Table 4, we give an extract from the dictionary.

Table 4. Frequency dictionary format

Rank	Transliteration	POS	RawFreq	PosFreq	Lemma
1	Fiy	prep	1,098,827	3,702,690	فِي
2	>an~a	conj_sub	830,705	1,745,928	أَنَّ
3	Min	prep	790,629	3,702,690	مِن
4	EalaY	prep	776,172	3,702,690	عَلَى
5	lA	part_neg	576,201	878,735	لا
6	qAl	verb	529,276	4,988,562	قال

Rank is the word position in the ranked word list. We get the lemma and transliteration in context (SVM prediction result from MADAMIRA). RawFreq means the number of appearances of the pair (Lemma, POS).

4.3 Readability Prediction Process

We consider the prediction of readability as machine learning task, and specifically a classification task. To automatically classify the GLOSS corpus files, we use a three stage process, which is depicted in Fig. 2 and further detailed below.

1. **Morphological Analysis:** The input of this phase is a GLOSS text file, which is segmented into sentences using the AraNLP library. We then add to the resulting sentence list some configuration options in an XML file. The latter is the input we give to MADAMIRA to get a result file, also in XML, containing a list of analyzed words with information such as POS, lemma, diacritics etc.
2. **Features Extraction:** We extract and calculate the 170 features. For the calculation of frequency-based features we use the frequency dictionary. In this phase we also eliminated 5 features (Sect. 4.4) from the original list of 170. Finally, for each file in the corpus, we obtained a features vector that we used to prepare the WEKA input file for the classification phase.
3. **Classification Phase:** In this final step, using WEKA, we apply a classification algorithm on the data using 80% of the data for training and 20% for testing. From the results, we get the accuracy value which specifies the percentage of well-classified text. This step is repeated six times for the six different chosen algorithms.

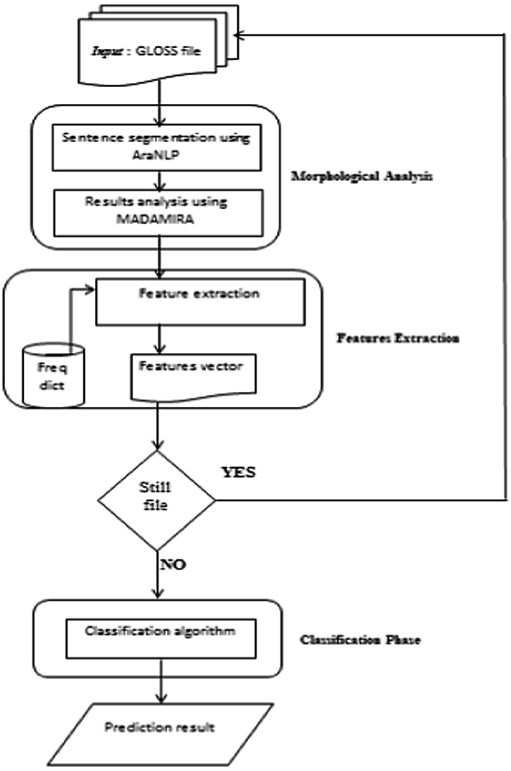


Fig. 2. Readability prediction process

The goal of the process described above is to evaluate the influence of the extracted features on the readability prediction.

4.4 Features

We initially collected 170 features grouping them into ten categories, and, after testing them in our corpus, we eliminated two categories (Foreign Word and Ambiguity categories) jointly containing four features (see Table 5). As a result, we used only eight feature categories containing 166 features in total and distributed as shown in Table 6.

Table 5. Features eliminated from the initial set of 170

Feature	Type
Ratio of foreign words to tokens	*Foreign word feature*
Number of ambiguous lemma	*Ambiguity features*
Ambiguous lemma to token ratio	*Ambiguity features*
Ambiguous lemma to lemma ratio	*Ambiguity features*

Table 6. Feature distribution into categories

Features category	Number of features
POS-based frequency	96
Type-to-token POS ratio	33
Token & type frequency	17
Type-to-token	4
Word length	5
Vocabulary load	3
Word class	4
Sentence length	4

Feature categories are defined as follows:

1. **POS-based Frequency features:** It includes the ratio of frequent adjectives to total number of adjectives, maximum frequency rank of adjectives tokens, minimum frequency rank of all noun* tokens (tokens whose POS tag starts with noun, e.g. noun_num), and the like.
2. **Type-to-Token POS Ratio features:** It includes adjective-to-token ratio, adverb-to-token ratio, conjunction-to-token ratio, and the like.
3. **Token & Type Frequency features:** It includes maximum dispersion[6] of frequent types, frequent type-to-token ratio, mean dispersion of frequent tokens, and the like.
4. **Type-to-Token features:** It includes morpheme type-to-token ratio, square root of morpheme type-to-token ratio, square root of lexeme (lemma) type-to-token ratio, and lexeme type-to-token ratio.
5. **Word Length features:** It includes average character length of surface forms, average length of voweled words, average morpheme length of words, token count, and number of characters per document.
6. **Vocabulary Load features:** It includes number of distinct types (lemmas) per document, number of frequent types (lemmas of types occurring more than once in the document), and frequent type-token ratios (calculated using values from the frequency dictionary).
7. **Word Class features:** It includes number of open-class tokens per document, open-class-token ratio, number of closed-class tokens per document and closed-class-token ratio.
8. **Sentence Length features:** It includes average sentence morpheme length, average sentence token length, number of sentences per document and average sentence length in characters.

This list of features was obtained from the previously mentioned approaches (Sect. 2).

[6] Dispersion = distribution over all sections of the corpus.

5 Evaluation Results and Discussion

As shown in Table 7, using for the classification phase the 166 features mentioned above, we achieved an accuracy value of 89.56% using five classes (see Table 1).

Table 7. Classification results with five classes

Model	Accuracy	F-score	Precision	Recall	RMSE
ZeroR	37.82%	0.208	0.143	0.378	0.3848
OneR	53.9%	0.518	0.557	0.539	0.4294
J48	83.9%	0.838	0.839	0.839	0.2248
IBk(k = 1)	**89.56%**	0.894	**0.905**	**0.896**	**0.1457**
SMO	82.17%	0.819	0.826	0.822	0.3286
Random forest	**89.56%**	**0.895**	0.897	**0.896**	0.1833

To improve the classification results, we then tried using four classes, combining classes 1 and 1+ to increase the number of files in to 46 (see Table 2). We recall that classes 1 and 1+ were the least populated with 27 and 19 documents respectively. The results are shown in Table 8.

Table 8. Classification results with four classes

Model	Accuracy	F-score	Precision	Recall	RMSE
ZeroR	37.82%	0.208	0.143	0.378	0.4246
OneR	58.26%	0.561	0.584	0.583	0.4568
J48	86.95%	0.869	0.881	0.87	0.2253
IBk(k = 1)	**89.56%**	0.894	**0.905**	**0.896**	**0.1628**
SMO	80.86%	0.806	0.814	0.809	0.3398
Random forest	**89.56%**	**0.895**	0.896	**0.896**	0.2021

Finally, even if the results with 4 classes were somewhat better than with 5 classes, the maximum accuracy achieved was the same. Hence, since there were also a limited number of texts at ILR level 3, we tried using 3 classes by merging the two most difficult levels of texts (see Table 3) and thereby achieved 90.43% as a maximum accuracy value, as is illustrated in Table 9.

Table 9. Classification results with three classes

Model	Accuracy	F-score	Precision	Recall	RMSE
ZeroR	42.17%	0.25	0.178	0.422	0.4615
OneR	66.52%	0.665	0.678	0.665	0.4724
J48	87.39%	0.873	0.888	0.874	0.2475
IBk(k = 1)	**90.43%**	0.873	**0.915**	**0.904**	**0.18**
SMO	82.60%	0.825	0.828	0.826	0.3447
Random Forest	**90.43%**	**0.905**	0.907	**0.904**	0.2193

To evaluate our approach, we present in Figs. 3 and 4 diagrams comparing our work with the results found by the Ibtikarat team, who used in their study only 35

features. They used a GLOSS corpus of 251 MSA texts, and for the tools, they were based on the same tools that we used, namely AraNLP, MADAMIRA, and WEKA. As shown in the figures, they achieved a classification accuracy rate of 73.31% using 5 classes and 59.76% using 3 classes.

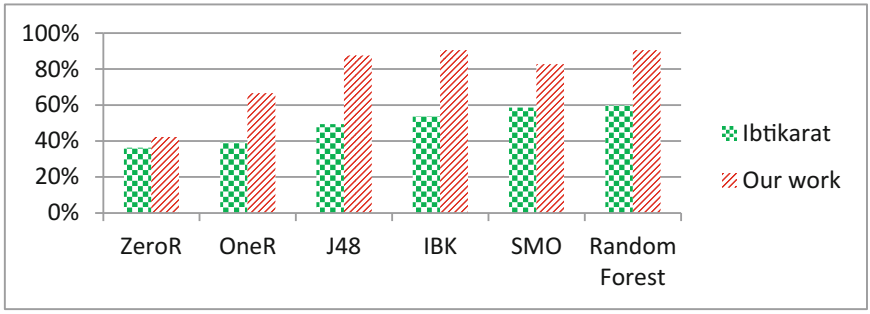


Fig. 3. Three-class comparative study

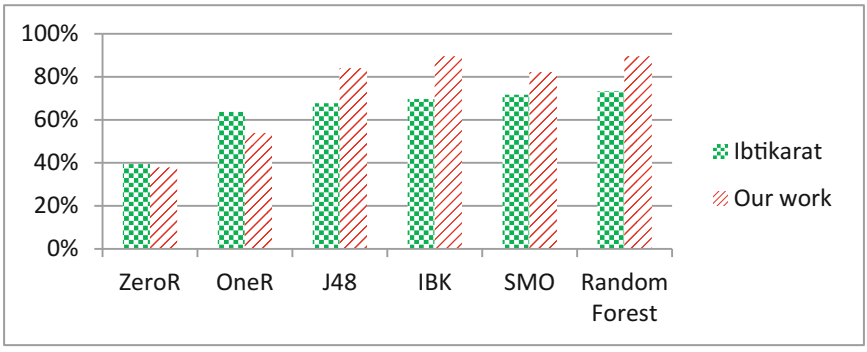


Fig. 4. Five-class comparative study

Figure 5 shows respectively the best accuracy rates obtained by Ibtikarat (73.31%), Arability (77.77%) and our work (90.43%).

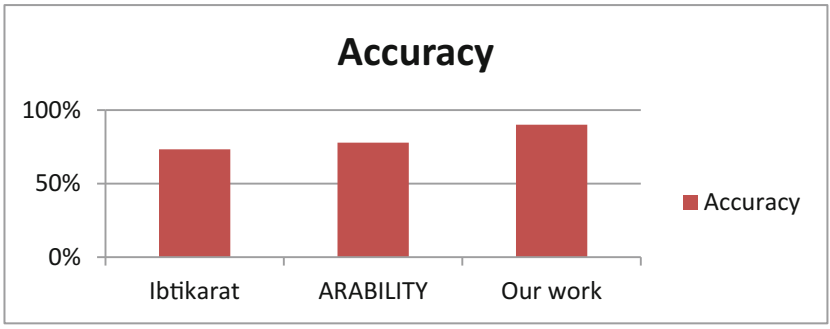


Fig. 5. Accuracy rate comparison of the 3 studies

6 Conclusion and Future Work

This paper proposes a method for predicting readability of MSA texts based on machine learning techniques and expressing readability in terms of the ILR scale, which is widely used for assessing second language competence. We were able to reach a prediction accuracy of 90.43%, using 3 classes with IBK and Random Forest classification algorithms, which is noticeably better than previous work on essentially the same corpus. Of the other algorithms that we used, some were able to achieve honorable results, with accuracy values which vary between 42.17% and 87.39% using 3 classes and between 37.82% and 89.56% using 4 and 5 classes. These results were obtained after examining 170 features used in readability measurement research and eliminating four of those features.

Although our results are good in terms of accuracy, it takes 12 h to generate the features vector, which is very time-consuming. This is due to the tools used in processing, and especially MADAMIRA. In a context where the speed of execution matters—such as searching for readable texts on the Web—it would be necessary to increase the speed of execution of our process or perform the search offline making available a collection of useful results from which to select rapidly when needed.

In our future work we aim to develop a tool for the Moroccan education system, increasing the accuracy rate and testing it with school students. In the same vein, we can imagine customizable readability measurements adapted to text domains such as medical notices, juridical texts, and other similar domains where readability is very consequential for readers.

References

1. Al-Khalifa, H.S., Al-Ajlan, A.: Automatic readability measurements of the Arabic text: an exploratory study. Arabian J. Sci. Eng. **35**(2C), 103–124 (2010)
2. Dale, E., Chall, J.S.: A formula for predicting readability: instructions. Educ. Res. Bull. **27**, 37–54 (1948)
3. Flesch, R.: A new readability yardstick. J. Appl. Psychol. **32**, 221 (1948)
4. Shen, W., Williams, J., Marius, T., Salesky, E.: A language-independent approach to automatic text difficulty assessment for second-language learners. DTIC Document (2013)
5. Senter, R.J., Smith, E.A.: Automated readability index. University of Cincinnati, Ohio (1967)
6. Al Tamimi, A.K., Jaradat, M., Al-Jarrah, N., Ghanem, S.: AARI: automatic Arabic readability index. Int. Arab. J. Inf. Technol. **11**, 370–378 (2014)
7. El-Haj, M., Rayson, P.E.: OSMAN: a novel Arabic readability metric (2016)
8. Saddiki, H., Bouzoubaa, K., Cavalli-Sforza, V.: Text readability for Arabic as a foreign language. In: 2015 IEEE/ACS 12th International Conference of Computer Systems and Applications (AICCSA), pp. 1–8 (2015)
9. Pasha, A., Al-Badrashiny, M., Diab, M.T., El Kholy, A., Eskander, R., Habash, N., Pooleery, M., Rambow, O., Roth, R.: MADAMIRA: a fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In: LREC, pp. 1094–1101 (2014)

10. Holmes, G., Donkin, A., Witten, I.H.: Weka: a machine learning workbench. In: Proceedings of the 1994 Second Australian and New Zealand Conference on Intelligent Information Systems, pp. 357–361. IEEE (1994)
11. Forsyth, J.: Automatic readability detection for modern standard Arabic. Theses dissertations (2014)
12. Zerrouki, T., Balla, A.: Tashkeela: novel corpus of Arabic vocalized texts, data for auto-diacritization systems. Data Brief. **11**, 147–151 (2017)

Document Similarity for Arabic and Cross-Lingual Web Content

Ali Salhi(✉) and Adnan H. Yahya

Birzeit University, Birzeit, Palestine
eng.salhi.ali@gmail.com, yahya@birzeit.edu

Abstract. Document similarity is basic for Information Retrieval. Cross Lingual (CL) similarity is important for many data processing tasks such as CL palgiarism detection and retrieval and document quality assessment. We study CL similarity based on the Explicit Semantic Association (ESA) adapted to a cross lingual setting with focus on Arabic. We compare the degree to which CL similarity testing performs where one of the language is Arabic with its monolingual counterpart for various text chunk sizes. We describe the used infrastructure and report on some of the testing results, study the possible sources of encountered weaknesses and point to the possible directions for improvement.

Keywords: Cross lingual information retrieval · Document similarity
Explicit Semantic Association · CL-ESA · Arabic information retrieval

1 Introduction

The growing size and diversity of online content necessitate sophisticated tools to retrieve needed information from the web. Search engines are some of the important tools to access web data. The main mode of operation is to match the user information need, generally expressed as a query, with web documents deemed similar, or related in some way, to that need. Similarity or relatedness can be applied to words, terms, phrases, text fragments and documents. It can take the shape of surface/lexical similarity in terms of having common words/characters, but could go deeper to look for semantically relevant documents by searching for terms not directly specified in the query. Similarity can be used to offer better formulations to the query posed. Classifying a document into one of a given set of categories can also be viewed as searching for similarity between the document at hand and sets of documents known to belong to given categories (training set). Document similarity is also important for plagiarism detection where one is interested in finding equivalent documents or document fragments that are adequately similar to the document or text fragment at hand, even when the text undergoes some editing. One can think of many more applications in IR where similarity may be utilized: detecting variants of proper names [11], detecting paraphrases with possible implications for document summarization, grading essay test answers by comparing with model answers and many more. One can also see the need for similarity between documents in different languages: Cross-Language (CL) document similarity [2, 6, 13]. Plagiarism can certainly cross languages and its detection will require CL similarity assessment. One may need to match

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 134–146, 2018.
https://doi.org/10.1007/978-3-319-73500-9_10

proper names in different languages [5, 7] and may use CL similarity to assess translation quality, CL information retrieval and CL text classification [6] and in retrieving multimedia elements annotated in a foreign language and related news articles [15] matching the user need. We are mostly interested in semantic text similarity/relatedness where we seek similarity in meaning even when the vocabularies of the texts are different. Compared text chunks need to be assigned a metric based on the likeness of their meanings or semantic content [3]. Unless explicitly specified, we use relatedness and similarity interchangeably. One needs to note that the concepts are not really interchangeable: while similar (in meaning) expressions are related (through their meanings), words can be related, say by frequently occurring together, but not necessarily semantically similar by being in the same domain or representing features of the same concept [12]. Examples are word pairs like (Cell, Phone), (Arab, Spring), (Press, Release), (الهندسي, الرسم) (الثانوية, العامة) which are related but are not strictly similar as opposed to pairs like (Fax, Phone), (Creek, Spring), (Press, Newspaper), (التقني, الهندسي), (الممثل, الفنان) which have similarity in meaning. Textual material like Wikipedia through term occurrence analysis tend to handle relatedness while knowledge bases like WordNet tend to better handle semantic similarity [9, 12]. One can talk about similarity between documents and also about similarity of shorter fragments of texts and tweets, blogs, discussion groups posts, captions of multimedia objects and headlines and mixes where similarity is assessed between a short text and longer texts such as matching an abstract with the corresponding document and query answering where the user query has to be matched to web documents of arbitrary length to answer user queries or matched against previous queries for query expansion/reformulation or for text summarization/abstracting.

Here we are mostly concerned with text similarity where Arabic is involved: similarity between Arabic text chunks and similarity between Arabic and Non-Arabic texts.

The rest of the paper is organized as follows: in the next section, we give survey the current state of the art in assessing text similarity. In Sect. 3, we review methods for assessing text similarity and discuss their applicability to Arabic. In Sect. 4, we report on our experiments on similarity assessment for Arabic and Cross Lingual. In the final section, we draw some conclusions and point to possible directions for future work.

2 Background

Text similarity, for single language and Cross-Lingual, has been a focus of much research lately, as a method for improved information retrieval. Work concentrated on similarity measures and uses of similarity for various tasks. [2] compares several approaches to text similarity between language pairs on Wikipedia. [5] offers a comprehensive survey of definitions, approaches, tools and evaluation methods for text similarity. Our interest in cross lingual text similarity stems from our desire to give users access to data in languages other than their own. We believe that the speakers of resource poor languages (Arabic is still in this category) can benefit from having access to data in resource rich languages (say English), even if they do not speak the foreign language.

2.1 Text Representation [10]

Similarity algorithms need to operate on text representations. Here we use words and use the bag of words paradigm. We use confusion letters normalization to account for Arabic letters that have multiple shapes and/or that are frequently misspelled {(ة,ه),(ي,ى),(إ,أ,آ,ا)}. We ignore Non-Arabic characters and numerals.

2.2 Similarity Measures

One needs to distinguish between lexical and semantic similarity. Each has its strong and weak points. We are interested in similarity of texts not only in a single language but also in cross lingual (CL). Our focus will be on the case when Arabic is an element of the texts compared. We deal exclusively with MSA texts (no dialects). The main similarity measure we employ is cosine similarity.

Lexical Similarity
The text is represented as a vector of its constituent words, possibly with term frequencies (TF) and inverse document frequency (IDF) and standard metrics (e.g. cosine) are used to measure the distance between the representations of the two text chunks as the similarity measure. Documents are ranked according to the metric used. The size of the vector can be as large as the number of words in the language/corpus (vocabulary), which can be quite large. Clearly, for moderately sized texts the resulting vectors are sparse and more so for shorter texts. One may use truncation techniques to limit the vector size and speed up the computations.

For Cross Lingual similarity one needs to transform one of the texts into the language of the other, say through dictionary lookup or more sophisticated translation, and compare the resulting vectors (in the same language). Clearly, table lookup is not adequate as it faces the problem of synonymy: multiple words with the same meaning, and the issue of which form to include in the translated text comes up. More sophisticated translation can be expensive. The best bet is probably to use machine translation methods to transform the text from one language to the other with all its advantages and drawbacks.

Semantic Similarity
Here one may want to exploit the meaning of the constituent tokens to assess the similarity of text chunks. This can take the form of exploiting Web content such as the Wikipedia and Categorized Text collections as is the case for Explicit Semantic Association (ESA) [3] or the corpora in which associations between words are sought as is the case for Latent Semantic Association (LSA) [14] or even the overlap of search engine results. One may also rely on web based knowledge infrastructure such as WordNet to estimate the similarity between words then generalize to text similarity. Using the Least Common Subsumer (LCS) based on WordNet and its variations are good examples of that. [17] uses Wikipedia categories of articles rather than articles themselves to compute semantic relatedness by representing a term by a list of articles containing the term in their title. [8] represent semantic meaning as a hierarchical structure derived from the Wikipedia category system as opposed to the Explicit Semantic Analysis approach which uses a flat vector representation in terms of

Wikipedia articles. [4] use knowledge based representation of texts that is based on the multilingual semantic network BabelNet and generate a graph to represent a document that accounts for multilingual synsets and the relationship between synsets and compare graphs to define similarity between documents. For the purposes of this paper, we will focus on ESA and its variations with an eye on using it or its variants for Arabic and CL similarity assessment. For CL settings, semantic similarity is the natural choice, if one is to avoid translation.

***Explicit Semantic Association (ESA)* [3]**

The ESA approach uses Wikipedia articles (or a sufficiently large finely categorized text corpus) as concepts to represent the meaning of text chunks as vectors with component values reflecting the associations of individual words with corpus concepts (articles, topical categories).

Under the variant of ESA, we use here each vocabulary word of the corpus Wi of language L is represented as an NL dimensional vector of concepts where N is the number of selected concepts, say Wikipedia articles or categories, in language L. Thus we have a matrix of |VL| rows and NL columns where |VL| is the size of the vocabulary in L and NL is the number of selected concepts (articles/categories) for L. The value of the jth component of the vector for the ith word wij is the tfi.df of word wi in the Wikipedia article number j. Variations on this weighting scheme that take into account factors like document size and category hierarchy features were discussed [6]. A word usually belongs to more than one concept (possibly with various weights) reflecting the different meanings of the word. One may truncate and look at the highest M concepts for a word (M << N) and zero the rest for simpler computations. An inverted table for the vocabulary is constructed to represent the matrix sorted by vocabulary words and for efficient storage one may keep only non-zero entries of the sparse matrix and to remove noise. The values for the word "bank" are likely to have larger values for concepts/articles talking about finance and articles talking about water bodies in case of article as concept representation and in categories dealing with water bodies and financial institutions in the case of category as a concept representation. The same reasoning can be applied to the Arabic word صف meaning class or queue.

The vector for an arbitrary sized text T is the sum of the vectors for its words (possibly normalized to account for text length variations) and thus has the same dimensions and structure as single word vectors; the format is independent of the text size. So given two texts T1 and T2 in Language L, possibly of different sizes, the similarity between these texts is the cosine similarity between the vector representations of T1 and T2. The vectors are likely to be sparse. The computational cost may be reduced by eliminating low frequency words and retaining concepts of reasonable quality, say of a particular length and link count: as important quality indicators.

On the surface of it, the vectors are language specific by the virtue of the concepts being Wikipedia Language specific. The size and composition of different Wikipedias vary a lot in terms of article numbers and quality. The number of articles needed is not a problem since Wikipedias in most languages meet the 100 K count needed for this technique to work for a single language. Of course, one has to worry about the quality and coverage to make sure that the representations adequately and correctly cover the different meanings of the language vocabulary terms.

Cross-Lingual Explicit Semantic Association (CL-ESA) [6, 16]

For Cross Lingual similarity assessment, one may need to translate one text into the language of the other to be able to compare the vectors representing both. However, this is likely to involve machine translation issues that may affect the quality of the results. A better option may be to use a common vector representation across the languages, say by using concepts shared in the Wikipedias of both languages (say Arabic and English) to form the vector representation. These concepts could be the parallel articles or common categories. One possible way to do that is to work with parallel Wikipedia portions: limit concepts to articles parallel in both languages. Each text is still processed in its own language but the representation is in the common article space induced by article parallelism. Basically, instead of translating the texts themselves we use the "translated" Wikipedia articles. To maximize dimensionality, one may try to increase the number of parallel articles by making all language links bidirectional for a pair of Wikipedias and employing transitivity through third languages [6]. That is an Arabic article A with a Spanish parallel article S will have E as the parallel English article when E is parallel to S [6]. Once these parallel articles are known, the vector representations for texts in both languages become compatible and text representations in both languages become comparable for similarity. The condition is that an adequate number of parallel articles in the pair of languages of interest be available with a reasonable distribution across topics to accommodate the various meanings of words and the diverse uses of these meanings. We may be talking about 100 K parallel articles for each pair as the acceptable range.

The Wikipedia language links may not be mature enough: articles may have links in only one direction the transitivity of such links may not work and there has been some effort to preprocess the Wikipedias to reconstruct the missing links [6].

The availability of sufficient parallel articles in the pair of languages of interest may be an issue. An added complication is that these parallel articles have to be of reasonable quality (e.g. length and number of links), but also one needs to make sure that they are really parallel, something that is not necessarily straightforward. It is our observation that many of the articles declared as parallel between Arabic and English are not really so. Good quality articles on "similar topics" may exist but being "not parallel" neutralizes their contribution to similarity. The approach ignores the wealth of knowledge that is not parallel but that may hold much info about word semantic associations.

We also believe that some of the parallel links can be misleading by pointing to empty or low quality articles, or even incompatible information. See for example the Wikipedia articles on Ramallah in Arabic, English and Russian with major variations in length and content. Combined with the need for a large dimensionality for the concept vectors there is a threat that such an approach may not work properly for many language pairs.

We have been working on an approach to CL-Similarity based on using concepts that are language independent and express word and text semantics in terms of these concepts. The mapping may still be done through the Wikipedia. The articles map texts to a Wikipedia induced category structure common to all languages. So rather than having Wikipedia articles serve as concepts, we employ select Wikipedia categories as concepts. As before, we compute word category vectors and then text category vectors

in all languages having the same dimensionality equal to the number of selected categories. So we still have to compute the inverted index for our vocabulary in each language over the selected common categories/concepts. The big advantage, in our view, is that categories are defined across languages and may be limited even when the Wikipedia itself continues to grow. What matters is having enough articles in a language Wikipedia spanning a sufficient number of categories to allow the construction of the inverted table for words in that language and the selection of categories used. One can opt for the high quality articles in each of the languages provided we account for size variations between languages in the vector-weighting scheme. For that, we started with the standard: excluding articles with less than 100 words and less than 5 links.

The big question is where do we get the working categories, how large they need to be and are they as good as concepts as the articles themselves? Our starting point is that we use the Wikipedia category system, we try to limit ourselves to a particular class of categories that are present in a sufficient number of reasonable quality articles and may avoid too general categories that are most likely to span too large a chunk of articles as non-discriminating. Our experiments, discussed later, show that more work is needed on category selection.

3 ESA Experiments

In this section, we describe the experiments we performed to measure semantic similarity.

3.1 ESA Through Wikipedia Articles as Concepts

Here, we did the following:

Infrastructure

- We selected a set of N Arabic Wikipedia articles (182,663 articles) and the set of words (vocabulary, V) in these articles. For each word, say w in V, we built a vector where dimension i is represented by the relative frequency (relfr: word frequency/total frequency) for w in Wikipedia article i (wA vector). Thus for each word w there will be an entry labelled by the article title (or ID) and has the relative frequency of w in that article as the value. This is done for each word to get a matrix MA of size |V|*N is generated.
- To compare two text chunks, we need to build an ESA vector for each text from matrix MA. To build an ESA vector for a text chunk T we sum up the vectors of each word occurrence in T. We could normalize by the max frequencies or T length.
- After building a vector for texts T1 and T2 the cosine similarity is calculated for these vectors as a measure of the similarity between T1 and T2.
- We also worked with Cleaned Vectors: the text vector is cleaned by keeping only the highest n values of components and resetting all other values to zero. We set n to be 300. We believe that such a truncation may help us get rid of the noise in the vectors and thus improve similarity between vectors.

Evaluation

In the monolingual setting, we are interested in match between the Arabic text chunk and its source document or in the text and the document parallel to its source for the CL case. Therefore, our main concern was on the position of the ideal document in the ranking resulting from similarity test. The average such ranking for all compared chunks was taken as the assessment of the overall performance. This can easily be converted into the standard Discounted Cumulative Gain (DCG) by taking the inverse of the log2 of the rank. The ideal solution should give an average of 1 by matching the test text/document with the corresponding document in the list. We also used the number of cases where the source article was ranked from 1 to 10 as another performance measure.

Experiments with Similarity of Arabic Text Chunks with Arabic Articles

We selected 500 Arabic Wikipedia articles with large word count (average word length 7191) and generated several (4) text packets of each article. The chunks were of size of 100, 200, 500 and 1000 words. We experimented with consecutive words from the start of the article and with randomly selected words denoted by start and random, respectively, in Table 1. Then we tested for similarity between each text packet and each of the 500 articles using ESA text vectors with articles as concepts.

- The articles were ranked by similarity to the given text packet. This is done for all packets of words sizes 100, 200, 500 and 1000 and for the two word selection approaches (start of article and random).
- We did the same with the ESA_Cleaned vectors (vectors truncated to the highest valued 300 dimensions).

Table 1 summarizes the results by giving the average rank of the words source article, the number and percentage of cases when the real source article had rank 1 and when it was ranked at most 10 for complete (Non cleaned) vectors. The same parameters are repeated for truncated (Cleaned) vectors.

Table 1. Arabic semantic text similarity using start of article consecutive word chunks

Selected words		Not cleaned vectors (based on article vectors)			Cleaned vectors (based on article vectors)		
Count	Position in article	Average source article rank	Rank 1 articles (#, %)	Rank 1–10 articles (#, %)	Average source article rank	Articles at rank 1 (#, %)	Articles with rank 1–10 (#, %)
100	Start	30.42	218, 43.6%	366, 73.2%	26.65	224, 44.8%	376, 75.2%
	Random	13.09	302, 60.4%	405, 81.0%	13.78	226, 45.2%	379, 75.8%
200	Start	23.75	279, 55.8%	410, 82.0%	20.92	272, 54.4%	399, 79.8%
	Random	5.89	392, 78.4%	464, 92.8%	7.62	283, 56.6%	430, 86%
500	Start	16.98	340, 68%	430, 86.0%	15.98	332, 66.4%	425, 85%
	Random	1.10	478, 95.6%	499, 99.8%	4.26	342, 68.4%	456, 91.2%
1000	Start	10.43	387, 77.4%	456, 91.2%	10.75	371, 74.25%	445, 89%
	Random	1.03	488, 97.6%	500, 100%	3.48	358, 71.6%	469, 93.8%
Full article	–	1	500, 100%	500, 100%	1	500, 100%	500, 100%

While start of article word selection is giving reasonably good results even for 200 words, the results are much better for random word selection. One can attribute the good results for the first words by the fact that they may reflect the article introduction/summary. Random words seem to be giving a better picture about the entire article. Cleaning vectors does not seem to give any returns and the results for that case are a little worse than for the original vectors. For random word selection, 500 words seem sufficient to represent the article. This looks quite interesting given that the average articles size is around 7000 words.

3.2 ESA Similarity Based on Wikipedia Tags (Categories) as Concepts

To use tags as concepts we did the following modifications on the ESA infrastructure:

- Instead of articles as concepts, we use Wikipedia categories with each selected category representing a dimension. The vector for each word (w^T vector) now represents the categories of articles in which that word appears. So words in a certain article A are processed by incrementing the value of the dimensions representing the categories of A by times frequency of the word in that article.
- Thus, each word will have a vector of tags with length |T|, where T is the set of selected Wikipedia tags, instead of a vector of Articles. A Matrix M^T is created with size of |V|*|T|.
- The vector for a text chunk is computed as before from word vectors as before. Similarity is computed as before the tag vectors of text chunks.

Table 2 summarizes the testing results for Arabic articles using tag based vectors. The results show that one could rely on tags as replacement for words within the single language (in this case Arabic).

Table 2. Using chunks of W random words from the selected articles based on tag-ESA

Word count	Not cleaned vectors (based on tag vectors)			Cleaned vectors (based on tag vectors)		
	Average source article rank	Rank 1 articles (#, %)	Rank 1–10 articles (#, %)	Average source article rank	Articles at rank 1 (#, %)	Articles with rank 1–10 (#, %)
100	33.82	169, 33.8%	316, 63.2%	34.36	169, 33.8%	309, 61.8%
200	19.50	261, 52.2%	389, 77.8%	18.95	258, 51.6%	385, 77%
500	3.49	386, 77.2%	472, 94.4%	3.59	380, 76%	472, 94.4%
1000	**1.73**	**448, 89.6%**	**488, 97.6%**	**1.59**	**439, 87.8%**	**490, 98%**

3.3 ESA Based Arabic Word Similarity for Articles and Tags as Concepts

So far, we reported on similarity tests between Arabic text chunks and full articles using ESA vectors. We employed the same approach to test similarity between Arabic word pairs using some of the gold standards reported in the literature [1]. Again to assess the performance we used ranking of word pair similarity. We assumed that the gold standard similarity score induced a ranking on the pairs and we assumed that the deviation from that ranking constitutes an aggregate measure of the success of a

similarity evaluation approach. We ran our experiments on two sets: one consists of 32 word pairs and another had 353, mostly translations of pairs originally developed for English. We used both tags and articles as concepts in different test runs. We experimented with various preprocessing parameters of the ESA like stemming, expansion, tag selection, and different weighting. The results were not encouraging. The best results of rank divergence we got for the 352 pairs was a little below 100 for both articles and tags as concepts as opposed to the 176 one could expect from a random placement. For the 32 word pairs we achieved about 6 in contrast to the expected 16 for the random. Table 3 below shows a summary of our results.

Table 3. Arabic word similarity pairs based on article and tag as concept vectors

Similarity test parameters articles/tags	Articles as concepts ESA		Tags as concepts ESA	
	32 pairs	353 pairs	32 pairs	353 pairs
Plain/plain	8.33	96.50	8.67	111.99
Stemmed/filtered	7.73	**90.62**	9.47	**99.63**
Expanded	**6.73**	**113.23**	**8.27**	**111.99**
New weight/new weight	**8.27**	**NA**	**9.87**	**NA**

We believe that the poor results of ESA for assessing similarity of word pairs are due to the inability of ESA to distinguish the different senses of the same word and that the word similarity for word pairs takes these senses into account, something that cannot be achieved through ESA. ESA is more likely to work for larger text chunks similarity providing better contexts for particular word senses. It is only that single word similarity may not be the best domain of ESA. It may be of value to check the performance of slightly larger, but still small, text chunks like paraphrases.

3.4 Cross Lingual ESA Similarity

Cross Lingual Similarity Infrastructure

In CL setting, we need to find similarity between text chunks in different languages. So for article A1 in language L1 and article A2 in language L2 we need to find the similarity between A1 and A2 based on their ESA representations. To do that we need a common map between articles (in the case of Article-ESA) and tags (in the case of Tag-ESA). In the case of Article-ESA we could take that to be the parallel articles of L1 and L2. So each dimension i for the Arabic word vector is an Article ID that has a parallel article in English and thus defines the same i dimension in ESA vector for English words. For the tags the same should apply with equivalent tags rather than parallel articles. In the Tag-ESA both Arabic and English words have vectors with the same dimensions. Each word vector is computed using the respective language Wikipedia articles with no parallelism restrictions.

In article ESA each Wikipedia article used in CL-ESA has an equivalent in the other language. The parallelism may not be close to equivalence, though. For tag-as-a-concept

ESA, the picture is mixed. Tagging is a community effort so no guarantees that the tag structure even for truly parallel articles are the same. This is the down side. The up side is that for tags no parallelism demands are placed on participating articles. The equivalences between the tags is straightforward to establish and is readily available. The problem is in the lack of consistency of tag assignments across languages that may reflect on the CL similarity testing results.

We completed our infrastructure by building ESA vectors for Arabic and English words, one suing parallel articles as concepts then using common tags as concepts. For the former we parsed only parallel articles and for the latter we placed no restriction on the articles parsed.

Now to compare (test for similarity) text chunks TA in Arabic with chunk TE in English we need the ESA vector of TA (sum of all words vectors TA computed from the Arabic Wikipedia) and the ESA vector of TE (sum of all words vectors in TE computed from the English Wikipedia) and then compute the cosine similarity. The fact that the IDs of the concepts involved in creating the word vectors are the same makes it possible to do a cosine similarity for vectors in different languages.

Cross Lingual Similarity Experiments

For CL-ESA testing, we performed experiments on the CL-ESA based on articles and tags as concepts. We selected 500 articles from the Arabic Wikipedia, with at least 1500 unique Arabic words each and which have equivalent (parallel) English versions within 500 words of the Arabic article count. So we had 500 Arabic articles and 500 parallel English Articles of comparable length. We ran each English article over the Arabic articles and ranked the Arabic articles by similarity to the English article being compared to find the rank (position) of the equivalent Arabic article. Again we used plain ESA and ESA_Cleaned vectors and did the testing for both Article-as-concept and Tags-As-Concept. Then we tried a preprocessing step involving the normalization through down-casing (UC vs. LC) and using log vs. relative frequency in the vectors of the English articles. The results are reported in Table 4 for Article-ESA and Tag-ESA.

Table 4. English articles similarity with Arabic articles based on article-as-concept and tag-as-concept vectors

Similarity test parameters CL_ESA+	Not cleaned vectors			Cleaned vectors		
	Average parallel article rank	Rank 1 parallel articles (#, %)	Rank 1–10 parallel articles 1–10 (#, %)	Average parallel article ranking	Rank 1 parallel articles (#, %)	Rank 1–10 parallel articles 1–10 (#, %)
relfr, UC: article tag	210.64	1, 0.2%	24, 4.8%	166.83	27, 5.4%	51, 10.2%
	244.07	4, 0.8%	20, 4.0%	245.4	3, 0.6%	2, 0.4%
log, UC: article tag	83.32	69, 13.8%	171, 34.2%	19.43	194, 29.8%	360, 72.0%
	91.53	21, 4.2%	112, 22.4%	69.05	60, 12.0%	198, 39.6%
relfr, LC: article tag	**99.73**	**95, 19%**	**206, 41.2%**	66.12	157, 31.4%	291, 58.2
	138.64	26, 5.2%	93, 18.6%	132.58	18, 3.6%	84. 16.8%
log, LC: article tag	113.07	16, 3.2%	49, 9.8%	**6.51**	**249, 49.8%**	**445, 89%**
	144.40	16, 3.2%	62, 12.4%	137.87	31, 6.2%	120, 24.0%

While Article-as-concept ESA seems to have performed well, reaching close to 90% for the 1–10 placement result, one can easily observe a major weakness in the results for Tags-ESA. Vector cleaning seems to have improved the results for both tests, but more so for the first case. Our explanation is that the noise introduced in the CL_ESA processing that may vary from one language to another is removed in the cleaning process. In the single language case one may assume that the noise is equally present on all vectors and thus has minimal effect on the results. More importantly, the tag-as-concept approach seems not to be good enough for the cross lingual case. We believe that this has to do with the type of tags we used and that the tag system may not be consistent as should be in the Arabic Wikipedia. It may be the case that more careful tag selection will produce better results. Of course the issue is not only to get the better results (though still below article) but the ease with which the results can be expanded to other languages without the need to work with the scarce parallel resources. We need only to have parallel tags, something that isn't as demanding as parallel articles. One of our explanations currently being tested is that the tag assignment process may not be consistent across languages and that some sort of homogenization is needed if one is to get reasonable results from this approach. We are continuing to investigate this issue.

4 Conclusions and Future Work

We reported on a series of experiments we performed to test for Cross Lingual similarity. Our approach was based on Explicit Semantic Association (ESA) and used Wikipedia as the underlying structure. The results were mixed based on the concepts used and the work is still ongoing. One of our conclusions is that the tags vectors are not working as good as the standard ESA with some preprocessing. We need further experimentation to see if that can be improved based on more cleaning and better category selection. The standard ESA on the other hand seems to be giving reasonable results, though not necessarily as good as reported for other languages. The quality of the Arabic Wikipedia may be one of the contributors to this and a possible direction of future work is to see if a better selection of articles can help improve the results. We are currently investigating the effect careful selection of tags on the performance of the system. We are also investigating the effects of combining article and tag representations on the system performance. We would like also to study the possible use of deep learning including neural nets [6] in our approach. We will also study the computational costs of the used methods and the practicality of their utilization.

Acknowledgement. The authors would like to thank Birzeit University which supported this work under a University research grant. They would like to thank the three anonymous referees for their valuable input.

References

1. Azmi-Murad, M., Martin, T.: Asymmetric word similarities for information retrieval, document grouping and taxonomic organization. In: Proceedings of EUNITE 2004 - Aachen, Germany, pp. 277–282 (2004)
2. Barrón-Cedeño, A., Paramita, M.L., Clough, P., Rosso, P.: A comparison of approaches for measuring cross-lingual similarity of Wikipedia articles. In: de Rijke, M., Kenter, T., de Vries, A.P., Zhai, C., de Jong, F., Radinsky, K., Hofmann, K. (eds.) ECIR 2014. LNCS, vol. 8416, pp. 424–429. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-06028-6_36
3. Gabrilovich, E., Markovitch, S.: Computing semantic relatedness using Wikipedia-based explicit semantic analysis. In: Proceedings of the 20th International Joint Conference on Artifical Intelligence, pp. 1606–1611. Morgan Kaufmann Publishers Inc., San Francisco
4. Franco-Salvador, M., Rosso, P., Navigli, R.: A knowledge-based representation for cross-language document retrieval and categorization. In: Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2014, Gothenburg, Sweden, 26–30 April 2014
5. Freeman, A., Condon, S., Ackerman, C.: Cross linguistic name matching in English and Arabic: a "one to many mapping" extension of the Levenshtein edit distance algorithm. In: Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL, New York, pp. 471–478, June 2006
6. Gupta, P., Banchs, R., Rosso, P.: Continuous space models for CLIR. Inf. Process. Manage. **53**(2), 359–370 (2017)
7. Hayashi, Y., Luo, W.: Extending monolingual semantic textual similarity task to multiple cross-lingual settings. In: Proceedings of the 10th Language Resources and Evaluation Conference (LREC2016), 23–28 May 2016, Portorož–Slovenia (2016)
8. Liberman, S., Markovitch, S.: Compact hierarchical explicit semantic representation. In: Proceedings of IJCAI09 WS on User Contributed Knowledge and Artificial Intelligence, Pasadena, CA, July 2009
9. Mihalcea, R., Corley, C., Strapparava, C.: Corpus-based and knowledge-based measures of text semantic similarity. In: Proceedings of the 21st National Conference on Artificial Intelligence, pp. 775–780. AAAI Press, Boston (2006)
10. Metzler, D., Dumais, S., Meek, C.: Similarity measures for short segments of text. In: ECIR07 (2007)
11. Moreau, E., Yvon, F., Cappe, O.: Robust similarity measures for named entities matching. In: Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), Manchester, pp. 593–600, August 2008
12. Ponzetto, S., Strube, M.: Knowledge derived from Wikipedia for computing semantic relatedness. J. Artif. Intell. Res. JAIR **30**, 181–212 (2007)
13. Potthast, M., Stein, B., Anderka, M.: A Wikipedia-based multilingual retrieval model. In: Macdonald, C., Ounis, I., Plachouras, V., Ruthven, I., White, R.W. (eds.) ECIR 2008. LNCS, vol. 4956, pp. 522–530. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-78646-7_51
14. Resnik, P.: Semantic similarity in a taxonomy: an information-based measure and its application to problems of ambiguity in natural language. JAIR **11**, 95–130 (1999)
15. Rupnik, J., Muhic, A., Leban, G., Skraba, P., Fortuna, B., Grobenik, M.: News across languages - cross-lingual document similarity and event tracking. J. Artif. Intell. Res. **55**, 283–316 (2016)

16. Sorg, T., Cimiano, P.: Cross lingual information retrieval with explicit semantic analysis. In: Working Notes of the Annual CLEF, 2008 Workshop (2008)
17. Strube, M., Ponzetto, S.P.: Wikirelate! computing semantic relatedness using Wikipedia. In: AAAI, vol. 6 (2006)

Empirical Evaluation of Word Representations on Arabic Sentiment Analysis

Mourad Gridach[1(✉)], Hatem Haddad[2], and Hala Mulki[3]

[1] High Institute of Technology, Ibn Zohr University, Agadir, Morocco
m.gridach@uiz.ac.ma
[2] Department of Computer and Decision Engineering,
Université Libre de Bruxelles, Brussels, Belgium
Hatem.Haddad@ulb.ac.be
[3] Department of Computer Engineering, Selcuk University, Konya, Turkey
halamulki@selcuk.edu.tr

Abstract. Sentiment analysis is the Natural Language Processing (NLP) task that aims to classify text to different classes such as positive, negative or neutral. In this paper, we focus on sentiment analysis for Arabic language. Most of the previous works use machine learning techniques combined with hand engineering features to do Arabic sentiment analysis (ASA). More recently, Deep Neural Networks (DNNs) were widely used for this task especially for English languages. In this work, we developed a system called CNN-ASAWR where we investigate the use of Convolutional Neural Networks (CNNs) for ASA on 2 datasets: ASTD and SemEval 2017 datasets. We explore the importance of various unsupervised word representations learned from unannotated corpora. Experimental results showed that we were able to outperform the previous state-of-the-art systems on the datasets without using any kind of hand engineering features.

Keywords: Arabic language · Arabic sentiment analysis
Convolutional neural networks · Pretrained word representations

1 Introduction

Sentiment analysis [1], also known as opinion mining [2], is a Natural Language Processing (NLP) task that receives much attention these years where the main goal is to classify text to sentiment classes such as positive or negative, or more fine-grained classes such as very positive, positive, neutral, etc. In the last years, sentiment analysis plays an essential role where it helps to develop many online applications for customer reviews and public opinion analysis [2–5]. The classical sentiment analysis systems focus on classical opinion such as binary classification (positive or negative), while others developed systems for multiple categories such as six basic emotions (anger, happiness, fear, sadness, disgust, and surprise) [6]. Sentiment systems can then be used to identify sentiment categories from texts.

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 147–158, 2018.
https://doi.org/10.1007/978-3-319-73500-9_11

Most of the previous work on sentiment analysis developed systems for English language because resources are publicly available for the community. For Arabic language, there has been less progress, but in the last few years, more resources are freely available for the Arabic language community especially after integrating the Arabic sentiment classification as one of the shared tasks in SemEval workshop in 2016. Most of the previous work in ASA used hand engineering features [7] or machine learning approaches such as SVM combined with features such as morphological features, POS tagging, etc. [8,9]. More recently, [10] explored different deep learning architectures to do Arabic sentiment classification.

In recent years, deep neural networks are widely used machine learning models. They have shown large success by achieving state-of-the-art results in various NLP applications such as dependency parsing [11], language modeling [12], question answering [13,14], speech recognition [15] and machine translation [16,17]. In addition, deep neural networks are widely used for sequence modeling tasks such as Named Entities Recognition (NER), Part-of-Speech (POS) tagging, recurrent neural networks (RNN) and obtained state-of-the-art in various languages such as English [18], German [19], Italian [20] and Arabic [21]. Various RNN models, like Long-Short Term Memory (LSTM) [22,23] and Gated Recurrent Unit (GRU) [16], have shown success in modeling sequential data like speech recognition [15] and POS tagging [24].

In this paper, we present CNN-ASAWR (Convolutional Neural Networks for Arabic Sentiment Analysis using Word Representations), our system uses Convolutional Neural Networks (CNN) as a deep learning model for ASA. Originally invented by [25] for pattern recognition and rediscovered after that by [26], convolutional neural networks are widely used mainly in various applications in computer vision where they obtained the state-of-the-art results, as well as natural language processing. In image classification, CNN based models (AlexNet, ZF Net, GoogLeNet, VGGNet and ResNet) won the ImageNet competition since 2012 [55]. Other computer vision applications using CNN are object detection [27,28], image segmentation [29], image captioning [30,31] and more recently, they are used in self driving cars [14,32].

In NLP, CNN received less attention from the NLP community compared to computer vision. More recently, CNNs are used in many NLP applications and have been shown to be effective in NER [18,33], sentence modelling [34], search query retrieval [35], semantic parsing [36] and more recently in machine translation [37] and text summarization [38].

In this paper, we present a deep learning model based on convolutional neural network for ASA. We explored the use of different word representations trained on unannotated corpora. We investigate three main word representations: Stanford Glove vectors, Skip-gram (SG) model and Continuous bag of words (CBOW) model. These Arabic word embeddings developed by [39] are publicly available for the community. The final architecture of our model can be described as a CNN trained on top of word representations. Despite little tuning of hyperparameters, our model achieves excellent results on two datasets, suggesting that

the pretrained word representations are universal feature extractors that can be used for various classification tasks.

Experiments on SemEval 2017 and ASTD dataset that are the two largest available datasets for the community showed that we were able to get the state-of-the-art results by outperforming the previous best system on ASTD dataset that uses SVM classifier with features by a large margin. In addition, CNN-ASAWR outperforms the winner system of the SemEval 2017 shared task on ASA.

The rest of this paper is structured as follows. In Sect. 2, we discuss the related work done in Arabic sentiment analysis and in Sect. 3 we present our ASA approach which is based on CNN trained on pretrained word representations. The experimental results will be presented in Sect. 4. Finally, we present the conclusion with the future work in Sect. 5.

2 Related Work

In this section, we present an overview of the most dominant approaches in ASA. In general, most of the previous Arabic sentiment analysis systems used machine learning approaches where some systems used supervised methods while others used unsupervised methods. Furthermore, these systems combined the previous machine learning approaches with features to perform the Arabic sentiment classification task.

[40] built a system where they focused on conducting sentiment classification at document level. The authors developed a method called EWGA (Entropy Weighted Genetic Algorithm), which is a hybridized genetic algorithm that incorporates the information-gain heuristic for feature selection. This method allowed the authors to improve performance and get a better assessment of key features. They evaluate this method on a benchmark movie review dataset and U.S. and Middle Eastern Web forum postings. The experimental results using EWGA with SVM indicate high performance levels, with accuracies of over 91% on the benchmark dataset as well as the U.S. and Middle Eastern forums.

[56] performed sentence-level sentiment classification for MSA. After a series of experiments, the authors discovered that the appearance of a positive or negative adjective, based on their lexicon, is the most influential feature. [7] developed a sentiment analysis system for both Modern Standard Arabic (MSA) news articles and dialectal Arabic microblogs from Twitter. Their model used many features such as stemming, part-of-speech tagging and tweet specific features. Finally, they extend the Arabic lexicon using Arabic-English phrase tables by adopting a random graph walk approach.

[8] developed a system called SAMAR for ASA based on SVM to do classification. This system uses lots of features such as morphological features: word forms and POS tagging, standard features such as UNIQUE (Q) feature and Polarity Lexicon (PL), dialectal Arabic features and lastly genre specific features (Gender, User ID, and Document ID).

[9] developed an ASA system called iLab-Edinburgh, using a hybrid approach where the development of their system passes through two stages: the first stage

consists of training a set of linear models on lexicon-based word-lemma unigrams. In the second stage, they experimented with different lexica for training the LR models. Their system attained the best performance at a Kendall score of 0.5362. This system was the winner of the Arabic Twitter Task 7 in SemEval 2016.

More recently, [10] explored different deep learning architectures to do Arabic sentiment classification. They used Deep Belief Networks and Deep Auto Encoders combined with features based on an Arabic Sentiment Lexicon. In addition, this system uses other standard lexicon features. To tackle the lack of context handling in the last system, they used another deep learning model called Recursive Auto Encoder (RAE). The experimental results showed that RAE model outperforms all the other models by a large margin of around 9%. RAE model takes advantage from semantic context and parsing order of words. In addition, RAE didn't use any no lexicon, and also no special features, but only raw words as input.

3 Our ASA Approach

In this section, we present the main ideas behind our approach for ASA. We begin by introducing the different word representations used in this paper. After that, we do a quick review to the Arabic sentiment datasets used to evaluate our model. Then, we present the main convolutional neural network architecture used on the top of word vectors. Lastly, we show through a series of experiments the importance of using weight normalization for training our neural network to do ASA.

3.1 Word Representations

The research in representations of words as continuous vectors has a long history where many ideas were proposed [41,42]. More recently, [43] proposed a model architecture based on feedforward neural networks for estimating neural network language model. The most popular model for word representations was developed by [44] called word2vec where they used either of two model architectures to produce a distributed representation of words: continuous bag-of-words (CBOW) model or Skip-Gram (SG) model. Another popular model for word representations developed by [45] called "GloVe" (Global Vectors). The main difference between this model and word2vec models is the representations of a word in vector space: word2vec models use a window approach while GloVe uses the global statistics of word-word co-occurrence in the corpus to be captured by the model.

[46] used word embeddings features for English dependency parsing where they employed flat (non-hierarchical) cluster IDs and binary strings obtained via sign quantization of the vectors. For chunking, [47] showed that adding word embeddings allows the English chunker to increase its F1-score. [48] showed that adding word embeddings as features for English part-of-speech (POS) tagging task helped the model to increase its performance. [49] argued that using word embeddings in parsing English text improved the system performance.

3.2 ASA Datasets

We have examined the performance of the proposed model on MSA/multi-dialectal textual content. Experiments were conducted on two datasets manually annotated for positive, negative or neutral polarity, they are:

- Arabic Sentiment Tweets Dataset (ASTD): A large sized-dataset of 10006 MSA/dialectal tweets collected and annotated by [50]. In our experiments, we have considered objective tweets as neutral ones. In addition, the set adopted for training was unbalanced.
- SemEval: Represents the Arabic dataset provided for the shared Task 4 entitled "Sentiment Analysis in Twitter" in the international contest of SemEval-2017 [51]. It is a medium-sized dataset of 3355 tweets written in MSA and several Arabic dialects (Egyptian, Syrian, etc.).

Each dataset has been divided into a training set to train the model, a development set to tune it and a test set for evaluation. The detailed statistics of the polarity distribution across these sets are reviewed in Tables 1 and 2.

Table 1. The polarity distribution across positive, negative and neutral classes in training sets

Dataset	Size	Positive	Negative	Neutral
SemEval	2684	521	1014	1149
ASTD	8005	613	1280	6112

Table 2. The polarity distribution across positive, negative and neutral classes in test sets

Dataset	Size	Positive	Negative	Neutral
SemEval	671	222	128	321
ASTD	2001	186	404	1411

3.3 Convolutional Neural Networks Architecture

Convolutional neural networks are powerful machine learning models that shown good performance in many NLP tasks such NER, machine translation, text summarization and language modeling. In this work, we use CNN for ASA. The main architecture of our CNN is depicted in Fig. 1 where we are classifying the Arabic sentence شاهد زبون فرنسي غاضب يقتحم متجر شركة ابل تكنولوجيا which means in English: An angry French customer breaks into the Apple Tech store.

In the first step, we initialize each word in the sentence with pretrained word representations from one of the three models: Glove, Skip-Gram or CBOW. Then, we feed these word vectors to the convolutional neural network as input sentences which will be used in the convolutional layer and max pooling as the next step. Finally, the sentence is classified as "Positive", "Negative" or

"Neutral" which can be seen in the last layer (FC with Softmax) containing three output classes. Our CNN architecture is a variant of [33] model where the authors used it to perform part-of-speech tagging, chunking, named entity recognition, and semantic role labeling.

Given a tweet T with length n where we add padding whenever it is necessary for the model, T is represented as the following:

$$v_1^n = v_1 \oplus v_2 \oplus ... \oplus v_n \tag{1}$$

where $v_i \in R^k$ represents the word vector of the $i-th$ word in the sentence S with k is the dimension of the word vector and $\oplus$ represents the concatenation operator. We use successive filters $w \in R^{mk}$ to obtain multiples feature map. Each filter is applied to a window of m words to get a single feature map:

$$F_i = f(w.v_i^{i+m-1} + b) \tag{2}$$

where b is the bias and f is the non-linearity where we used ReLU (Rectified Linear Unit). The general form of a feature map is the following:

$$F = [F_1, F_2, ..., F_{n-m+1}] \tag{3}$$

where $F \in R^{n-m+1}$. In the next step, we applied a max-over-time pooling operation [33] to the feature map and take the maximum value. The results are feed to a fully connected softmax layer to get probabilities over the tweets (Fig. 1).

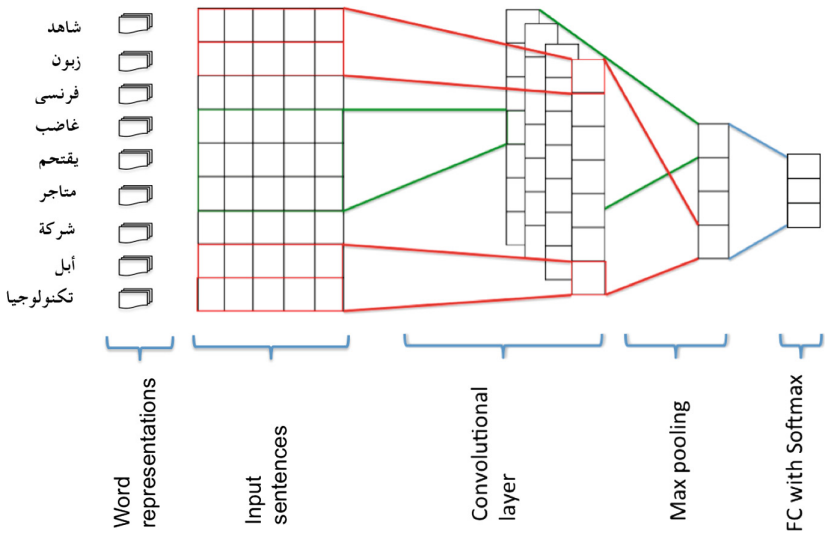


Fig. 1. The system architecture. The word vectors are initialized with pretrained word representations from one of the three models: Glove, Skip-Gram or CBOW. We feed these word vectors to the convolutional neural network as input sentences. Here, the sentences will be classified as "Positive", "Negative" or "Neutral" which can be seen in the last layer (FC with Softmax)

We repeat the same operation to many filters to get the next convolutional layer. To fight overfitting and prevent co-adaptation of hidden units, we use Dropout method [52] as the main method for regularization in our model.

4 Experiments

In this section, we begin by presenting the details about the training process and hyperparameters used by our model. After that, we give the experimental results obtained by using our model based on convolutional neural networks, pretrained word representations and weight normalization.

4.1 Training Details

To train our convolutional neural network, we use backpropagation algorithm to update our model parameters on every training example using stochastic gradient descent (SGD) over mini-batches. Several methods have been proposed to enhance the performance of SGD, such as Adadelta [53] or Adam [54]. Although we observe faster convergence using these methods, none of them perform as well as SGD.

For other specific CNN hyperparameters, we note that we used the SemEval dev set to choose the best ones. We used ReLU (Rectified Linear Unit) as the non-linearity function, which proved its efficiency. It should be noted that we explored other non)linearity functions such *Sigmoid* function, *Tanh* function, and others, but none of them were able to be efficient than ReLU. To overcome the problem of model overfitting, we use a dropout rate of 0.5. For filter windows, we set their values to 3, 4 and 5 with 100 feature maps each. We choose the size of the mini-batches to be 50. For the ASTD dataset, it should be noted that we randomly select 10% of the training data to be the dev set.

4.2 Results

The experiments were carried out on ASTD and SemEval 2017 datasets. We begin by presenting the effect of using pretrained word representations compared to the random word vectors on ASA.

Table 3 presents the results of our CNN trained on the top of different pretrained word representations on both ASTD and SemEval datasets. The results are compared to the random word vectors.

From the results showed in Table 3, we conclude that the best F1-score was obtained by using the pretrained word vectors from the CBOW model. Furthermore, we observed that using pretrained word embeddings with all models (Glove, Skip-Gram and CBOW) allowed the model to improve significantly its performance over the model used just randomly initialized word vectors.

Firstly, we improved the F1-score by 7.68 points and 4.13 points respectively on the ASTD and SemEval datasets when we used GloVe pretrained word embeddings. Secondly, we obtained 7.87 points and 4.35 points improvement in

Table 3. Results using three pretrained word vectors: Glove, Skip-Gram and CBOW models and the comparison with randomly initialized word vectors.

Models	F1-score	
	ASTD	SemEval
CNN + Random word vectors	64.15%	58.54%
CNN + Glove vectors	71.83%	62.67%
CNN + Skip-Gram model	72.02%	62.89%
CNN + CBOW model	**72.14**%	**63.00**%

F1-score on respectively ASTD and SemEval datasets when use pretrained word embeddings from Skip-Gram model. Lastly, we were able to improve the F1-score by 7.99 points and 4.46 points on respectively ASTD and SemEval datasets by using pretrained word embeddings from the CBOW model.

The last results are consistent with the fact that these pretrained word vectors are universal feature extractors that shown an important results in different NLP applications such named entity recognition and Part-of-speech tagging. To compare our model with the previous state-of-the-art systems, we will use the best results obtained from the combination of CNN and CBOW model.

In the next stage, we compare CNN-ASAWR with the previous best models. We note that for SemEval 2017, until now, we were not able to see the description of the winner system in this shared task competition about Arabic sentiment classification. Table 4 shows the comparison between our model and the winner system called NileTMRG. We were able to outperform their system by 0.95 points in F1-score. Table 5 presents the comparison between our model and [50] system. Their system is based on SVM classifier. Related to the results, we were able to outperform their system by a large margin (9.54 points in F1-score).

As far as we know, we are the first to explore the effect of pretrained word representations on ASA. By combining the power of deep convolutional neural

Table 4. Comparison between CNN-ASAWR and NileTMRG which is the winner on SemEval 2017.

Model	F1-score
Model F1-score NileTMRG	61%
CNN-ASAWR	**63**%

Table 5. Comparison between CNN-ASAWR and previous best system on ASTD dataset.

Model	F1-score
Nabil et al. (2015)	62.60%
CNN-ASAWR	**72.14**%

networks with word representations learned from unannotated corpora, we were able to obtain state-of-the-art on two publicly available datasets without resorting to any kind of hand engineering features.

5 Conclusion and Future Work

This paper presents a deep convolutional neural network for ASA that provide the best sentiment classification results on two Arabic sentiment datasets: ASTD and SemEval 2017. We took advantage of using convolutional neural networks in ASA since this model works well in computer vision applications such as image captioning, image classification and others, where it achieved state-of-the-art results. The model truly is end-to-end which means that it does not rely on hand engineering features considered as time consuming.

A key aspect of our model is that it explores the power of deep convolutional neural networks trained on the top of pretrained word representations trained on unannotated corpora. We investigated different models of word representations (CBOW, Skip-Gram and Glove) and compare the results with randomly initialized word vectors. Experimental results showed that the pretrained word representations largely outperformed the random ones with a good margin. This is consistent to the previous results on English sentiment analysis and also to other NLP applications such NER, POS and chunking and confirm the general idea that these pretrained word vectors are universal feature extractors.

In the future work, we will test our model on more Arabic datasets, especially from Arabic dialects such Moroccan, Tunisian, and others, in order to show its stability to various dialects. Furthermore, we will investigate how adding to our model with more features such as part-of-speech tags features, morphological analysis features, and other would influence the results. Finally, we will explore the use of another pretrained word embeddings trained on multilingual Arabic dialects and compare the results with the word embeddings used in this work.

References

1. Nasukawa, T., Yi, J.: Sentiment analysis: capturing favorability using natural language processing. In: Proceedings of the 2nd International Conference on Knowledge Capture, pp. 70–77 (2003)
2. Liu, B.: Sentiment analysis and opinion mining. Synthesis lectures on human language technologies **5**(1), 1–167 (2012)
3. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Found. Trends Inf. Retr. **2**(1–2), 1–135 (2008)
4. Calvo, R.A., D'Mello, S.: Affect detection: an interdisciplinary review of models, methods, and their applications. IEEE Trans. Affect. Comput. **1**(1), 18–37 (2010)
5. Feldman, R.: Techniques and applications for sentiment analysis. Commun. ACM **56**(4), 82–89 (2013)
6. Ekman, P.: An argument for basic emotions. Cogn. Emotion **6**(3–4), 169–200 (1992)

7. Mourad, A., Darwish, K.: Subjectivity and sentiment analysis of modern standard Arabic and Arabic microblogs. In: Proceedings of the 4th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, pp. 55–64 (2013)
8. Abdul-Mageed, M., Kuebler, S., Diab, M.: SAMAR: a system for subjectivity and sentiment analysis of social media Arabic. In: Proceedings of the 3rd Workshop on Computational Approaches to Subjectivity and Sentiment Analysis (WASSA), ICC Jeju, Republic of Korea (2012)
9. Refaee, E., Rieser, V.: iLab-Edinburgh at SemEval-2016 Task 7: a hybrid approach for determining sentiment intensity of Arabic Twitter phrases. In: Proceedings of SemEval-2016, pp. 474–480 (2016)
10. Al Sallab, A.A., Baly, R., Badaro, G., Hajj, H., El Hajj, W., Shaban, K.B.: Deep learning models for sentiment analysis in Arabic. In: ANLP Workshop 2015, p. 9 (2015)
11. Ballesteros, M., Dyer, C., Smith, N.A.: Improved transition-based parsing by modeling characters instead of words with LSTMs. arXiv preprint arXiv:1508.00657 (2015)
12. Mikolov, T., Kombrink, S., Burget, L., Černocký, J., Khudanpur, S.: Extensions of recurrent neural network language model. In: 2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 5528–5531 (2011)
13. Xiong, C., Merity, S., Socher, R.: Dynamic memory networks for visual and textual question answering. arXiv, 1603 (2016)
14. Chen, D., Fisch, A., Weston, J., Bordes, A.: Reading Wikipedia to Answer Open-Domain Questions. arXiv preprint arXiv:1704.00051 (2017)
15. Graves, A.: Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850 (2013)
16. Cho, K., Van Merriënboer, B., Bahdanau, D., Bengio, Y.: On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259 (2014)
17. Luong, M.T., Manning, C.D.: Achieving open vocabulary neural machine translation with hybrid word-character models. arXiv preprint arXiv:1604.00788 (2016)
18. Ma, X., Hovy, E.: End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. arXiv preprint arXiv:1603.01354 (2016)
19. Reimers, N., Eckle-Kohler, J., Schnober, C., Kim, J., Gurevych, I.: Germeval-2014: nested named entity recognition with neural networks. In: Proceedings of the KONVENS GermEval Shared Task on Named Entity Recognition, Hildesheim, Germany (2014)
20. Bonadiman, D., Severyn, A., Moschitti, A.: Deep neural networks for named entity recognition in Italian. In: Proceedings of the Second Italian Conference on Computational Linguistics CLiC-it 2015, p. 51, December 2015
21. Gridach, M.: Character-aware neural networks for Arabic named entity recognition for social media. In: WSSANLP 2016, p. 23 (2016)
22. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
23. Gers, F.A., Schmidhuber, J., Cummins, F.: Learning to forget: continual prediction with LSTM. Neural Comput. **12**(10), 2451–2471 (2000)
24. Huang, Z., Xu, W., Yu, K.: Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991 (2015)
25. Fukushima, K.: Neocognitron: a hierarchical neural network capable of visual pattern recognition. Neural Netw. **1**(2), 119–130 (1988)

26. LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proc. IEEE **86**(11), 2278–2324 (1998)
27. Ren, S., He, K., Girshick, R., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: Advances in Neural Information Processing Systems, pp. 91–99 (2015)
28. Girshick, R., Donahue, J., Darrell, T., Malik, J.: Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 580–587 (2014)
29. Farabet, C., Couprie, C., Najman, L., LeCun, Y.: Scene parsing with multiscale feature learning, purity trees, and optimal covers. arXiv preprint arXiv:1202.2160 (2012)
30. Vinyals, O., Toshev, A., Bengio, S., Erhan, D.: Show and tell: a neural image caption generator. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3156–3164 (2015)
31. Karpathy, A., Fei-Fei, L.: Deep visual-semantic alignments for generating image descriptions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3128–3137 (2015)
32. Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L.D., Monfort, M., Muller, U., Zhang, J., Zhang, X.: End to end learning for self-driving cars. arXiv preprint arXiv:1604.07316 (2016)
33. Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., Kuksa, P.: Natural language processing (almost) from scratch. J. Mach. Learn. Res. **12**, 2493–2537 (2011)
34. Kalchbrenner, N., Grefenstette, E., Blunsom, P.: A convolutional neural network for modelling sentences. arXiv preprint arXiv:1404.2188 (2014)
35. Shen, Y., He, X., Gao, J., Deng, L., Mesnil, G.: Learning semantic representations using convolutional neural networks for web search. In: Proceedings of the 23rd International Conference on World Wide Web, pp. 373–374 (2014)
36. Yih, W.T., Toutanova, K., Platt, J.C., Meek, C.: Learning discriminative projections for text similarity measures. In: Proceedings of the Fifteenth Conference on Computational Natural Language Learning, pp. 247–256 (2011)
37. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.N.: Convolutional Sequence to Sequence Learning. arXiv preprint arXiv:1705.03122 (2017)
38. Paulus, R., Xiong, C., Socher, R.: A deep reinforced model for abstractive summarization. arXiv preprint arXiv:1705.04304 (2017)
39. Zahran, M.A., Magooda, A., Mahgoub, A.Y., Raafat, H., Rashwan, M., Atyia, A.: Word representations in vector space and their applications for Arabic. In: Gelbukh, A. (ed.) CICLing 2015. LNCS, vol. 9041, pp. 430–443. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-18111-0_32
40. Abbasi, A., Chen, H., Salem, A.: Sentiment analysis in multiple languages: feature selection for opinion classification in web forums. ACM Trans. Inform. Syst. (TOIS) **26**(3), 12 (2008)
41. Hinton, G.E., Rumelhart, D.E. and Williams, R.J.: Learning internal representations by back-propagating errors. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, 1 (1985)
42. Hinton, G.E., McClelland, J.L., Rumelhart, D.E.: Distributed representations. In: Parallel Distributed Processing: Explorations in the Microstructure of Cognition (1986)
43. Bengio, Y., Ducharme, R., Vincent, P., Jauvin, C.: A neural probabilistic language model. J. Mach. Learn. Res. **3**, 1137–1155 (2003)

44. Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., Dean, J.: Distributed representations of words and phrases and their compositionality. In: Advances in Neural Information Processing Systems, pp. 3111–3119 (2013)
45. Pennington, J., Socher, R., Manning, C.D. Glove: global vectors for word representation. In: EMNLP, vol. 14, pp. 1532–1543 (2014)
46. Hisamoto, S., Duh, K., Matsumoto, Y.: An empirical investigation of word representations for parsing the web. In: Proceedings of ANLP, pp. 188–193 (2013)
47. Turian, J., Ratinov, L., Bengio, Y.: Word representations: a simple and general method for semi-supervised learning. In: Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, pp. 384–394 (2010)
48. Huang, F., Ahuja, A., Downey, D., Yang, Y., Guo, Y., Yates, A.: Learning representations for weakly supervised natural language processing tasks. Comput. Linguist. **40**(1), 85–120 (2014)
49. Bansal, M., Gimpel, K., Livescu, K.: Tailoring continuous word representations for dependency parsing. In: ACL, vol. 2, pp. 809–815 (2014)
50. Nabil, M., Aly, M.A., Atiya, A.F.: ASTD: Arabic sentiment tweets dataset. In: EMNLP, pp. 2515–2519 (2015)
51. Rosenthal, S., Farra, N., Nakov, P.: SemEval-2017 task 4: sentiment analysis in Twitter. In: Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval 2017. Association for Computational Linguistics, Vancouver, Canada (2017)
52. Srivastava, N., Hinton, G., 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)
53. Zeiler, M.D.: ADADELTA: an adaptive learning rate method. arXiv preprint arXiv:1212.5701 (2012)
54. Kingma, D., Ba, J.: Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
55. Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A.C., Fei-Fei, L.: ImageNet large scale visual recognition challenge. Int. J. Comput. Vis. (IJCV) **115**(3), 211–252 (2015)
56. Abdul-Mageed, M., Diab, M.T., Korayem, M.: Subjectivity and sentiment analysis of modern standard Arabic. In: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers, vol. 2, pp. 587–591, Association for Computational Linguistics (2011)

A Survey of Extractive Arabic Text Summarization Approaches

Samira Lagrini[1(✉)], Mohammed Redjimi[2], and Nabiha Azizi[1]

[1] Labged Laboratory, Computer Science Department,
Badji Mokhtar University, PO BOX 12, Annaba 23000, Algeria
samiraboite@yahoo.fr, nabihalll@yahoo.fr
[2] Université 20 Août 1955, Skikda 21000, Algeria
redjimimed@yahoo.fr

Abstract. Automatic text summarization is an important research area originating from the late 50's but not losing its celebrity until now. Over the past half a century, automatic text summarization has seen a great interest especially in English language. However, in Arabic language, few works have been done in this field. This paper intends to survey the most relevant approaches in Arabic text summarization, giving special emphasis to extractive techniques. The limitation of these approaches and the main difficulties faced when dealing with such application are also discussed. Special attention is devoted to the peculiarities of Arabic language, which have posed challenges to the task of summarization.

Keywords: Arabic text summarization · Clustering
Rhetorical Structure Theory (RST) · Machine learning
Graph theory · Text entailment · Extractive approaches
Summary evaluation

1 Introduction

Text summarization is an essential application of Natural Language Processing (NLP). It is an imperative and timely tool for understanding text information. The objective of automatic text summarization is abridging texts into briefer version, conserving its overall meaning [1]. This allows the reader to decide whether a document contains required information with minimum effort. There is no doubt about the importance of such application. For example, it could be used as an informative tool in search engine web pages to find the pertinent and required information [2].

According to [3], a summary is "a text produced from one or more texts, that conveys important information of the original texts and that is no longer than half of the original text(s) and usually significantly less than that".

Summarization systems can be categorized according to several characteristics: language, input, method output, generality…etc. (see Fig. 1). This enables summaries to be characterized by various properties [4]. For example, according to the degree of generality, a summary can be classified into generic or query driven summaries. Generic summary attempts to represent all relevant topics in the input document while

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 159–171, 2018.
https://doi.org/10.1007/978-3-319-73500-9_12

query driven summary depends on the user information need. We can also distinguish between single document summarization and multi-document summarization depending on the number of input documents to be summarized. Regarding the output, a summary can be either indicative or informative. Indicative summary is used to specify what topics are tackled in the input document. This will allow users to get an overall and a brief idea of the source text. Informative summary is intended to cover all topics addressed in the source text with further details.

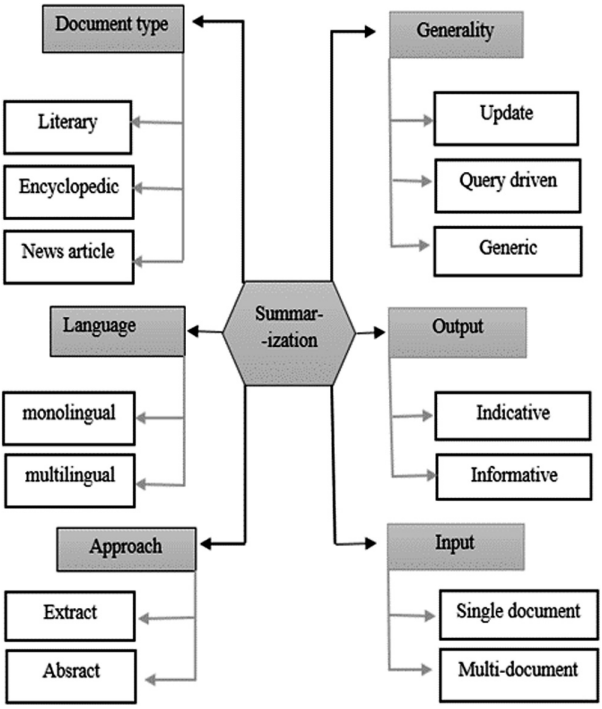


Fig. 1. Summarization taxonomy

Furthermore, we can talk about monolingual and multilingual summarizer. Monolingual summarization systems are designed to work with only one language and have the input document and the output summary in the same language, unlike multilingual summarizers, which cover more than one language.

Moreover, a broad difference is made between extract and abstract depending on the adopted approach. An extractive approach consists in selecting key sentences from the source document based on statistical and linguistic features, and concatenating them into a briefer form [1]. Abstractive approach differs mainly from extractive approach by providing summaries having some degree of inference about background knowledge not necessary presented in the original document [5]. In other words abstractive summarization means that, a new text is generated using the lexical, syntactical, semantic and rhetoric ingredients of the original text.

The goal of this paper is to survey the most salient extractive Arabic text summarization approaches.

The rest of this article is organized as follows: Sect. 2 gives an overview of some salient peculiarities of Arabic language. Section 3 focusses on summary evaluation issues. Then, Sect. 4 resumes the main proposed approaches for extractive Arabic text summarization. Section 5 explains the limitation of these approaches and the major challenges faced when dealing with such application. Finally, Sect. 6 concludes this paper.

2 Arabic Language Particularities

Arabic is the first language of more than 200 million persons through the world, and the official language of 21 countries [6]. Arabic language possesses specific peculiarities that make it distinctive, but at the same time, they pose several challenges to various Arabic natural language processing (ANLP) tasks, such as automatic summarization, sentence segmentation, and even word stemming. Some of these challenges include its complex morphology, the ambiguity, and its inflectional and derivational nature.

Regarding morphology, Arabic language is very rich and very complicated. Indeed, several words (sometimes more than ten) in Arabic can be formed using one single root, some patterns and some affixes. Affixed letters are very similar to root letters, which leads to several ambiguity. Thus, one single word could have diverse morphological features, as well as different POS. For instance, the word "فهم" can be tagged as a conjunction "ف" followed by the pronoun "هم" (they), or as a verb (to understand), or as a noun "فهم" (understanding).

Many reasons lead to this ambiguity. One salient reason is the lack of vowels that are only used in the holy Quran, and which are completely omitted in Modern Standard Arabic (MSA) written texts. Taking as an example, the word (علم) that can be read as (عَلّم/Ellama/he teaches), or as (عَلِمَ/Elima/he knew), or as (عِلْم/Elm/a science) or as (عَلَم/Elam/a banner)

Another possible reason is the omission of writing marks, like Hamza (ء) and dots on letters. Therefore, dissimilar words can be written in the same way. For instance omitting the two dots on (ــة) in the word (معلمة/a teacher), makes it exactly similar to (معلمه/his teacher). Similarly, omitting Hamza in the words (لأن/because) makes it identical to the verb (لان/it softened). This type of ambiguity causes serious problems in many ANLP tasks including word sense disambiguation, machine translation and even word stemming.

Furthermore, Arabic does not have capital letters [7], which affect the recognition of named entities in the annotation process. For example, the word "وفاء" can be annotated as a named entity, or as an Accusative of purpose of the verb "وفى" which means 'to honor'.

Finally, Arabic is a highly derivational and inflectional language [8]. Arabic words are generally composed of several morphemes. Thus, we can easily find one single word that can be represented by a complete statement. For instance, the word (أَنُلزِمُكُموها) represents a statement that means 'Shall we compel you to accept it' (see Fig. 2.)

Fig. 2. Example of Arabic inflection

In fine, it is to be pointed out here, that the aforementioned challenges make difficult ANLP tasks, which can probably explain the lack of publically available tools and resources for Arabic language.

3 Summary Evaluation

Assessing the quality of summary is a challenging task in the field of automatic texts summarization. Indeed, there is no sole "perfect" summary. Summaries written by different people can be different at the content level. Writing this type of documents requires a deep understanding of the text in order to identify ideas, style and arguments, which each person does differently.

Another factor behind this challenging task is the fact that evaluating summary requires a comparison with reference summaries [9]. This implies the existence of benchmark corpora that contains documents to be summarized and their reference summaries. Creating such benchmark corpora is an expensive and time-consuming task [10]. Moreover, several summaries can be appropriate for the same document, and even the same person can summarize the same document in different way over time [11].

Moreover, evaluation process itself is a great problem. Person evaluation is time-consuming [12], and provides unsteady evaluation score. To overcome these problems, automatic methods such as ROUGE [13] and AUTOSUMENG [14] have been introduced.

According to [15], Evaluation methods are classified into intrinsic and extrinsic methods.

In extrinsic methods, summaries should be evaluated based on their utility and ability to perform certain tasks, such as classifying documents, or using summaries instead of original documents in question/answer systems. A summary is then considered effective if it allows its reader to answer the questionnaire as well as other readers who have read the source text. Intrinsic methods evaluate summaries based on their properties and content. Intrinsic methods consist of comparing machine

summaries with expected output data, such as one or more reference summaries, or relevant sentences chosen by human subjects to be included in the summary.

ROUGE (Recall-Oriented Understudy for Gisting Evaluation) [13] is a prominent measure that involves the differences between words distribution. It consists of a package that includes various ROUGE measures, like ROUGE-L (Longest Common Subsequence), ROUGE-N (N-gram Co-Occurrence Statistics), ROUGE-W (Weighted Longest Common Subsequence), etc.

ROUGE is highly used by DUC (Document Understanding Conferences) ever since 2004. This measure is considered as a standard by the community, because of its strong correlation with manual notations.

Although, based solely on the content of summary, ROUGE suffers from numerous drawbacks related to its dependence on the units (N-grams) used for the calculation of the scores. The multi-word units as "United States of America" and relatively unimportant words such as "the", "but", etc. biased the number of co-occurrences. In addition, many preprocessing steps that rely on language dependent resources are required previously [9].

Other automatic methods are also used. AutoSummENG (Automatic Summary Evaluation based on N-gram Graphs) [14] has been introduced as a language independent evaluation method. The basic idea behind this method is to create at first an n-gram graph for the candidate summary as well as for reference summaries. Then, the average of the similarities between the candidate summary and each reference summary is calculated in order to evaluate the system. As a variation of AutoSummENG, the MeMoG (MergedModel Graph) [14] relies on one merged graph representing the references summaries to calculate its similarity with the candidate summary rather than using all graphs of reference summaries.

At the ACL 2013 Multi-Ling Workshop, NPowER (N-gram graph Powered Evaluation via Regression) [16] was added to the automatic evaluation methods. The authors used linear regressions to Combine AutoSummENG and MeMoG methods, and the evaluation process is formulated as a machine-learning problem. For more details about this method, see [16].

4 Arabic Text Summarization Approaches

This section describes the principal approaches proposed in the field of Arabic text summarization.

4.1 Discourse Theories

Rhetorical Structure Theory. The Rhetorical Structure Theory (RST) [17] is perhaps the most popular theory of discourse. In the RST framework, texts are represented by labeled trees, whose leaves correspond to atomic text segments, called elementary discourse units (EDUs), and internal nodes correspond to the rhetorical relations. Adjacent nodes in the tree structure are linked by rhetorical relations (causal, joint, manner, etc.) forming a discourse sub tree, which can then be subject to this linking. For more details about this theory, one should see [17].

The first employment of this theory on Arabic text summarization has been addressed by [18]. The authors suggested different techniques, algorithms and design patterns to be considered when developing Arabic summarizers based on RST.

Then, in [19] the rhetorical structure theory was also used for classifying Arabic security documents. The authors propose a technique that parses each paragraph in the document, build the rhetorical tree that represents its structure, and then determines the importance of each paragraph by examining the tree root. If the importance of the paragraph conforms to the user instruction, the classifier labels it with the required classification.

In [20], the authors propose a two-pass algorithm. In the first pass, RST is used to generate a primary summary. Therefore, a rhetorical analysis of a text is performed in order to generate all possible RS trees, upon which the primary summary is generated. In the second pass, each sentence within the primary summary is awarded a score based on word frequency and overlap with title keyword. To produce the final summary, sentences having the highest score are selected tacking into account the user compression ratio.

Other approaches provide a hybrid model like in [21]. The proposed model combines RST and vector space model (VSM). The model discovers at first the most important paragraphs based on semantic criteria, and then uses the VSM to rank these paragraphs based on the cosine similarity feature. Results revealed that combining VSM with RST improves the precision of the summary over employing RST only.

Segmented Discourse Representation Theory (SDRT). The Segmented Discourse Representation Theory (SDRT) proposed by [22] is a theory of discourse interpretation that seeks to combine two paradigms: discourse analysis and dynamic semantics. According to SDRT, a text is segmented into text units linked to each other via rhetorical relations, resulting into directed graphs called SDRS graphs. Unlike RST, in SDRT multiple discourse relations can link one discourse unit to adjacent or non-adjacent units. That is to say, several discourse relations can simultaneously link two text units in SDRT.

For the best of our knowledge, [23] addressed the first employment of this theory on Arabic text summarization. The authors tackle discourse analysis of Arabic documents following the SDRT framework. They explore how discourse structure can be exploited to produce indicative summaries. To this end, they design several algorithms that take as input the document discourse structure and generate as output a set of elementary discourse units, which will be used to produce the summary. To check the effect of discourse structure on producing indicative summaries, a comparison was made between the produced summaries and reference summaries, manually generated from two discourse annotated corpora following RST and SDRT framework. Results revealed that all discourse structure (graphs vs. trees) are very useful and can highly improve the results of automatic Arabic text summarization.

4.2 Cluster Based Approach

Many Arabic text summarization systems use clustering to generate a summary. For instance, [24] proposed an Arabic single and multi-document summarization approach

based on automatic sentence clustering and an adapted discriminate analysis method. Their system uses a clustering algorithm to group similar sentences into clusters. The proposed approach takes advantage of term's discriminate power to score sentences.

In the same context, [25] proposed a model based on document clustering and key phrase extraction. The model used a hybrid clustering (partitioning and k-means) to group Arabic documents into several clusters, then it extracts important key-phrases from each cluster. The model reached good results for single and multi-document summarization but no comparison with other systems was achieved.

Unlike the previous presented systems, [26] uses clustering to group words with the same root in the same cluster. The number of words in that cluster determines the weight of each word in the cluster. Then the score of each sentence is calculated based on several features. Sentences having the highest score are selected to be included in the final summary.

Finally, in [27], the authors investigate the use of clustering in Arabic multi-document summarization and for redundancy elimination. To this end, the authors conducted two experiments. In the first one, K-means algorithm is used to cluster sentences. More precisely, a number of sentences are selected randomly as the initial centroids, and then all sentences are assigned to the closest cluster based on their cosine similarity measure. To produce the summary, two selection methods are used: In the first method, the first sentence of each cluster is selected, while in the second one, all sentences in the biggest cluster are selected. For the second experiment, sentences selection is carried out before the clustering, and only the first sentence from each document and the most similar sentence are selected. Then, all the subsequent steps are similar to the first experiment. For evaluation, DUC-2002 dataset and an Arabic parallel translation version are used. Evaluation results are compared with the best five systems in the DUC 2002 competition. The proposed summarizers achieved the best scores when comparing ROUGE-1 results.

4.3 Machine Learning Based Approach

In the machine learning based approach, the summarization process is formulated as a binary classification problem. A set of training documents and their references summaries are required. Sentences are classified based on statistical features as summary or non-summary sentences.

Several Arabic summarization systems have been adopting machine learning and statistical techniques. For instance, in [28] the authors integrate Bayesian and genetic programming (GP) classification methods to generate summaries, using a reduced set of features. The system uses manually labelled corpora for training. Experiments show that Bayesian classifier tends to have high recall unlike GP classifier, which has a high precision. When combining both classifiers, the authors found that the recall and the summary size are increased, but when using the intersection of the two classifiers, the precision is increased and the summary size is decreased.

Later, in [29], the authors investigate the use of several classification methods including: probabilistic neural network (PNN), genetic algorithm (GA), Gaussian mixture model (GMM), feed forward neural network (FFNN), and mathematical regression (MR) for automatic text summarization task. The authors proposed a

trainable summarizer that use ten features such as sentence centrality, position, keywords, sentence similarity to the title, etc. The authors investigate, at first, the contribution of each feature on the summarization process. Then all features are used to train the previously mentioned methods on a manually created corpus, in order to obtain features weights. After that, the models are used to rank sentences in the testing corpus. Highest-ranking sentences are selected to produce summaries. Numerous experiments were also performed using DUC 2001 corpus. The obtained results indicated that GMM model is the best.

In the same context, [30] use support vector machine (SVM) algorithm in their system to produce summaries. The authors use eight statistical features among which sentence position, title keyword, indicative expression, TF-IDF score, etc. Only 60 Arabic documents are used in their experiments. The preliminary results published are encouraging (F-measure = 0.991). However, the authors could have extended their evaluation on a larger corpus to prove the effectiveness of their approach.

Recently, [31] proposed a supervised approach using AdaBoost to produce Arabic extracts. The authors use a set of statistic features such as overlap with word title, sentence position, sentence length, etc.

After building the AdaBoost learning model, all features are extracted from each sentence in the input document (to be summarized). Then, the features vectors are passed to the AdaBoost classifier, which decides whether their corresponding sentences should be included in the summary. The authors use a manually created corpus. The performance evaluation in term of F-measure is compared to those obtained using j48 decision trees as well as multilayer perceptron (MLP). The obtained results indicate that the proposed model outperforms multilayer perceptron and j48 decision trees.

4.4 Graph Based Approach

In the graph-based approach, the document is represented in the form of undirected graph. For every sentence, there is a node. An edge between two nodes is drawn if there is a relation between these two nodes. A relation can be a cosine similarity above a threshold, sharing a common word, or any other type of relationships. After drawing a graph, it is possible to view the sub-graphs of connected nodes as a cluster of distinct topics covered in the document.

Recently [32] proposes a graph-based approach for Arabic document summarization. In this approach, each document is represented by a weighted directed graph, whose nodes correspond to document sentences, and edges weights correspond to similarity between sentences. This similarity is determined by ranking the sentences according to some statistical features. The summary is extracted by finding the shortest path between the first and the last nodes in the graph considering the user compression ratio. Evaluation is done using EASC corpus, and intrinsic methods.

4.5 Textual Entailment Based Approach

Textual entailment has been introduced as a general framework for modelling semantic variability in several NLP tasks. An entailment relation consists in determining whether the meaning of one sentence can be inferred by another one [4]. The summary obtained

by using entailment inferences only includes sentences that are not entailed by any of the sentences in the previously accumulated summary.

Very little research has been done to combine Arabic text summarization and text entailment to produce extracts. In a single case [33], the authors tackle the problem of developing Arabic text summarization system (LCEAS), that produces extracts without redundancy. Lexical cohesion is applied to distinguish the important sentences from the unimportant ones in the text. As a result, poor information is removed from the text before applying the text entailment algorithm. In the next stage, cosine directional similarity method is applied to decide which sentences are not redundant. The text entailment algorithm suggested in [34] is enhanced to make it suitable for Arabic language. Performances evaluation of LCEAS are compared with previous Arabic text summarization systems. Results indicate that LCEAS outperforms the previous Arabic text summarization systems.

4.6 Ontology Based Approach

Arabic WordNet is a lexical database for Arabic. It clusters words into sets of synonyms called synsets, together with short general definitions called gloses, and determines the different semantic relations between these synonym sets.

Some researchers tend to use this lexical database in their systems. For instance, [12] presented a new query based Arabic text summarization system (OSSAD) using Arabic WordNet and an extracted knowledge base. Both Arabic WordNet and the domain specific knowledge base are used to expand the user's query. For summarization, the authors use decision tree algorithm. When comparing OSSAD generated summary against other Arabic summarization systems tested on the same data, the results show that OSSAD reach the best performances.

We end this section by Table 1, which presents a summary of the surveyed studies in chronological order.

Finally, it should be noted that, we can't compare the results obtained by these studies, because these systems are not evaluated using the same corpus and the same evaluation methods. As we can see in Table 1. In the majority of the surveyed researches, authors used their own corpus to evaluate their systems. This is due to the lack of publically available Arabic gold-standard summaries for several years.

5 Limitations of Extractive Approaches and Main Challenges

As we over mentioned, all the summarization approaches described in this paper are extractive. This means that sentences are selected from the input document to produce a summary. Unless a background repository is being used, the system is limited only to the words explicitly mentioned in the input text [5]. In machine learning based approach such as [28–30] other limitations appear. One limitation is ignoring relevant words that appear in abundance in the testing document but not in the training document, so the system lacks the ability to analyze such words, and it will treat them as unimportant words.

Table 1. A summary of the surveyed studies in chronological order

Research work	Year	Approach	Input	Evaluation method	Used corpus
[18]	2005	RST	Single document	Precision	Author's corpus
[28]	2006	Bayesian and Genetic programming	Single document	Recall, precision, F-measure	Authors' corpus
[29]	2009	GA, MR, FFNN, PNN, GMM	Single document	ROUGE-1, recall, precision	Authors' corpus
[19]	2009	RST	Single document	Not mentioned	Authors' corpus
[30]	2010	SVM	Single document	Precision, recall, F-measure	Authors' corpus
[27]	2011	K-means algorithm	Multi-document	ROUGE-1, recall, precision	DUC 2002 corpus and Arabic translated version
[26]	2012	Clustering	Single document	Recall and Precision	Authors' corpus
[20]	2012	RST	Single document	Precision, recall, F-measure, All ROUGE measures	Authors' corpus
[21]	2013	RST and Vector space model	Single document	Recall, precision, F-measure	Authors' corpus
[12]	2013	Ontology + decision tree algorithm (C4.5)	Single document	ROUGE-L	Authors' corpus and EASC corpus
[24]	2014	Clustering + mRMR	Single and multi-document	ROUGE-1 ROUGE-2	EASC, TAC2011 MultiLing Pilot corpus
[25]	2014	clustering	Single and multi-document	ROUGE measures	EASC corpus
[32]	2014	Graph	Single document	Precision, recall, F-measure	EASC corpus
[31]	2015	AdaBoost	Single document	Precision, recall, F-measure	Authors' corpus
[23]	2015	SDRT	Single document	Precision, recall, F-measure	Authors' corpus, Arabic Treebank (ATB v3.2 part3)
[33]	2015	Text entailment and Lexical cohesion	Single and Multi-document	ROUGE-2, ROUGE-L, ROUGE-W, ROUGE-S, AutoSummEng	Authors' corpus and EASC corpus

Another limitation is the lack of detection for the implicit relationships between words in the input document. The ability to detect such relationships requires an external knowledge and an analysis module. Most of the Arabic text summarization approaches are affected by a similar limitation in the detection of concepts and the relatedness between them. We think that, this is due to the shortage of linguistic resources for Arabic language.

For discourse theories based approach, other challenges appear. For instance, identifying discourse units boundaries in Arabic texts is not an easy task. One possible reason is the irregular use of punctuation marks in Arabic texts.

Furthermore, Arabic discourse connectives are highly ambiguous. Indeed, we can easily find an Arabic discourse connective that can signal more than one discourse relation and in some cases has no discourse usage. This leads to several problems in discourse segmentation and even relations labeling. Taking as an example the connectives "و". According to [35] this connective has six meaning, which can be classified into two classes called "fasl" and "wasl". The first class includes the states where the connective is a good indicator to begin a segments (it has a discourse usage). This class contains: (1) "واو القسم" that means testimony, (2) "ورب" that means few or someone and (3) "واو الاستئناف" that simply joins two unrelated sentences. The second class includes the different states where the connector has no discourse usage, and it has no effect on the segmentation. This class contains: (1) "واو الحال" that introduces a state, (2) "واو المعية", which means the accompaniment and (3) "واو العطف" that relates words or sentences.

Finally, we can say that determining the effective features that extract the main ideas from the input document and that cover all important themes is a greater challenge in extractive text summarization especially for Arabic language.

6 Conclusion

This survey paper is focusing on extractive Arabic text summarization approaches. We presented the most recent progresses and researches raised in this field.

At first, we described some basic notions related to automatic text summarization, and some salient characteristics of Arabic language, and then we presented the main approaches proposed in this field to generate Arabic extracts. Finally, we discussed the limitations of these approaches and the major challenges faced when dealing with such application.

As a conclusion, we can say that Arabic text summarization is still in its initial stage compared to works done in English and other languages. This is partially, due to the shortage of ANLP tools and the complex morphology of Arabic language.

References

1. Gupta, V., Lehal, G.S.: A survey of text summarization extractive techniques. Emerg. Technol. Web Intell. **2**(3), 258–268 (2010)
2. El-Shishtawy, T., El-Ghannam, F.: keyphrase based Arabic summarizer (KPAS). In: 8th International Conference on Informatics and Systems (INFOS), p. NLP-7. IEEE (2012)
3. Radev, D.R., Hovy, E., McKeown, K.: Introduction to the special issue on summarization. Comput. Linguist. **28**(4), 399–408 (2002)
4. Lloret, E., Ferrández, O., Munoz, R., Palomar, M.: A Text summarization approach under the influence of textual entailment. In: NLPCS, pp. 22–31 (2008)
5. Bawakid, A.: Automatic documents summarization using ontology based methodologies. Doctoral dissertation. University of Birmingham (2011)
6. Hasanuzzaman, H.: Arabic language: characteristics and importance. The Echo. J. Humanit. Soc. Sci. **1**(3), 11–16 (2013)
7. Keskes, I., Zitoune, F.B., Belguith, L.H.: Splitting Arabic texts into elementary discourse units. ACM Trans. Asian Lang. Inf. Process. (TALIP) **13**(2), (2014)
8. Shaheen, M., Ezzeldin, A.M.: Arabic question answering: systems, resources, tools, and future trends. Arabian J. Sci. Eng. **39**(6), 4541–4564 (2014)
9. Lloret, E., Palomar, M.: Text summarization in progress: a literature review. Artif. Intell. Rev. **37**(1), 1–41 (2012)
10. El-Haj, M., Kruschwitz, U., Fox, C.: Creating language resources for under-resourced languages: methodologies, and experiments with Arabic. Lang. Res. Eval. **49**(3), 549–580 (2015)
11. Luhn, H.P.: The automatic creation of literature abstracts. IBM J. Res. Dev. **2**(2), 159–165 (1958)
12. Imam, I., Nounou, N., Abdul Khalek, H.: An ontology-based summarization system for Arabic documents (ossad). Int. J. Comput. Appl. **74**(17), 38–43 (2013)
13. Lin, C.Y.: Rouge: a package for automatic evaluation of summaries. In: Moens, M., Szpakowicz, S. (eds.) Text Summarization Branches Out, Proceedings of the ACL 2004 Workshop, vol. 8 (2004)
14. Giannakopoulos, G., Karkaletsis, V.: AutoSummENG and MeMoG in evaluating guided summaries. In: TAC 2011 Workshop, Maryland, USA (2011)
15. Jones, K.S., Galliers, J.R. (eds.): Evaluating Natural Language Processing Systems. LNCS, vol. 1083. Springer, Heidelberg (1995). https://doi.org/10.1007/BFb0027470
16. Giannakopoulos, G.: Multi-document multilingual summarization and evaluation tracks. In: Proceedings of the MultiLing 2013 Workshop on Multilingual Multidocument Summarization, Sofia, Bulgaria, pp. 20–28 (2013)
17. Mann, W.C., Thompson, S.A.: Rhetorical structure theory: toward a functional theory of text organization. Text-Interdiscipl. J. Study Discourse **8**(3), 243–281 (1988)
18. AlSanie, W.: Towards a suitable rhetorical representation for Arabic text summarization. In Kotsis, G., Taniar, D., Bressan, S., Ibrahim, I.K., Mokhtar, S. (eds.) iiWAS, vol. 196, pp. 535–542. Austrian Computer Society (2005)
19. Mathkour, H.I.: A Novel rhetorical structure approach for classifying Arabic security documents. Int. J. Comput. Theory Eng. **1**(3), 195 (2009)
20. Azmi, A.M., Al-Thanyyan, S.: A text summarizer for Arabic. Comput. Speech Lang. **26**(4), 260–273 (2012)
21. Ibrahim, A., Elghazaly, T., Gheith, M.: A novel Arabic text summarization model based on rhetorical structure theory and vector space model. Int. J. Comput. Linguist. Nat. Lang. Process. **2**(8), 480–485 (2013)

22. Asher, N., Lascarides, A.: Logics of Conversation. Cambridge University Press, Cambridge (2003)
23. Keskes, I.: Discourse analysis of Arabic documents and application to automatic summarization. Doctoral dissertation, Université de Toulouse (2015)
24. Oufaida, H., Nouali, O., Blache, P.: Minimum redundancy and maximum relevance for single and multi-document Arabic text summarization. King Saud Univ.-Comput. Inf. Sci. **26**(4), 450–461 (2014)
25. Noori Fejer, H., Omar, N.: Automatic Arabic text summarization using clustering and keyphrase extraction. In: International Conference on Information Technology and Multimedia (ICIMU), pp. 293–298. IEEE (2014)
26. Haboush, A., Al-Zoubi, M., Momani, A., Tarazi, M.: Arabic text summarization model using clustering techniques. Word Comput. Sci. Inf. Technol. **2**(3), 62–67 (2012)
27. El-Haj, M., Kruschwitz, U., Fox, C.: Exploring clustering for multi-document arabic summarisation. In: Salem, M.V.M., Shaalan, K., Oroumchian, F., Shakery, A., Khelalfa, H. (eds.) AIRS 2011. LNCS, vol. 7097, pp. 550–561. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25631-8_50
28. Sobh, I., Darwish, N., Fayek, M.: An optimized dual classification system for Arabic extractive generic text summarization. In the 7th Conference on Language Engineering, pp. 149–154 (2006)
29. Abdel Fattah, M., Ren, F.: GA, MR, FFNN, PNN and GMM based models for automatic text summarization. Comput. Speech Lang. **23**(1), 126–144 (2009)
30. Boudabous, M.M., Maaloul, M.H., Belguith, L.H.: Digital learning for summarizing arabic documents. In: Loftsson, H., Rögnvaldsson, E., Helgadóttir, S. (eds.) NLP 2010. LNCS (LNAI), vol. 6233, pp. 79–84. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-14770-8_10
31. Belkebir, R., Guessoum, A.: A supervised approach to Arabic text summarization using adaboost. In: Rocha, A., Correia, A.M., Costanzo, S., Reis, L.P. (eds.) New Contributions in Information Systems and Technologies. AISC, vol. 353, pp. 227–236. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-16486-1_23
32. Al-Taani, A.T., Al-Omour, M.M.: An extractive graph-based Arabic text summarization approach. In: The International Arab Conference on Information Technology, Jordan (2014)
33. Al-Khawaldeh, F., Samawi, V.: Lexical cohesion and entailment based segmentation for Arabic text summarization (LCEAS). Word Comput. Sci. Inf. Technol. **5**(3), 51–60 (2015)
34. Tatar, D., Mihis, A., Lupsa, D., Tamaianu-Morita, E.: Entailment based linear segmentation in summarization. Int. J. Softw. Eng. Knowl. Eng. **19**(08), 1023–1038 (2009)
35. Khalifa, I., Feki, Z., Farawila, A.: Arabic discourse segmentation based on rhetorical methods. Int. J. Electric. Comput. Eng. **11**(1), 10–15 (2011)

Information Retrieval Systems

Toward a Context-Aware Multilingual Personalized Search

Mohamed Seghir Hadj Ameur(✉), Youcef Moulahoum(✉), Lamia Berkani(✉), and Ahmed Guessoum(✉)

Laboratory for Research in Artificial Intelligence (LRIA), NLP, Machine Learning and Applications (TALAA) Group, Department of Computer Science, University of Science and Technology Houari Boumediene (USTHB), Bab-Ezzouar, Algiers, Algeria
{mhadjameur,lberkani,aguessoum}@usthb.dz, moulahoum.youcef@gmail.com

Abstract. In recent years, personalized search has widely been used in Information Retrieval Systems (IRS) to provide the end user with more sophisticated and accurate search results. A basic element that plays an important role in personalized search is the user context which contains several aspects such as the user preferences, navigation history, habits, etc. A user may express his information needs in various languages. This requires the IRS to be able to consider all the contextual information provided in these languages. In this work, we present M-CAIRS, a Multilingual Context-aware Information Retrieval System that takes into account multilingual user contexts to better model the user search interests. Experimental results show a strong correlation between the user's relevance judgment and the automatic results obtained by our system, which proves the consistency and adequacy of our proposal.

Keywords: Information retrieval · Multilingual information retrieval
Reference ontology · Document indexing · User context · User profile
Relevance judgment

1 Introduction

In recent years, the amount of information available on the web has seen an exponential growth. According to the Internet live stats website[1], there were at least 1.2 billion websites on the indexed web as of May 2017, and in every second, approximately more than 60,000 Google queries are launched. This explosion in both the amount of data and the launched queries has made it hard for Information Retrieval Systems (IRS) to accurately find and identify relevant information that can address the users needs and preferences in a precise manner.

Search engines are one of the most popular tools to find information on the web. Classical search engines return the same results to different users (one size

[1] Internet live stats is a website that provides live statistics regarding the Internet http://www.internetlivestats.com.

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 175–190, 2018.
https://doi.org/10.1007/978-3-319-73500-9_13

fits all) even though each one of them has a distinct context and a specific goal when searching for information. This generally ends up providing the users with some irrelevant results that fail to address their specific information needs. The problem of query ambiguity is also one of the main reasons for search quality decay [1]. The query ambiguity is due to several reasons including polysemy (a single word may have multiple meanings), and synonymy (different words may have the same meaning). For example, a user who searches for the word "apple" may be interested in either a fruit or a company. Therefore, in order to solve these problems, the information retrieval community has made a huge focus on personalization. Personalized Search aims to reduce the queries ambiguity and return the most probable results that are more likely to be of interest to the user based on specific user modeling techniques.

User Modeling aims to build an adequate representation that models the user's interests either individually [2] or as part of a community [3] that shares similar preferences. When user modeling techniques are incorporated, the search process is generally called Personalized Search; which has been widely used in several domains such as Information Retrieval [2,4–7], Recommender Systems [8–10] and many applications such as e-learning [11,12] to provide the end user with more relevant personalized services. One of the main elements that play an essential role in personalized search is the user context. When an IRS takes into account the user context it is called Context-aware Information Retrieval System (CIRS). Two factors are important in contextual information retrieval: (1) the user's short-term context, which includes his requests and various aspects such as spatio-temporal information, and (2) the user's long-term context which includes his interests, preferences, knowledge, habits, expertise, etc. An important issue that needs to be addressed when attempting to model the user's context is the problem of incorporating all the languages he uses in his queries. Indeed, if a user queries the web in various languages, for example English, Arabic and French, the CIRS should be able to effectively model and maintain this user's preferences and interests in each one of these languages.

In this work, we build M-CAIRS a complete context-aware information retrieval framework that effectively models the user's long term interests. Our proposal is based on the work of Sieg et al. [13] and Gupta et al. [14] and proposes improvements in profile updating and results re-ranking. Furthermore, one of the main contributions of this work is the proposition of a method to effectively incorporate multiple languages including Arabic in the ontological user profile modeling framework.

The remainder of this paper is organized as follows: the next section presents the background and some relevant related works that have been done in this area and motivates our contribution. The details of our proposal are then described in Sect. 3. In Sect. 4, we present and discuss the tests we have done and the results we have obtained. In the last Section, we conclude our work and highlight some possible future improvements.

2 Background and Related Work

In this section we define some important concepts that will be used in the remainder of this paper, then we shed light on some of the relevant works that have been made in this research area.

2.1 Background

The notions of context and profile may have several definitions depending on the application at stake. Here we give our own definitions of these concepts along with other relevant concepts that we will be using in this paper.

User Context: We define the user context as all the information about the user that can be used to improve the personalized retrieval process. Two types of user contexts are distinguished: (1) the short-term context, which includes the user's requests and various aspects such as spatio-temporal information (i.e. geographical position, time, etc.), and (2) the user's long-term context, widely known as the user profile, which contains useful information about the user such as his interests, preferences, knowledge, habits, expertise, search history, etc.

User Profile: We define the user profile as a source of knowledge that holds the user's long-term context. A certain structural representation is generally used to store, maintain and update a user profile according to the changes that occur in his interests and preferences.

The Open Directory Project (ODP)[2]: It is one of the largest existing web directories, developed and maintained by a vast community of editors (over 90,000 publishers). It contains about 4 million websites distributed into a hierarchy of over 1 million manually created categories (concepts)[3], where each concept contains a set of manually associated websites. The first level of this hierarchy contains generic concepts such as: Arts, Computers, Sport, etc. These concepts become more specific as one goes deeper and deeper into the hierarchy.

The Reference Ontology [15]**:** It is an instance of a preexisting hierarchy such as the ODP. This ontology is generally used to represent the user profile as a hierarchy of concepts in which each concept is associated with an interest score which indicates the degree to which the user is interested in this concept. This user profile representation is very useful for personalization to keep track of the user's interests and update them according to the user's daily activities.

[2] http://dmoztools.net/.
[3] The nodes of the ontology (hierarchy) are generally called concepts.

2.2 Related Work

The task of personalizing the search results is a very complicated process that includes several important steps, the major steps are: collecting the users' information, building and updating their profiles and re-ranking the search results according to the profiles being built. This section explain these steps and presents some of the most relevant works which attempt to address each one of them.

The first step is to collect the user's information which can be done either explicitly or implicitly. The gathering of explicit user data generally asks the users to provide their areas of interests, their preferences and/or their (positive/negative) feedbacks regarding the returned search results they are provided with. An example of such a system is the work of Syskill and Webert [16] in which the user is explicitly asked to rate web pages, and based on his feedback, a software agent learns to decide which page might be of interest to a specific user. Implicit data collection [17–20] on the other hand is done automatically, where the user context (clicks, bookmarks, search history, desktop information, etc.) is collected without any user intervention.

The second step is to build and maintain the user profiles. User profiles are generally represented based on keywords also known as "Keyword profiles" or concepts known as "Concept profiles". Keyword profiles [21] are created by extracting keywords from a set of documents, web pages and/or bookmarks visited by the user. They are stored as part of his browsing history. The most important keywords on a web page are identified using some specific weighting methods, and only the most highly weighted terms are kept. An example of such a system is the one presented by Moukas and Alexandros [22] in which extracted keywords were weighted using the $TF–IDF$ measure [23] and a vector representation was used to represent both the user profile and the retrieved documents. Concept profiles [20,21] are represented such that each concept represents an abstract domain. These concepts are usually driven from an existing hierarchy such as the ODP. A numerical value called "interest score" is generally associated with each concept to indicate the degree to which the user is interested in it. In terms of concept profiles Sieg et al. [13] constructed the user profile as an instance of a predefined ODP hierarchy. When a user interacts with the system by selecting or viewing new documents, the scores of each concept will be updated on the basis of its similarity with the viewed documents. A specific propagation algorithm is also used to allow activation weights to spread throughout the entire ontological profile.

The last step is to make use of the built profile to reorder the search results so as to better suit the user's interests. To that end, Sieg et al. [13] presented a re-ranking method that reorders the Search Engine results according to the user profile interest scores. Gupta et al. [14] proposed a similar approach with the incorporation of the original Search Engine ordering of the returned pages as a feature along side the user profile.

The amount of focus on personalized retrieval in regard to the Arabic language is very limited as stated in [24]. Some of the efforts in this area includes, for instance, the work of Houssem et al. [25] in which a query enhancement

system is proposed to extend the users' search queries on the basis of their profiles and the Arabic Wordnet (AWN) [26][4], and the work of Safi et al. [24] in which a hybrid profile construction method is introduced to incorporate both implicit and explicit users' information using the AWN as a reference hierarchy. A special method is then used to update and maintain the conceptual interest scores in the built profiles.

Even with the amount of research work done on personalized information retrieval, no single strategy has seemed to yield ideal results, thus the continuing efforts to improve them. This work aims to bring further improvements to several aspects of this area such as multilingual retrieval, user profile building and maintenance, as well as results re-ranking.

3 Context-Based Multilingual Personalized Search

This section presents M-CAIRS, our proposed system architecture, and describes in depth the functioning mechanism of each of its components.

Figure 1 illustrates the different components of our proposed system, along with the interactions between them. There are two main tasks: the first one attempts to gather the user daily browsing activities and use them to build and maintain his ontological user profile; the second one re-ranks the search results corresponding to the user's query reflecting the learned user profile.

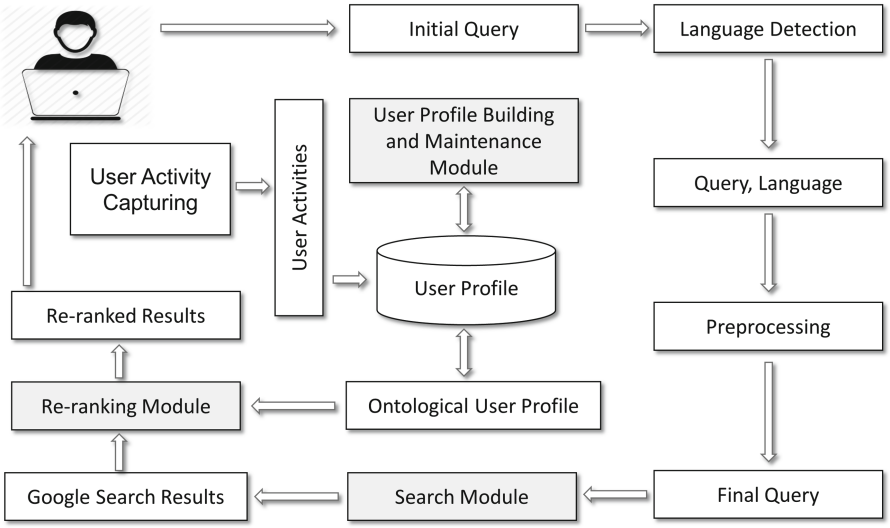


Fig. 1. The architecture of our proposed multilingual context-aware IRS (M-CAIRS)

[4] http://globalwordnet.org/arabic-wordnet/.

3.1 Building and Maintenance

The goal of this model is to gather the users' daily browsing activities and use them to build and maintain their user profiles.

First, we define the user profile as an instance of the ODP hierarchical concept database. The depth of the ODP hierarchy can reach up to 11 levels, which makes the concepts at the bottom very specific. We have used only the first two levels of the ODP hierarchy, which prevents the concepts from being too specific and keeps them relatively general. This, we believe, is more suitable for holding their long-term interests as shown in Fig. 2. We have considered only three languages: English, Arabic and French, which are the languages mostly used by the users we have investigated[5]. Since the ODP is principally an English-based hierarchy, all the other languages such as Arabic and French are found in the first level under the concept "World". We have tweaked the structure of the hierarchy to place the three languages at the top level as shown in Fig. 2. Figure 2 shows the reference ontology used in M-CAIRS. It includes a total of 548 concepts. We assume that our choice will keep the profile a little general yet suitable to hold the users long-term interests in each one of the three considered languages.

For each language, all the documents found under the same concept will be merged together to form a single super-document. These super-documents will then go through a preprocessing step which includes stemming, normalization and empty words removal according to each specific language. A vector space representation is then used to represent each concept by a vector of terms of length n where n is the vocabulary size in the considered language. The $TF–IDF$ weighting [23] is then used to produce a vector of weights for each concept.

$$TF–IDF_{i,j} = \frac{TF(t_i, d_j)}{max(TF(t, d_j))} * log(\frac{n}{n_i}) \quad (1)$$

$TF(t_i, d_j)$ is the frequency of the i^{th} term in the j^{th} document, $max(TF(t, d_j))$ is the highest term frequency in document j and $log(\frac{n}{n_i})$ is the inverse document frequency of term i in the collection where n is the total number of terms and n_i is the frequency of term i.

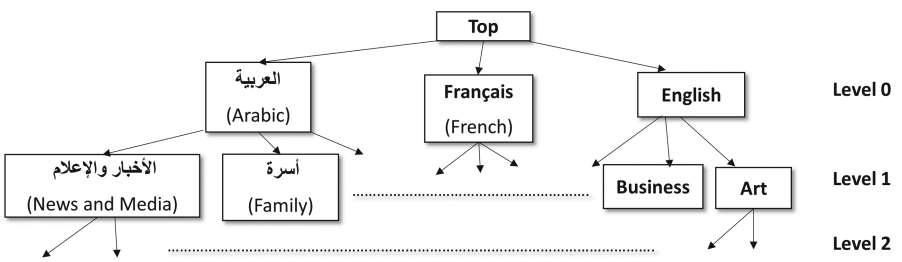


Fig. 2. The ODP reference ontology used in M-CAIRS

[5] We note that other languages can be integrated exactly in the same way.

When a user visualizes a document $d_j = \{t_1, t_2, ..., t_m\}$ containing m terms, this document will be similarly preprocessed and represented in a vector space representation as a vector of weights $w_1, w_2, ..., w_n$. Having this vector representation for both the concepts and the visualized documents allows us to easily estimate the similarity between them, which is very useful to update the user profile accordingly.

As mentioned above, the user profile is going to be an instance of the reference ontology shown in Fig. 2 with the addition of a specific weight called interest score $IS(c_i)$ which is associated to each concept c_i to indicate the user's level of interest in it. The following Algorithm 1 is proposed to update and maintain the user's interests scores:

Algorithm 1. Pseudo algorithm for updating the interest scores in the ontological user profile

Input : $C = c_1, ..., c_n$; $D = d_1, ..., d_n$; $T = t_1, ..., t_n$.
Output: Returns the updated concepts $C = c'_1, ..., c'_n$.

```
Function UpdateScores(C, I):
    Initialize the priorityQueue;
    Initialize all the activation scores;
    foreach d_i ⊆ I do
        L_i = language identification of d_i;
        foreach c_j ⊆ C found under L_i do
            if firstLevel(c_j) and sim(d_i, c_j) > 0 then
                c_j.activation = log(t_i / size(d_i)) * IS(c_i) * sim(d_i, c_j)
                priorityQueue.add(c_j)
            end
        end
        while priorityQueue.size > 0 do
            Sort the priorityQueue;
            c_s = priorityQueue[0];
            priorityQueue.dequeue(c_s);
            foreach concept c_k linked with c_j do
                c_k.activation+ = c_s.activation * c_k.weight;
                priorityQueue.enqueue(c_k)
            end
        end
    end
```

Algorithm 1 is based on the spreading algorithm proposed by Sieg et al. [13] with only some differences: we have included the time spent and the size of the visualized documents for the estimation of interest scores as suggested by Gupta et al. [14], and we have also extended it to take into account multiple languages.

Given a set of documents $D = d_1, ..., d_n$ visualized by the user for a given amount of time $T = t_1, ..., t_n$, the algorithm attempts to update the interest

scores of the user concepts $C = c_1, ..., c_n$ based on their similarity with the visualized documents.

For each visualized document d_i, we first start by identifying its language L_i[6], then we estimate its degree of similarity with each first level concept c_j in the user profile found under language L_i. Then we associate an activation score to c_j on the basis of its similarity with d_i along with the size and the time spent on it. This will give an activation score to all the concepts found in the first level (for the considered language)[7]. For the rest of the concepts we use the spreading mechanism presented by Sieg et al. [13] to spread the activation from each concept to its children, and so on. The weight of each relation w_{is} between a parent concept i and one of its children s determines how much activation this parent should spread to each one of its children. This weight is calculated using the following formula (proposed in [13]):

$$w_{is} = \frac{\overrightarrow{n_i} * \overrightarrow{n_s}}{\overrightarrow{n_i} * \overrightarrow{n_i}} \tag{2}$$

where $\overrightarrow{n_i}$ is the terms vector of concept i and $\overrightarrow{n_s}$ is the terms vector of its child s. This formula allow a parent to spread more weights to its children that are similar to it.

With this algorithm we make sure that the user profile will always be up to date as the user visualizes new documents on the browser, and expresses new interests in different languages.

3.2 Re-ranking the Search Results

The re-ranking module is used to improve the initial order presented by the Google Search Engine, in response to a certain user query using his built ontological profile.

First, we identify the language in which the user query has been issued to be preprocessed accordingly, then the Google Search API will be used to obtain the corresponding results.

Algorithm 2 is similar to the re-ranking algorithm proposed by Sieg et al. [13], the difference is that we discard the similarity between the document and the query and we instead leave it to be handled by the search engine. We also take account of the original Google ranking when estimating the new result order as suggested by Gupta et al. [14].

Given that the user profile is defined as a set of concepts $C = c_1, ..., c_n$ each with its interest score, and a set of documents visualized by the user $R(q) = d_1, ..., d_n$ in response to his query q in their original order decided by the Google Search Engine, we start by identifying for each document d_i its most similar concept *best_concept*. Then we estimate the user interest level in that document

[6] We have used the Google Language Detection API [27], to identify the document language.

[7] If a certain document contain textual information in several languages, the text identification process will chose the most dominant one among them.

Algorithm 2. Pseudo algorithm for search results re-ranking

Input : $C = c_1, ..., c_n$, $R(q) = d_1, ..., d_n$.
Output: Returns the new order for the documents of $R'(q)$.

```
Function Re-ranking(C, I):
    foreach d_i ⊆ R(q) do
        L_i = language identification of d_i;
        best_concept = C[0];
        best_score = 0;
        foreach c_j ⊆ C found under L_i do
            if sim(d_i, c_j) > best_score then
                best_concept = c_j
                best_score = sim(d_i, c_j);
            end
        end
        userIntrest(d_i) = IS(best_concept) * sim(q, best_concept)
        final_rank(d_i) = α * userIntrest_{d_i} + (1 − α) * Google_rank(d_i)
    end
```

as the product of the user interest score in d_i with the similarity between the query and the *best_concept*. The final rank of each document d_i is then estimated as a linear combination of the original Google rank and the user interest score in d_i.

Our intuition is that including the original Google ranking will be important since it uses the Google Page Rank (PR) algorithm [28][8] which assigns higher ranking for more frequently referenced web pages/sites. In the same time, we include our estimated profile-based interest scores to hopefully maintain a certain balance between the importance of a given page and the user's degree of interest in it.

4 Experiments

This section presents the details of our experiments and gives an in-depth discussion of the incorporated tests.

Our experiments examine two important aspects: first, we want to make sure that the interest scores in the user ontological profile stabilize after a finite number of updates, which implies that the long-term user interests are successfully learned. The second aspect aims to check if our proposed framework manages to bring an improvement to the ordering of the standard search results order making it more relevant to the user.

We first explain the process of data collection and preparation. We then present the evaluation metrics we have used. Finally, we address the two aforementioned key experiments.

[8] The page rank algorithm works by estimating the rank/quality of a web page based on the number and the importance of the web pages that reference it.

4.1 Users Data

Evaluating personalized systems is a very problematic task, since it requires direct user intervention in order to provide his judgment of relevance which poses a huge problem of confidentiality as pointed out by Gauch et al. [15].

In most of the personalized information retrieval research projects the authors tend to collect user data along with their relevance judgment from their own students or team members since random web users usually don't welcome the idea of sharing their personal search information and it is even more problematic to get them to provide their judgment of relevance since it will cost them a considerable amount of time and effort [14,29].

In order to automate and facilitate the capture of users' activities, we have developed a browser extension that gets installed on the user's web browser and instantly sends his browsing activities (the URLs of the visited websites, time spent on each web site, etc.) to our web server. We have provided this extension to 24 users from our university and automatically gathered their daily browsing activities for about two months period starting from March 2015. Only the top five profiles that have the maximum number of visited URLs have been considered in our evaluations. Table 1 shows the number of visited URLs for each one of these five selected profiles.

Table 1. Statistics about the number of URLs visited by each one of the 5 selected users

User profiles	URLs
1	233079
2	61553
3	49704
4	36694
5	33426

4.2 Evaluation Metrics

To investigate the effectiveness of our personalized re-ranking system we incorporate two measures: the Top-n Recall and the Top-n Precision. First, we define the standard recall and precision metrics then, based on these, we formulate the Top-n Recall and the Top-n Precision.

Recall. The recall is the ratio between the set of relevant documents retrieved by the system and the total number of relevant documents.

$$Recall = \frac{relevant\ documents\ retrieved\ by\ the\ system}{total\ number\ of\ relevant\ documents} \tag{3}$$

Precision. The precision is the ratio between the set of relevant documents retrieved by the system and the number of retrieved documents.

$$Precision = \frac{relevant\ documents\ retrieved\ by\ the\ system}{number\ of\ documents\ retrieved\ by\ the\ system} \tag{4}$$

F_measure. The F-measure metric combines both precision and recall to give a better relevance judgment. We use the F-measure defined as follows:

$$F\_measure = 2 * \frac{Precision * Recall}{Precision + Recall} \tag{5}$$

Top-n Recall [30]**:** The Top-n Recall is the ratio between the number of relevant documents found by the system in the first n returned documents, and the total number of relevant documents that exist in the first n documents.

$$Top\_n\ Recall = \frac{number\ of\ relevant\ documents\ in\ the\ first\ n\ returned\ documents}{total\ number\ of\ relevant\ documents\ in\ the\ first\ n\ documents} \tag{6}$$

Top-n Precision [30]**:** The Top-n Precision is the proportion of relevant documents found by the system in the first n returned documents.

$$Top\_n\ Precision = \frac{number\ of\ relevant\ documents\ in\ the\ first\ n\ returned\ documents}{n} \tag{7}$$

4.3 Convergence of the Users' Profiles

To ensure that the conceptual interest scores of the user profile will stabilize after a certain number of updates, we have investigated the average rate of their incremental increase.

Figure 3 shows the average increase rate in the conceptual user profile over incremental updates. We can see that initially there is a considerable change rate of the interest scores of the user profile. This changing rate starts decreasing with the number of processed documents, and then reaches a stability level (convergence) when about 600 documents (updates) have been processed, which indicates that the user profile is such that the system has managed to learn the long-term user interests.

4.4 Re-ranking Evaluation

This evaluation aims to investigate in a practical way if the reordered search results better suit the user. To this end we have built 5 ontological user profiles

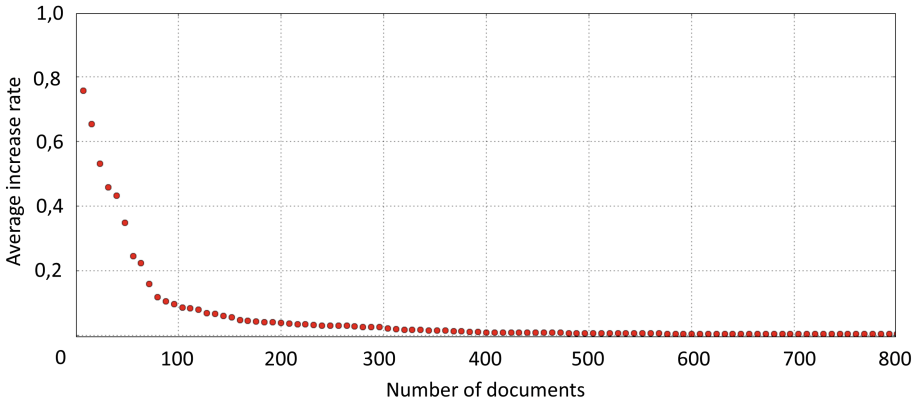


Fig. 3. Change in average increase rate when incrementally updating the user profile

Table 2. Manually prepared queries for re-ranking evaluation

	Arabic	English	French
Query 1	إعراب القرآن وبيانه	Command and conquer the game	Le concours Algeria Game Challenge
Query 2	شهادة إرجاء من الخدمة الوطنية	Find element vector C++	Venir au Canada
Query 3	تحميل المصحف المرتل للقارئ ماهر المعيقلي	Serious Game	Emploi et recrutement
Query 4	أحمد بن علي العجمي	Installing OpenCV 2.3.1 in Windows	Don du Sang Dz
Query 5	خمسات لبيع وشراء الخدمات المصغرة	Isometric character sprite	GTA V

for 5 selected users, and prepared for each one of them a set of 15 queries, 5 for each language (Arabic, English and French). These queries were manually selected according to the profile of each user. Each user was asked to provide his optimal results order for each one of his queries to be considered as references.

Table 2 shows five queries for each of the three languages for one of the five investigated users. We have executed these queries using our IRS with different α priorities (α decides the priority between Google ranking and the user-profile-based ranking). We have tested these queries with the reordering systems proposed by Gupta et al. [14] and Sieg et al. [13] for comparison purposes[9].

The graphs presented in Fig. 4 illustrate the average Top-n Recall and Top-n Precision concerning five users, reported for the original results returned by Google's Page Rank and the algorithms proposed by Sieg et al. [13] and

[9] We note that we have implemented ourselves all the systems we have compared; thus conclusions should be taken with some caution.

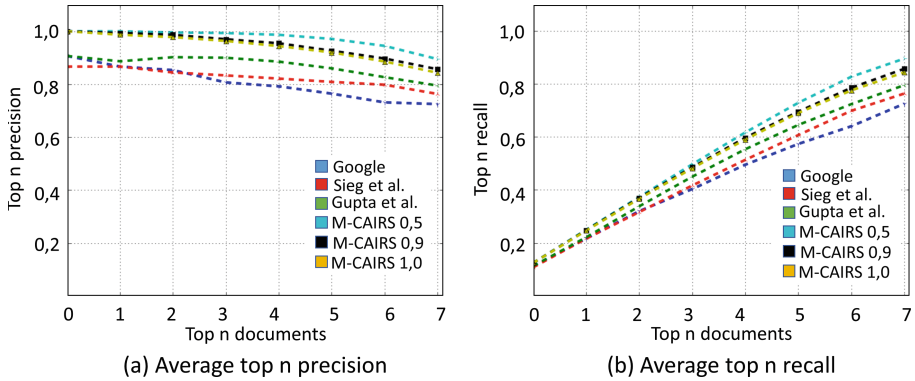


Fig. 4. Comparing the average Top-n Accuracy and Top-n Recall for the different re-ranking methodologies

Table 3. Comparing the different re-ranking methods based on the F-measure metric for each user

	User 1	User 2	User 3	User 4	User 5
Google	0,70	0,78	0,74	0,72	0,67
Sieg et al.	0,72	0,76	0,75	0,77	0,80
Gupta et al.	0,81	0,83	0,79	0,79	0,74
M-CAIRS 0.5	**0,87**	**0,92**	**0,90**	0,85	**0,90**
M-CAIRS 0.9	**0,87**	0,85	0,84	0,85	0,84
M-CAIRS 1.0	0,85	0,84	0,84	**0,87**	0,80

Gupta et al. [14], as well as our proposed re-ranking system for $\alpha = 0.5, 0.9, 1$. We can see that our proposed re-ranking approach yields more accurate results, and that when the value of $\alpha = 0.5$, the best re-ranking results are achieved.

Table 3 shows the F-measure scores for the aforementioned systems as reported for each individual user. The results show that our re-ranking approach for $\alpha = 0.5$ yields an improvement between 15% and 23% compared to those of the Google system and between 5% and 15% compared to the other re-ranking methods. This result confirms again the effectiveness of our combined linear re-ranking. As a matter of fact, we note that the reported F-measure of our proposed system for $\alpha = 0.5$ is always around 0.9 which is very close to the optimal possible re-ranking provided by the users.

The results we have obtained suggest that giving the same importance to the personalized profile-based re-ranking and the Google-based ordering produces more relevant re-ranking results. This also confirms our original intuition that encourages the inclusion of Google ranking of the returned pages along side the ontological user profile interest score.

5 Conclusion

In this work we have presented M-CAIRS, a multilingual Context-aware IRS in which the user's long-term interests are automatically learned and represented using an ontological user profile. The constructed profile is then used to re-rank the search results in a way that better fits the user's needs and preferences.

Our system can easily be deployed on a web server and accessed using any web browser. We believe that the contributions of this work are as follows:

- This work addresses the case of multilingual personalized retrieval with the inclusion of the Arabic language.
- An effective re-ranking method is proposed to better meet the users' information needs.
- A comparative study has been done between different IR systems.

This work can be developed in various directions. These include the incorporation of more contextual short-term features such as the user's geographical position, time, etc. Also other languages beside English, Arabic and French can be incorporated in the user profile.

References

1. Stokoe, C., Oakes, M.P., Tait, J.: Word sense disambiguation in information retrieval revisited. In: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 159–166. ACM (2003)
2. Sugiyama, K., Hatano, K., Yoshikawa, M.: Adaptive web search based on user profile constructed without any effort from users. In: Proceedings of the 13th International Conference on World Wide Web, pp. 675–684. ACM (2004)
3. Teevan, J., Morris, M.R., Bush, S.: Discovering and using groups to improve personalized search. In: Proceedings of the Second ACM International Conference on Web Search and Data Mining, pp. 15–24. ACM (2009)
4. Teevan, J., Dumais, S.T., Horvitz, E.: Personalizing search via automated analysis of interests and activities. In: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 449–456. ACM (2005)
5. Speretta, M., Gauch, S.: Personalized search based on user search histories. In: Proceedings of the 2005 IEEE/WIC/ACM International Conference on Web Intelligence, pp. 622–628. IEEE (2005)
6. Qiu, F., Cho, J.: Automatic identification of user interest for personalized search. In: Proceedings of the 15th International Conference on World Wide Web, pp. 727–736. ACM (2006)
7. White, R.W., Chu, W., Hassan, A., He, X., Song, Y., Wang, H.: Enhancing personalized search by mining and modeling task behavior. In: Proceedings of the 22nd International Conference on World Wide Web, pp. 1411–1420. ACM (2013)
8. Schafer, J.B., Frankowski, D., Herlocker, J., Sen, S.: Collaborative filtering recommender systems. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) The Adaptive Web. LNCS, vol. 4321, pp. 291–324. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72079-9_9

9. Berkovsky, S., Kuflik, T., Ricci, F.: Mediation of user models for enhanced personalization in recommender systems. User Model. User-Adapted Interact. **18**(3), 245–286 (2008)
10. Adomavicius, G., Tuzhilin, A.: Context-aware recommender systems. In: Ricci, F., Rokach, L., Shapira, B. (eds.) Recommender Systems Handbook, pp. 191–226. Springer, Boston (2015). https://doi.org/10.1007/978-1-4899-7637-6_6
11. Henze, N., Dolog, P., Nejdl, W., et al.: Reasoning and ontologies for personalized e-learning in the semantic web. Educ. Tech. Soc. **7**(4), 82–97 (2004)
12. Chen, C.M., Lee, H.M., Chen, Y.H.: Personalized e-learning system using item response theory. Comput. Educ. **44**(3), 237–255 (2005)
13. Sieg, A., Mobasher, B., Burke, R.: Web search personalization with ontological user profiles. In: Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, pp. 525–534. ACM (2007)
14. Gupta, K., Arora, A.: Web search personalization using ontological user profiles. In: Babu, B.V., Nagar, A., Deep, K., Pant, M., Bansal, J.C., Ray, K., Gupta, U. (eds.) SocProS 2012. AISC, vol. 236, pp. 849–855. Springer, New Delhi (2014). https://doi.org/10.1007/978-81-322-1602-5_90
15. Gauch, S., Speretta, M., Pretschner, A.: Ontology-based user profiles for personalized search. In: Sharman, R., Kishore, R., Ramesh, R. (eds.) Ontologies. ISIS, vol. 14, pp. 665–694. Springer, Boston (2007). https://doi.org/10.1007/978-0-387-37022-4_24
16. Pazzani, M.J., Muramatsu, J., Billsus, D., et al.: Syskill & webert: identifying interesting web sites. In: AAAI/IAAI, vol. 1, pp. 54–61 (1996)
17. Joachims, T.: Optimizing search engines using click through data. In: Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 133–142. ACM (2002)
18. Jay, P., Shah, P., Makvana, K., Shah, P.: An approach to identify user interest by reranking personalize web. In: Proceedings of the Second International Conference on Information and Communication Technology for Competitive Strategies, p. 64. ACM (2016)
19. Lovaraju, D., Devi, G.L.: An ontology like model for gathering personalized web information. Int. J. Adv. Res. Comput. Sci. **8**(3), 401–403 (2017)
20. Vicente-López, E., de Campos, L.M., Fernández-Luna, J.M., Huete, J.F.: Use of textual and conceptual profiles for personalized retrieval of political documents. Knowl.-Based Syst. **112**, 127–141 (2016)
21. Gauch, S., Speretta, M., Chandramouli, A., Micarelli, A.: User profiles for personalized information access. In: Brusilovsky, P., Kobsa, A., Nejdl, W. (eds.) The Adaptive Web. LNCS, vol. 4321, pp. 54–89. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72079-9_2
22. Moukas, A.: Amalthaea information discovery and filtering using a multiagent evolving ecosystem. Appl. Artif. Intell. **11**(5), 437–457 (1997)
23. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. Commun. ACM **18**(11), 613–620 (1975)
24. Safi, H., Jaoua, M., Belguith, L.H.: PIRAT: a personalized information retrieval system in arabic texts based on a hybrid representation of a user profile. In: Métais, E., Meziane, F., Saraee, M., Sugumaran, V., Vadera, S. (eds.) NLDB 2016. LNCS, vol. 9612, pp. 326–334. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-41754-7_31

25. Houssem, S., Maher, J., Lamia, B.H.: Axon: a personalized retrieval information system in Arabic texts based on linguistic features. In: 2015 6th International Conference on Information Systems and Economic Intelligence (SIIE), pp. 165–172. IEEE (2015)
26. Black, W., Elkateb, S., Rodriguez, H., Alkhalifa, M., Vossen, P., Pease, A., Fellbaum, C.: Introducing the Arabic wordnet project. In: Proceedings of the Third International WordNet Conference, pp. 295–300 (2006)
27. Google Corporation: Google language detection API. https://cloud.google.com/translate/docs/detecting-language. Accessed 2017
28. Page, L., Brin, S., Motwani, R., Winograd, T.: The pagerank citation ranking: bringing order to the web (1999)
29. Daoud, M., Tamine, L., Boughanem, M.: A personalized graph-based document ranking model using a semantic user profile. In: De Bra, P., Kobsa, A., Chin, D. (eds.) UMAP 2010. LNCS, vol. 6075, pp. 171–182. Springer, Heidelberg (2010). https://doi.org/10.1007/978-3-642-13470-8_17
30. Hawking, D., Craswell, N., Bailey, P., Griffihs, K.: Measuring search engine quality. Inf. Retriev. **4**(1), 33–59 (2001)

Fusion Based Authorship Attribution-Application of Comparison Between the Quran and Hadith

Halim Sayoud(✉) and Hassina Hadjadj

USTHB University, Algiers, Algeria
halim.sayoud@uni.de, hadjadj.has@gmail.com

Abstract. In this paper, we conduct an investigation of automatic authorship attribution on seven Arabic religious books, namely: the holy Quran, Hadith and five other books, by using two fusion techniques. The Arabic dialect is the same (i.e. Standard Arabic) for the seven books. The genre is the same and the topic of the different books is also the same (i.e. Religion).

The authorship characterization is based on four different features: character trigrams, character tetragrams, word unigrams and word bigrams. The task of authorship identification is ensured by four conventional classifiers: Manhattan distance, Multi-Layer Perceptron, Support Vector Machines and Linear Regression. Furthermore, we propose two fusion approaches to strengthen the classification performances. Finally, a particular application is dedicated to the authorship discrimination between the Quran and Hadith, in order to see if the two books could have the same Author or not. Results have shown the importance of the fusion techniques in authorship attribution and confirm that the two books (Quran and Hadith) should belong to two different Authors, which implies that the Quran could not be written by the Prophet.

Keywords: Computational linguistics · Fusion approach · Authorship attribution
Automatic text classification · Author discrimination · Quran authorship

1 Introduction

Stylometry or author recognition is a research field that consists in recognizing the authentic author of a piece of text. It is evident that the recognition accuracy is not as high as some biometric modalities that are used in security purposes, but it has been shown that for texts with more than 2500 tokens, the recognition task becomes significantly accurate [1, 2].

Stylometry can be divided into several research fields: Authorship Attribution (referred to as AA) [3], Authorship verification, Authorship discrimination, Authorship Indexing and Plagiarism detection.

That is; determining the real author of a piece of text has raised several questions and problems for centuries. Problem of authorship can be of interest not only to humanities researchers, but also to politicians, historians and religious scholars in particular. Thorough investigative journalism, combined with scientific analysis (*e.g., chemical analysis*) of documents has traditionally given good results [4].

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 191–200, 2018.
https://doi.org/10.1007/978-3-319-73500-9_14

Furthermore, the recent development of improved statistical techniques in conjunction with the large availability of digital corpora, have made the automatic and objective inference of authorship a practical and easy task. That is why, this research field has seen an explosion of scholarship, resulting in several related works [5, 6].

Research works on authorship attribution usually appear at several types of debates ranging from linguistics and literature through machine learning and computation, to law and forensics. Despite this interest, the field itself is somewhat in confusion with a certain sense of best practices and techniques [4].

As mentioned above and concerning the different existing related works, despite the large utilization of stylometry in the occidental languages, there are not a lot of articles (relatively) related to Arabic text categorization [7], especially for religious texts.

One can find a couple of recent works of author discrimination in Arabic [8], but very few ones are applied on the Quran: in 2012 for instance, Sayoud presented a series of author discrimination experiments between the holy Quran and Hadith [9]. Once, the author used the two books in their entirety and another time, he segmented the books into 4 segments each. In both experiments he showed that the authors of the two books are different. Later on, he published another article showing an experiment of author discrimination between the holy Quran and Hadith by using a hierarchical clustering [10]. Results were interesting since they sharply showed two important clusters representing the two corresponding authors: Quran author and Hadith author.

In this investigation, we are interested in conducting a stylometric analysis on these two religious books in a larger textual corpus and with several authors. So, in order to enlarge the dataset and increase the number of authors, we have decided to use 7 different books and then 7 different authors (*Quran, Hadith and 5 other religious books*). These experimental conditions are theoretically more consistent for the discrimination/attribution task.

An interesting new idea is the proposal of the Fusion approach, which we applied in two different forms: Fusion of Classifiers (FC) and Fusion of Features (FF). In the knowledge of the author, it is the first time that it has been applied in stylometry with the proposed forms (*i.e. FC and FF*).

2 Corpus of the Seven Religious Books

As cited previously, there are seven different books written by seven different authors: the holy Quran, Hadith and 5 other books written by 5 religious scholars. We recall that the Arabic styles are almost the same (*i.e. Standard Arabic*) for the 7 books, the genre of the books is the same and the topics are also the same (*i.e. Religion*). We called this dataset: *SAB-1* (*Seven Arabic Books – dataset One*). These books are described as follows:

1stbook: the holy Quran, it is considered as the divine book of Islam [11]. The Quran is considered to be written by Allah (God) and only sent down to the Prophet Muhammad fourteen centuries ago (Fig. 1).

2nd book: the Hadith contains the authentic statements and speeches of the Prophet Muhammad in different situations [12]. In this investigation we used the Bukhari Hadith. Moreover, we removed the Quranic verses present in the Hadith to get only pure statements of the Prophet (Fig. 2).

Fig. 1. Old page of the *holy Quran.*

Fig. 2. Old page of the *Hadith.*

3rd book: text collection of Alghazali (*Author: Mohammed al-Ghazali al-Saqqa*): it contains some articles and dissertations of Alghazali. This author is a contemporary Egyptian religious scholar, who is born in 1917 and died in 1996. Sheikh al-Ghazali held the post of Chairman of the Academic Council of the International Institute of Islamic Thought in Cairo.

4th book: text collection of Alquaradawi (*Author: Yusuf al-Qaradawi*): it contains some articles and dissertations of Alquaradawi. This author is a contemporary Egyptian/Qatari religious scholar, who is born in 1926. He is the head of the European Council for Fatwa and Research, an Islamic scholarly entity based in Ireland. He also serves as the chairman of International Union for Muslim Scholars (*IUMS*).

5thbook: text collection of Abdelkafy (*Author: Omar Abdelkafy*). This text collection contains some articles and dissertations of Dr. Omar Abdelkafy, who was born in Almenia, Egypt on May 1, 1951. He memorized the Holy Quran completely when he was ten years old. Dr. Abdelkafy also memorized Sahih Al-Bukhary and Muslim with full references. Abdelkafy studied Islamic Theology and Arabic Linguistics from clever scholars and started serving the Islamic Dawah in 1972.

6thbook: text collection of Al-Qarni (*Author: Aaidh ibn Abdullah al-Qarni*). This text collection contains some articles and dissertations of Shaykh Aaidh ibn Abdullah al-Qarni, who was born in 1960. He is a Saudi religious scholar and author of a famous book. Al-Qarni is best known for his distinguished book "La Tahzan" (*in English: Don't Be Sad*), which had a lot of success over the time.

7thbook: text collection of Amr Khaled (*Author: Amr Mohamed Helmi Khaled*). Several articles and dissertations of Amr Khaled have been collected into a unique text. This author was born in 1967 in Egypt. He is an Egyptian Muslim activist and television preacher. He is often described as "the world's most famous and influential Muslim television preacher".

Those seven books are preprocessed and segmented into different and distinct text segments, and every segment is about 2900 tokens each.

3 Authorship Attribution Methods

Several experiments of Authorship Attribution (AA) are conducted on the 7 segmented religious books. For a purpose of feature selection and evaluation, four types of characteristics are employed: character-trigram, character tetra-gram, word and word-bigram. Two of these features are based on characters and the two others are typically lexical.

Also, four different classifiers are used for the automatic authorship classification (*into ideally 7 different classes*), where every class should represent one particular author. The different classifiers are defined as follows: Manhattan centroid distance [9]; Multi Layer Perceptron NN [13]; SMO based Support Vector Machines [14, 15] and Linear Regression [16, 17].

Furthermore, in this investigation, a Fusion approach is proposed to enhance the attribution accuracy of the conventional classifiers/features.

In order to enhance the authorship attribution performance, we have proposed the use of several classifiers and several features, which are combined in order to get a lower identification error: this combination is technically called Fusion [18].

Theoretically, the fusion can be performed at different hierarchical levels and forms. A very commonly encountered taxonomy of data fusion is given by the following techniques [19, 20, 21]:

- Feature level where the feature sets of different modalities are combined. Fusion at this level provides the highest flexibility but classification problems may arise due to the large dimension of the combined (*concatenated*) feature vectors.

- Score (*matching*) level is the most common level where the fusion takes place. The scores of the classifiers are usually normalized and then they are combined in a consistent manner.
- Decision level where the outputs of the classifiers establish the decision via techniques such as majority voting. Fusion at the decision level is considered to be rigid for information integration [22], but it is not complicated in implementation.

In this investigation, we propose the use of the third technique, namely the decision level based fusion. Furthermore, two types of combinations are employed: combination of features, called FDF *or Feature-based Decision Fusion*, and combination of classifiers, called CDF *or Classifier-based Decision Fusion*.

– **Feature-based Decision Fusion (FDF)**: In the first proposed fusion (*combination of several features*), three different features are employed: Character-tetragram; Word and Word Bigram.

The fusion technique fuses the different corresponding scores of decision into one decision (*the final decision*). The chosen classifier is *Manhatan centroid* because it has shown excellent performances during the previous experiments.

The Feature-based Decision Fusion or FDF (*see* Fig. 3) consists in fusing the outputs of the classifier according to a specific vote provided by the different decisions: each decision concerns one feature *Fj*.

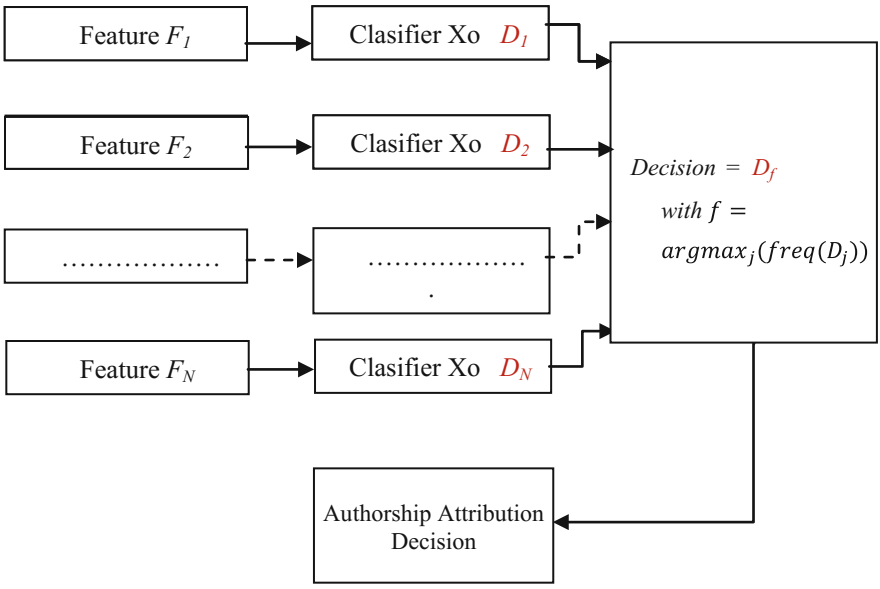


Fig. 3. Principle of the Feature-based Decision Fusion (FDF)

The fused decision D_f of N features is given by the following equation:

$$Decision = D_f, \; with f = argmax_j(freq(D_j)) \tag{1}$$

freq denotes the occurrence frequency of a specific decision and j = 1..N.

– **Classifier-based Decision Fusion (CDF)**: In the second proposed fusion (*combination of several classifiers*), three different classifiers are employed: Manhattan centroid; SMO-SVM and MLP.

As previously, the fusion technique fuses the different corresponding scores of decision into one decision (*the final decision*). Concerning the choice of the features, the *word* descriptor has been used because it has been shown that this type of feature presented relatively good performances during our experiments.

The Classifier-based Decision Fusion or CDF (*see* Fig. 4) consists in fusing the outputs of the different classifiers according to a specific vote provided by their different decisions: each decision concerns one classifier *Cj*.

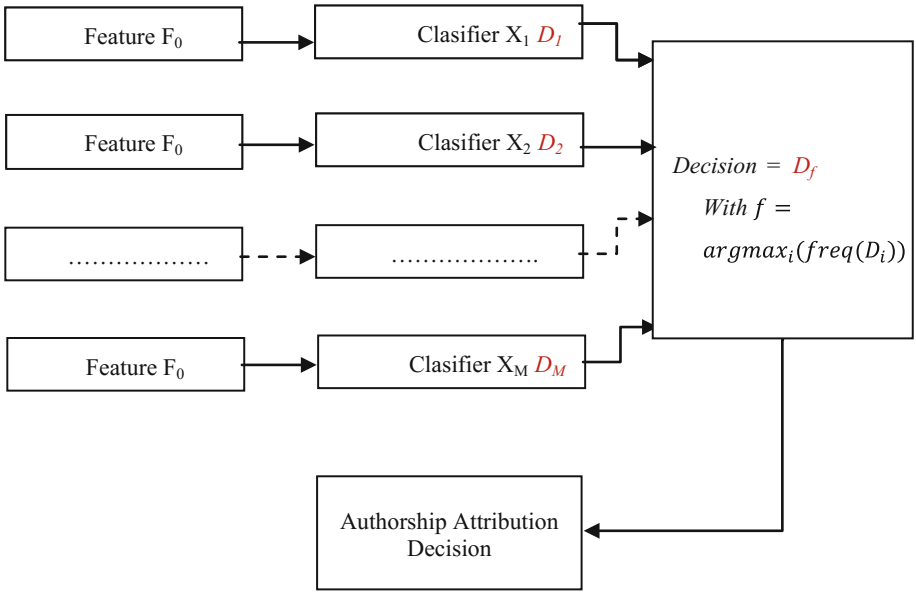


Fig. 4. Principle of the Classifier-based Decision Fusion (CDF)

The fused decision D_f of M classifiers is given by the following equation:

$$Decision = D_f, \; with f = argmax_i(freq(D_i)) \tag{2}$$

freq denotes the occurrence frequency of a specific decision and i = 1..M.

All the results of the fusion approach are represented in Tables 1 and 2, summarizing the corresponding AA scores of the first and second fusion techniques respectively.

Table 1. Error of identification with and without *feature-based fusion (FDF)*

	Total Identification error on the 7 books	Quran book	Hadith book	Aaid's book	Abdelkafy's book	Alghazali's book	Alquar-adawi's book	A-Khaled's book
Char-tetragram	**1.05%**	0%	0%	0%	0%	0%	11.1%	0%
Word	**1.05%**	0%	0%	6.25%	0%	0%	0%	0%
Word-bigram	**1.05%**	0%	0%	0%	0%	3.1%	0%	0%
FDF fusion	**0%**	**0%**	**0%**	**0%**	**0%**	**0%**	**0%**	**0%**

Table 2. Error of identification using the *classifier-based fusion (CDF)*

	Total Identification error on the 7 books	Quran book	Hadith book	Aaid's book	Abdelkafy's book	Alghazali's book	Alquar-adawi's book	A-Khaled's book
Manhattan	**1.05%**	0%	0%	6.25%	0%	0%	0%	0%
SVM	**2.1%**	0%	0%	0%	0%	33.3%	0%	0%
MLP	**1.05%**	0%	0%	0%	16.7%	0%	0%	0%
CDF fusion	**0%**	**0%**	**0%**	**0%**	**0%**	**0%**	**0%**	**0%**

4 Experiments of Authorship Attribution

As mentioned previously, seven Arabic religious books are investigated and analyzed in order to make a classification of the text documents per author: the experimented corpus is called ***SAB-1***. We also recall that several features and several classifiers are used in the experiments of authorship attribution.

We noticed that the best results were obtained by the Manhattan distance and the MLP, which give an error of only 1.05%, the SVM present an identification error of 2.1%. Furthermore, in other experiments (*not mentioned in this paper*) the Manhattan distance outperformed all the evaluated classifiers, showing the high performances of this last one. The three authors: Aaid-Alkarni, Abdelkafy and Alghazali presented some problems of authorship attribution depending on the choice of the classifier. Again, the two first ones are often confused with other authors. We also noticed that the Quran and Hadith books are attributed without any error of attribution (*error of 0%*).

We noticed that Manhatan distance, which is a relatively simple statistical classifier, outperforms the other machine learning classifiers in many cases. However we do know that these last ones are usually better than the distance based classifiers especially for the SVM classifier, which is considered as the state-of-the-art classifier in many research fields. The main possible reason is the low dimensionality of the training dataset, which usually leads to a weak training process (*note that some books are too small with only 8 or 9 text segments per book*).

In order to further enhance the authorship attribution performances, two fusion techniques have been proposed and implemented: the FDF and CDF fusion techniques (as explained in the previous section). In Tables 1 and 2 we can see the corresponding results of those two fusion techniques respectively.

As we can see in the last line of Tables 1 and 2, the authorship attribution error is equal to zero for every author. The total identification score is 100%, showing the superior performances of the fusion techniques over the conventional classifiers/features as expected in theory. This result is very interesting since it shows that a combination of different features and/or classifiers can lead to high authorship attribution performances.

So, the first conclusion one can state is that the fusion approach is quite interesting in multi-classifier or multi-feature authorship attribution. Furthermore, since there are no cross-errors of attribution between the Quran and Hadith texts (*each other*) and more generally, since there was not any cross-error of attribution for the Quran texts or Hadith texts with regards to the 6 other investigated books, we can state that these 2 books are completely different in style each other, and also different from all the other investigated books.

5 Discussion and Conclusion

As described in this paper, several experiments of authorship attribution have been conducted on seven Arabic religious books, namely: the holy Quran, Hadith and 5 other books written by 5 religious scholars. We recall that the Arabic dialect is the same (*i.e.*

Standard Arabic) for the 7 books, the genre of the books is the same and the topic is also the same (*i.e. Religion*).

To conduct these experiments, several features have been proposed: character tri-grams, character tetra-grams, word uni-grams and word bi-grams. On the other hand, several classifiers have also been employed: Manhattan distance, Multi-Layer Perceptron, Support Vector Machines and Linear Regression. Furthermore we have proposed and implemented 2 fusion methods called FDF and CDF to enhance the AA performances.

Results have shown good authorship attribution performances with an overall score ranging from 96% to 99% of good attribution (*depending on the features and classifiers that are employed*) without the use of fusion.

However, this score reaches 100% of good attribution by using the proposed fusion techniques (*FDF and CDF*). This result shows that the fusion approach is interesting and should be strongly recommended for authorship attribution methods that require high degree of accuracy, such as in religious disputes or in criminal investigations.

Concerning the comparison between the Quran and Hadith books, the related results (*of this investigation*) have shown that the Quran texts and Hadith texts are statistically different with a discrimination score of 100% (*i.e. discrimination error of 0%*), with or without using the fusion, and should probably belong to two different Authors, which also implies that the Quran could not be written by the Prophet.

References

1. Signoriello, D.J., Jain, S., Berryman, M.J., Abbott, D.: Advanced text authorship detection methods and their application to biblical texts. In: Proceedings of SPIE (2005), vol. 6039, pp. 163–175. SPIE (2005)
2. Eder, M.: Does size matter? Autorship attribution, short samples, big problem. In: Digital Humanities 2010 Conference, London, pp. 132–135 (2010)
3. Luyckx, K., Daelemans, W.: Authorship attribution and verification with many authors and limited data. In: Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), Manchester, pp. 513–520, August 2008
4. Juola, P.: Authorship attribution. Found. Trends Inf. Retrieval **1**(3), 233–334 (2006). https://doi.org/10.1561/1500000005. Now Publishing, USA
5. Love, H.: Attributing Authorship: An Introduction. Cambridge University Press, Cambridge (2002)
6. McMenamin, G.R.: Forensic Linguistics — Advances in Forensic Stylistics. CRC Press, Boca Raton (2002)
7. Fodil, L., Ouamour, S., Sayoud, H.: Theme classification of arabic text: a statistical approach. In: TKE 2014 Conference: Terminology and Knowledge Engineering, 19–21 June 2014, Berlin, Germany (2014)
8. Baraka, R., Salem, S., Abu-Hussien, M., Nayef, N., Abu-Shaban, W.: Arabic text author identification using support vector machines. J. Adv. Comput. Sci. Technol. Res. **4**(1), 1–11 (2014)
9. Sayoud, H.: Author discrimination between the Holy Quran and Prophet's statements. LLC J. Lit. Linguist. Compt. **27**(4), 427–444 (2012)

10. Sayoud, H.: Authorship classification of two old arabic religious books based on a hierarchical clustering. In: LRE-Rel: language resources and evaluation for religious texts, Lütfi Kirdar Convention & Exhibition Centre Istanbul, Turkey, pp. 65–70 (2012)
11. Ibrahim, I.A.: A brief illustrated guide to understanding Islam. Library of Congress, Catalog Card Number: 97-67654. Published by Darussalam, Publishers and Distributors, Houston (1999). http://www.islam-guide.com/contents-wide.htm, ISBN: 9960-34-011-2
12. Islahi, A.A.: Fundamentals of Hadith Interpretation; Hashmi, T.M. (English Trans.): Mabadi Tadabbur-i-Hadith. Al-Mawrid, Lahore (1989). www.monthly-renaissance.com/DownloadContainer.aspx?id=71
13. Sayoud, H.: Automatic speaker recognition – Connexionnist approach. Ph.D thesis, USTHB University, Algiers (2003)
14. Witten, I.H., Frank, E., Trigg, L., Hall, M., Holmes, G., Cunningham, S.J.: Weka: practical machine learning tools and techniques with Java implementations. In: Kasabov, N., Ko, K. (eds.) Proceedings of the ICONIP/ANZIIS/ANNES 1999 Workshop on Emerging Knowledge Engineering and Connectionist-Based Information Systems, Dunedin, New Zealand, pp. 192–196 (1999)
15. Keerthi, S.S., Shevade, S.K., Bhattacharyya, C., Murthy, K.R.K.: Improvements to Platt's SMO algorithm for SVM classifier design. Neural Comput. **13**, 637–649 (2001)
16. Linear Regression. Last visit in 2013. http://en.wikipedia.org/wiki/Linear_regression
17. Huang, X., Pan, W.: Linear regression and two-class classification with gene expression data. Bioinformatics **19**(16), 2072–2078 (2003)
18. Tchechmedjiev, A., Schwab, D., Goulian, J.: Fusion strategies applied to multilingual features for an knowledge-based word sense disambiguation algorithm: evaluation and comparison. In: CICLING 2013 Conference: 14th International Conference on Intelligent Text Processing and Computational Linguistics, 24–30 March 2013, University of the Aegean, Samos, Greece (2013)
19. Jain, A.K., Ross, A., Prabhakar, S.: An introduction to biometric recognition. IEEE Trans. Circuits Syst. Video Technol. **14**(1), 4–20 (2004)
20. Dasarathy, B.V.: Decision Fusion. IEEE Computer Society Press, Los Alamitos (1994)
21. Verlinde, P.: A Contribution to Multimodal Identity Verification using Decision Fusion. Ph.D thesis, Ecole Nationale Supérieure des Télécommunications, Paris, France, 17 September 1999
22. Stylianou, Y., Pantazis, Y., Calderero, F., Larroy, P., Severin, F., Schimke, S., Bonal, R., Matta, F., Valsamakis, A.: GMM- based multimodal biometric verification. Final Project Report 1, Enterface 2005, 18 July–12 August, Mons, Belgium (2005)

Automatic Identification of Moroccan Colloquial Arabic

Ridouane Tachicart[1(✉)], Karim Bouzoubaa[1], Si Lhoussaine Aouragh[2], and Hamid Jaafa[3]

[1] Mohammadia School of Engineers, Mohamed V University in Rabat, Rabat, Morocco
tachicart@gmail.com, bouzoubaa@emi.ac.ma
[2] Law, Economics and Social Sciences Faculty, Mohamed V University in Rabat, Rabat, Morocco
aouragh@hotmail.com
[3] Polidisciplinary Faculty of Safi, CaddiAyyad University, Safi, Morocco
jaafarhamid1973@gmail.com

Abstract. Language Identification is an NLP task which aims at predicting the language of a given text. For the Arabic dialects many attempts have been done to address this topic. In this paper, we present our approach to build a Language Identification system in order to distinguish between Moroccan Colloquial Arabic and Arabic languages using two different methods. The first is rule-based and relies on stop word frequency, while the second is statically-based and uses several machine learning classifiers. Obtained results show that the statistical approach outperforms the rule-based approach. Furthermore, the Support Vector Machines classifier is more accurate than other statistical classifiers. Our goal in this paper is to pave the way toward building advanced Moroccan dialect NLP tools such as morphological analyzer and machine translation system.

Keywords: Language identification · Arabic dialect · Corpus
Moroccan colloquial dialect · Natural language processing · Language model
Standard arabic · Machine learning · Classifier

1 Introduction

Arabic is one of the most popular languages in the world. Recent numbers put this language at the 4th rank with more than 420 million speakers[1]. Arabic can be categorized to three varieties:

- The first is Classical Arabic (CLA) and can be considered as an oldest variety. It is used in Quran and literary texts before the 9th century.
- The second is Modern Standard Arabic (MSA). It is widely used in formal situations today.
- The third is Colloquial Arabic (CA) or Arabic dialect (AD). It represents the mother tongue of Arabic people and it is used in informal venues.

[1] "World Arabic Language Day | United Nations Educational, Scientific and Cultural Organization". www.unesco.org.

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 201–214, 2018.
https://doi.org/10.1007/978-3-319-73500-9_15

There are many dialects spoken around Arab countries. However, it can be divided to two major groups: "North African or Maghrebi" spoken in the west side of the Arab Nation and "Mashrequi" spoken in the east side. Each group displays several dialects and differs from each other according to Arabic regions. In this paper we focus on the Moroccan Colloquial Arabic (MCA) which is spoken in Morocco and belongs to North African dialects group.

The recent decade experienced an important emergence of using internet in this country. Official estimations reported that only 15% of Moroccan people had access to internet in 2005. However, this estimation significantly increased to 66% in 2016[2]. Therefore, this situation increased the number of Moroccan people using Arabic in internet, especially the colloquial variety MCA (Salia 2011). As a result, we observe several web sites and social network pages in different domains where MCA dominates (moustache.ma, coding-darija.com, baboubi.ma, etc.…).

Nowadays, there is a considerable need to process this dialect because of the important amount of the digital MCA. However, MCA content is affected in various context by code-switching phenomena (Benmamoun 2001), where it is noticed that Moroccan people use MSA and MCA together in the same speech text. In addition, some Moroccan users are influenced by Arabic media (especially Egyptian and Gulf TV and music), which led them to use in some cases Arabic dialects common expressions such as (/Yes/ ايوه كده, /I love you/ انا بحبك). Thus, it is important to distinguish between MCA and other closed varieties (MSA and Arabic dialects) before engaging in MCA processing. This task can be done throughout a language identification system.

Language Identification (LID) is a basic natural language processing (NLP) task which consists of guessing the language of a given text. It is a needed step for preprocessing text towards performing advanced NLP tasks such as Morphological analysis, machine translation, sentiment analysis, etc. LID systems may rely on linguistic rules, statistical methods or mixing these two techniques. In this paper we present a comparative study on Language Identification of the Moroccan Colloquial Arabic texts written in Arabic letters using the two different methods. The first one is human-expertise-based that takes into consideration a list of MCA stop words. While the second one consists of building an MCA corpus from different sources, and then applying a statistical technique using several machine learning techniques such as Support Vector Machines (SVM), Naïve Bayes (NB), and Logistic Regression (LR).

The remainder of the paper is organized as follows: Sect. 2 gives a brief description of MCA. In Sect. 3, we discuss the related works done on Arabic language identification. Then, Sect. 4 describes the building of the MCA corpus that we used in this work as a dataset. In Sect. 5, we present our selected approach and describes the experimental setup. The final section concludes the paper.

[2] https://www.anrt.ma/sites/default/files/rapportannuel/cp-enquete-tic-2015-fr_0.pdf.

2 Moroccan Colloquial Arabic Overview

Beside Tamazight language, MCA is the mother tongue of Moroccan people. However, according to the official census performed in 2014[3], they prefer and tend to use MCA with huge difference compared to Tamazight.

According to (Laghouat 1995) MCA can be divided mainly to four varieties (Fig. 1):

Fig. 1. MCA varieties according to regions

- "Aroubia" dialect: spoken in the Middle Western side of Morocco (near the Atlantic Ocean). This region is characterized by the highest population density in Morocco.
- "Jeblia" dialect: spoken in north of Morocco.
- "Badaouia" dialect spoken in east of Morocco.
- "Hassania" dialect: spoken in south of Morocco.

The other Moroccan regions consider new MCA varieties influenced by the previous varieties. In this work, we focus on the first variety (Aroubia) since it is used and understood by the majority of Moroccan people.

From an historical point of view, MCA is the result of interaction between Arabic and Tamazight when Arabs came to Morocco in spreading Islam period (before the 8th century). From this time until the beginning of the 20th century, MCA was a mixture of Tamazight and Arabic in different levels (lexicon, syntax, morphology and phonology). After the establishing of the French and Spanish protectorate, the relation between

[3] http://rgph2014.hcp.ma.

Moroccan culture on one hand, and French and Spanish on another hand was characterized by cultural cross-fertilization. Hence, this situation led MCA to borrow several features from these languages (rueda /wheel/ رويدة, cosina /kitchen/ كوزينة, Tomobil /car/ طوموبيل). However, instead of the influence of these languages and the MCA evolvement through the ages, it remains strongly influenced by Arabic according to a previous study (Tachicart et al. 2016) especially at the lexical level.

3 Related Works

Many works have attempted to build Arabic dialect LID systems. They adopt different approaches and apply several techniques. In the following, we present and review the most related researches to this topic.

Man and Moustafa built the LAHGA system (Man and Moustafa 2011) which can classify three varieties of Colloquial Arabic namely: Egyptian (EGY), Levantine (LEV) and Maghrebi dialects (MAG). They collected and pre-processed Arabic dialect tweets in order to prepare a dataset for training and testing LAHGA. Then, they manually extracted features by reading collected tweets. Authors trained their classifier using three machine learning techniques: Naïve Bayes, Logistic Regression and SVM. The testing phase was divided into manual test that reached 90% of performance, and cross-validation test that reached only 75% of performance.

Another research study (Elfardy and Diab 2013) focused on sentence level to distinguish between MSA and EGY. The authors adopted a supervised approach using the Naïve Bayes classifier. They used WEKA toolkit (Smith and Frank 2016) to train and cross-validate the classifier on a portion of an annotated dataset built in another work (Zaidan and Callison-Burch 2011). This dataset contains comments on Egyptian news articles and divided to 300k tokens for each class (EGY or MSA). The system achieved an accuracy of 85%.

Efardy (Elfardy et al. 2014) proposed AIDA system which is designed to identify code switching between MSA and EGY. Their approach relies on language modeling and morphological analysis (hybrid system). First, they prepared an annotated dataset with morphological tags. It is composed of web data (8000k words) and collected tweets (120k words). The dataset was splitted to train and test set. Then, they used SRILM toolkit (Andreas 2002) to build a language model where the goal is to find the best sequence of tags for a given sentence. By using MADAMIRA morphological analyzer (Pasha et al. 2014), AIDA can perform on token level and reached an accuracy of 93,6%.

Another study (Malmasi et al. 2015) focused also on sentence-level by performing supervised classification of Arabic dialects. They used 1k sentences of a Multidialectal Parallel Corpus (Bouamor et al. 2014) covering six classes: MSA, EGY, Tunisian (TUN), Syrian (SYR), Jordanian (JOR) and Palestinian (PAL). Their experiments consist of performing multi-class classification based on linear SVM using LIBLINEAR SVM package (Fan et al. 2008). The overall accuracy achieved 74%.

In the work of Sadat (Sadat et al. 2014), authors performed two experiments in order to identify 18 different Arabic dialects. The first experiment relies on n-gram language model, while the second uses the Naïve Bayes machine learning. They trained

their system using a data set collected from Arabic dialect web pages. As a result, they showed that the Naïve Bayes classifier based on character bi-gram performs better with 98% of accuracy.

In the work of (Alshutayri et al. 2016) authors used WEKA to develop an Arabic dialect identification system. They used transcript text as dataset containing between 1000 and 1700 utterances for each class (MSA and Arabic dialects). On this dataset, they trained and tested several classifiers such as: Naïve Bayes, ZeroR, J48, Jrip and Sequential Minimal Optimization (SMO). The best result was achieved using SMO classifier which rates to 42,85% of accuracy.

While previous studies worked on sentence and word level to identify the language, the work of (Belinkov and Glass 2016) focused on character-level using Neural Network model. The latter embeds a sequence of input characters in vector space and generates the sequence representation (used to predict the language) via multiple layers. The system accuracy reached almost 60% when experimenting on a dataset containing MSA, MAG, EGY, LEV and Gulf Arabic text.

In a recent paper related to MCA (Samih and Maier 2016) which consists of word-level language identification using Conditional Random Fields Classifier (CRF). Authors ran an automatic process to collect MCA text from web pages and used CRF in addition to character 5-gram language model to run several experiments. The best system accuracy reached 91.4%.

In the last work (Adouane and Dobnik 2017), authors presented a system that identify the language at word level. They considered six classes: Algerian Arabic (ALG), MSA, French, English, Berber, and mixed languages. They built and annotated (in word level) a corpus containing 215k tokens where ALG class represents 54%. Then, they used a supervised machine learning with HMM[4] and N-gram classification tagging to test and train their system which achieved an accuracy of 93,14%.

Finally, Table 1 below Highlights basic specifications of each surveyed LID system. By examining these specifications we can synthesize with the following comments:

- The topic of Language Identification is recent since researches began to address this topic recently.
- Several Arabic dialects were addressed and EGY is the most Arabic dialect processed.
- Existing works tried several methods and achieved good accuracy in general.
- Due to time consuming of the rule-based approach, researchers tend to adopt the statistical approach using classifiers especially the one focusing on sentence level. However, the latter needs to ensure first necessary resources to run experiments.
- Only one work addressed MCA and reached an accuracy of 91%. However, in addition to the unavailability of neither the system nor the data, the followed rules to annotate MCA in the training data do not match to the MCA rules that we adopt. For example, the verb غاتمشي /You will go/ is annotated as word with mixed morphology. In contrast, we consider it as an MCA word in our work since it is originated from MSA but adapted to MCA rules.

[4] Hidden Markow Model.

Table 1. Arabic dialect language identification systems

LID system	Approach	Method	Dialects	Accuracy
(Man and Moustafa 2011) LAHGA	Statistical classification based on sentence level	Naïve Bayes, Logistic Regression and SVM classifiers	EGY, LEV and MAG	90%
(Elfardy and Diab 2013)	Statistical classification based on sentence level	Naïve Bayes classifier using WEKA	MSA and EGY	85%
(Elfardy et al. 2014) AIDA	Hybrid system	Language model using SRILM and morphological analysis using MADAMIRA	MSA and EGY	93%
(Sadat et al. 2014)	Statistical classification based on sentence level	Naïve Bayes classifier and N-gram language model	18 Arabic dialects	98%
(Malmasi et al. 2015)	Statistical classification based on sentence level	SVM classifier using LIBLINEAR	MSA, EGY, TUN, SYR, JOR and PAL.	74%
(Alshutayri et al. 2016)	Statistical classification based on sentence level	Naïve Bayes, ZeroR, J48, Jrip and SMO	EGY, Gulf, LEV, MAG and MSA	42%
(Belinkov and Glass 2016)	Statistical classification based on character level	Neural Network classifier	MSA, MAG, EGY, LEV and Gulf	60%
(Samih and Maier 2016)	Statistical classification based on word level	CRF classifier	MSA and MCA	91%
(Adouane and Dobnik 2017)	Statistical classification based on word level	HMM and N-gram classifiers	MSA and ALG	93%

As mentioned in the surveyed works above, one of the biggest practical challenges that face the building of LID systems is ensuring enough training data. In the next section we address this challenge related to our work.

4 Dataset

In fact, we believe that the availability of an MCA corpus is necessary to our work. However, its lack (inexistent, ongoing or not free) led us to create our own MCA corpus. In this section, we present the MCA corpus that we built in order to serve as a dataset to train and test our system.

4.1 MCA Corpus Creation

We have used various sources of MCA text in order to ensure a good coverage of MCA throughout different topics. To this end, we selected three different sources of MCA text: internet, literature and recorded conversation between Moroccan people.

Collecting MCA web text
Currently, MCA web text is very close to Moroccan daily language. For this reason, we first scraped data text from popular Facebook Moroccan pages in different topics for which we knew that MCA dominates either in posts or users comments. For this purpose, we used a page data scraper[5] implemented as Python 2.7 scripts. In total, we collected about 50 documents (.csv files) containing 99k posts and 4000k comments. In addition to Facebook platform, we automatically collected MCA pages from some Moroccan web sites such as goud.ma. The final sub-corpus (C1) is composed of 87k MCA sentences (Fig. 1) containing about 1100k tokens.

Collecting literature text
We used TV series text and written plays that are intended for theatrical purposes. It consists mainly of (.txt) files containing dialogues between characters (actors). This dataset is the second sub-corpus (C2) and was also preprocessed (as C1) and updated to keep only Arabic text with 9k sentences (Fig. 2).

Transcript recorded conversation
Finally, we recorded some life conversations between Moroccan people in different contexts (restaurant, traveling, phone calls, etc.…) to obtain the third sub-corpus (C3) with 1,5k sentences. Note that since our goal is to launch the statistical learning to quickly detect the text language among MSA and MCA, we considered only limited constraints (sentence length, total size) in order to build this corpus. Involving specialists in linguistics will be taken in consideration in a future work.

At this stage, and till performing the above tasks, our preliminary MCA corpus is sorted by sentences and contains 130k sentences. However, we decided to reduce its size due to the noisy data through an automatic process. The latter was performed by removing Non-Arabic text, punctuation, emoticons and diacritics. We checked and reviewed the final MCA corpus since we are native speakers of MCA and kept only 34k MCA sentences with 370k tokens. This MCA corpus is freely available as TEI format[6].

[5] https://github.com/minimaxir/facebook-page-post-scraper.
[6] http://arabic.emi.ac.ma:8080/MCAP.

(C1) – Sentence 1245

اه هو برهوش ، لكنه قدرها وكبر بيها وتزوجها

He is so young; however he respected her and got married.

(C2) – Sentence 35

عبدالله :اعطيني النمرة تاع الحاج قاسم أولدي..تبعني ..أنا اباك وتنعرف أكتر منك..

Abdellah: Son, give me Hadj Kacem phone number, follow me, I'm your father and I know better than you

(C3) – Sentence 235

عيط ليك اخويا باش نقولك مغانمشيش معاكم غدا للماتش غي شوفو كيديرو و ديو معاكم شي واحد بلاصتي

I'm calling you to inform you about the match of tomorrow. I can't go with you . you can search for another person to replace me.

Fig. 2. Samples of MCA corpus

4.2 Data Annotation

Since our goal in this work is only to distinguish between MCA and MSA, we considered two classes (MCA and MSA) for the data annotation. Hence, we used the full MCA corpus above with 34k sentences to label it with the tag 'MCA'. In addition, we used a portion of the MSA Watan-2004 corpus[7] containing 53k sentences with 660k tokens labeled with 'MSA' tag. The final dataset prepared to train and test our system consists of 100k sentences where MSA represents 66% and MCA represents 34% (statistics in Table 2). Moreover, the average of count words composing each sentence is 11.

Table 2. Statistics about annotated corpus

	MSA	MCA
#Sentences	66000	34000
#Words	1220000	390000

5 Methodology and Experiments

5.1 Method

We formulate the task as a binomial classification problem, for which MSA and MCA are the binary classes. First we used linguistic rules (rule-based approach) to build and

[7] http://arabic-corpus.soft112.com/download.html.

test the first classifier. Then we used a supervised approach in order to try several statistical techniques commonly used for binary classification including Naive Bayes, Support Vector Machines and Logistic Regression.

Rule-based classifier

The proposed approach relies on stop word lists which has been shown to be effective for LID tasks because these words are discriminative for a given language (Peters et al. 2012).

To build the corresponding classifier, we apply an algorithm based on stop words frequencies. Given that input text may include MSA and MCA content, we calculate the frequencies elements of two lists: the first contains MCA stop words that we have prepared (sample on Table 3), while the second gathers MSA ones extracted from the work of (Namli et al. 2015). In general, the term can be either a simple stop word or the latter combined with MCA or MSA affixes. Each list receives a score calculated by summing the frequencies of their elements in the input text. The highest score furthers the corresponding class and then this class is selected as the language of the input text. In case the scores are equivalent, the highest frequency element in each list is considered. Otherwise, the text language cannot be identified.

Table 3. MCA stop words

MCA stop word	Type	Translation
على	Stop word	On
ماعليهمش	Combined Stop word	Without blame
ديال	Stop word	Their
بديالهم	Combined Stop word	With their
كاع	Stop word	All
وبكاع	Combined Stop word	And with all

Supervised Learning

Machine Learning is key to many interesting problems. For the LID tasks, several Machine Learning techniques have shown great promise. For this reason, we selected for our experiments three popular classifiers suited to binary classification namely:

Naive Bayes classifier

The Naive Bayes Classifier relies on Bayesian theorem with strong independence assumptions between the features. Since our dataset is composed of two classes (MSA and MCA), the first step is to transform this data to word vector in order to estimate the parameters necessary of classification. This classifier relatively requires little data training and usually gives good results despite its simplicity. Moreover, the classifier model is fast to build and can be modified with new MCA or MSA data without having to rebuild the model.

SVM classifier

The main goal of this technique is to optimally separate MCA data from MSA content by building N-dimensional hyper plane in a multidimensional space. SVM can perform

both regression and classification tasks. In this work we consider the second task (classification) which is suited to LID systems.

Logistic Regression classifier
This linear method relies on the Logistic function used at the core of this classifier. First, the function models the probability that an input (text) belongs to the default class MSA (Y = 1) as in Eq. 1.

$$P(\text{text}) = P(Y = 1|\text{text}). \quad (1)$$

Then, it transforms the probability computed into binary values in order to distinguish between MSA and MCA input.

5.2 Experimental Setup

Rule-based
We got the idea of building a seed list (SL) including MCA stop words. SL is composed of simple stop words in addition to their combination with MCA affixes. First, simple stop words were extracted from MDED lexicon (Tachicart et al. 2014). Then, their possible combinations with affixes were formulated. Finally, a linguist validated this list and kept only 360 MCA stop words. A sample of SL is presented in Table 3 above. To illustrate the processing of the rule based method based on this SL, let us consider the sentence شكون اللي قالك أنا مازال خدام في الشركة/Who told you that I am still working in the company/. It contains 5 MCA stop words (شكون, اللي, أنا, مازال and في) and only 3 MSA stop words (أنا, مازال and في). Thus, the detected language is MCA regarding the previous rules.

Supervised Learning
To ensure our experiments following the supervised learning, we used WEKA data analytic tool (Smith and Frank 2016) which provides several features for machine learning classifiers. It is Java-based, available for free under GNU license and has a general API that can be embedded in other applications.
To begin our experiments, we converted first our dataset to word vector in order to extract words as features from dataset sentences. Then we proceeded to the training phase.

The training was performed applying first the NB classifier which gives results quickly, then the SMO implementation of SVM and Logistic Regression classifiers. After building corresponding models of these classifiers, we used resampling methods like cross validation and train-test splits to estimate the skill of these models when making predictions on new data.

The first is using 10 cross-validation. In this process, the dataset is randomly divided into 10 equal sized sub-datasets. Then, a single dataset is retained as the validation data for testing the built models. On another hand, the remaining 9 sub-datasets are used as training data when repeating the validation 10 times with each of the 10 sub-datasets used exactly once as the validation data. The 10 results from the folds can then be averaged to produce a single estimation.

The second is to split our dataset into training set (9/10 of the data) and test set (1/10 of the data) in order to quickly evaluate the performance of the classifiers.

5.3 Results and Discussion

After running the two types of testing methods cited above, we realized that in general results were promising. Table 4 gives the rule-based classifier results evaluated on the test set (1/10 of the data). While Tables 5 and 6 show the sentence-level evaluation on the three supervised learning algorithms using the two test methods. In addition to common performance measures (Precision, Recall, F-Measure, etc.…), we considered the Kappa statistic (Jean 1996) which is suitable for classification tasks. Note that our LID system (including the rule-based and the statistical approach) is available as a public interface[8].

Table 4. Rule-based classifier results

Kappa	Precision	Recall	F-Measure	Accuracy (%)
0,64	0,821	0,842	0,831	**83.84**

First point that we can discuss is that the statistical approach outperforms the rule-based one. This is to be expected since only stop word frequency was considered in order to build the classifier of the rule-based approach. Including advanced rules such as morphological analyzer will extend the processing to full text instead of only stop words, and thus increase rule-based classifier performance. Unfortunately, this tool currently lacks for the MCA.

Table 5. Detailed accuracies of different classifiers using 10-fold cross validation

	NB	LR	SVM
Kappa	0.69	0.90	0.91
Mean absolute error	0.15	0.04	0.04
Root mean squared error	0.36	0.20	0.16
Relative absolute error (%)	34.52	9.14	11.14
Root relative squared error (%)	77.65	42.76	34.33
Precision	0,888	0,961	0,961
Recall	0,851	0,960	0,960
F-Measure	0,855	0,960	0,960
Accuracy (%)	**85.06**	**95.97**	**96.02**

On one hand, and looking at supervised learning results, we got different results for the same training dataset when evaluating different classifiers. This may be explained

[8] http://arabic.emi.ac.ma:8080/MCAP.

Table 6. Detailed accuracies of different classifiers using splitted test set

	NB	LR	SVM
Kappa	0.69	0.91	0.91
Mean absolute error	0.15	0.03	0.046
Root mean squared error	0.36	0.19	0.15
Relative absolute error (%)	34.51	8.35	10.47
Root relative squared error (%)	77.64	40.87	32.22
Precision	0,888	0,964	0,965
Recall	0,851	0,963	0,964
F-Measure	0,855	0,963	0,964
Accuracy (%)	**85.08**	**96.31**	**96.42**

Table 7. Precision and recall of different classifiers according to classes

Eval	Classifier	Class	Precision	Recall
Splitted test	NB	MCA	0,690	0,973
		MSA	0,983	0,786
	SVM	MCA	0,921	0,959
		MSA	0,980	0,960
	LR	MCA	0,918	0,961
		MSA	0,981	0,958
Cross validation	NB	MCA	0,695	0,972
		MSA	0,983	0,792
	LR	MCA	0,919	0,961
		MSA	0,981	0,959
	SVM	MCA	0,923	0,959
		MSA	0,979	0,961

Table 8. Average of precision and recall according to classes

	Precision	Recall
MCA	0,846	0,965
MSA	0,981	0,903

by the fact that machine learning algorithms are stochastic and this behavior of different performance on the same dataset is to be expected. Moreover, our best setup is reached by the SVM classifier following the second test method with an accuracy of 96.42%. This classifier outperforms all other classifiers in each test method.

On the other hand, and looking at Tables 7 and 8 which presents different results according to classes, we found out that the MCA had the highest recall (0,973) and the lowest precision (0,690). The recall of this class reached an average of 0,965 while the precision was 0,846 on average. In fact, the Moroccan Colloquial Arabic has distinctive features in spite of it is much influenced by Arabic. As we mentioned in Sect. 2, the

lexical similarities between MCA and MSA led to common features which explain the lower precision. However, some MCA features raised by adapting lexical Arabic features to Tamazight phonology resulting discriminative features with new morphology. Hence, it was relatively easy to distinguish MCA using these features whether for the trained classifier or the rule-based classifier. This explains the high recall.

In contrast, the MSA had the lowest recall (0,786) but the highest precision (0,983). The precision reached an average of 0,981 and the recall was 0,903 on average. This is due to the ambiguity existing between MSA and MCA features. This situation is explained by the fact that MCA borrows an important amount of its lexicon from MSA.

6 Conclusion

In this work, we explored two approaches in order to build a LID system discriminating between MCA and MSA. We followed first the rule-based approach to implement its classifier. The latter uses two stop word lists of MCA and MSA in order to identify the MCA text reaching an accuracy of 83,84%. In the second task, we experimented with three machine learning techniques and showed that the SMO implementation of SVM classifier perform quite well regarding the Naïve Bays and LR results. Future directions of this work include improving our MCA corpus by covering new domains, increasing its size and considering new constraints with respect to linguistic principles. Moreover, we plan to deal with MCA written in both Latin and Arabic letters and detect code-switching in the same text.

References

Adouane, W., Dobnik, S.: Identification of languages in Algerian Arabic. In: The Third Arabic Natural Language Processing Workshop (WANLP), pp. 1–8. Association for Computational Linguistics, Valencia (2017)

Alshutayri, A., Atwell, E., Alosaimy, A., Dickins, J., Ingleby, M., Watson, J.: Arabic language WEKA-based dialect classifier for Arabic automatic speech recognition transcripts. In: The Third Workshop on NLP for Similar Languages, Varieties and Dialects, Osaka, Japan, pp. 204–211 (2016)

Andreas, S.: SRILM — an extensible language modeling toolkit. In: The ICLSP Conference, Denver, USA, pp. 901–904 (2002)

Belinkov, Y., Glass, J.: Character-level Convolutional Neural Network for Distinguishing Similar Languages and Dialects. CoRR, abs/1609.07568 (2016)

Benmamoun, E.: Language identities in morocco in a historical context. Stud. Linguist. Sci. **31**, 95–106 (2001)

Bouamor, H., Habash, N., Oflazer, K.: A multidialectal parallel corpus of Arabic. In: LREC 2014, pp. 1240–1245 (2014)

Elfardy, H., Diab, M.: Sentence level dialect identification in Arabic. In: 51st Annual Meeting of the Association for Computational Linguistics, Sofia, Bulgaria, pp. 456–461 (2013)

Elfardy, H., Al-Badrashiny, M., Diab, M.: AIDA: identifying code switching in informal Arabic text. In: EMNLP 2014: Conference on Empirical Methods in Natural Language Processing, Doha, p. 94 (2014)

Fan, R.-E., Chang, K.-W., Hsieh, C.-J., Wang, X.-R., Lin, C.-J.: LIBLINEAR: a library for large linear classification. J. Mach. Learn. Res. **9**, 1871–1874 (2008)

Jean, C.: Assessing agreement on classification tasks: the kappa statistic. Comput. Linguist. **22** (2), 249–254 (1996)

Laghouat, M.: L'espace Dialectal Marocain, sa structure actuelle et son évolution récente. Dialectologie et Sciences Humaines au Maroc, pp. 9–41. Faculté des Lettres Mohammed V, Rabat (1995)

Malmasi, S., Refaee, E., Dras, M.: Arabic dialect identification using a parallel multidialectal corpus. In: Hasida, K., Purwarianti, A. (eds.) Computational Linguistics. CCIS, vol. 593, pp. 35–53. Springer, Singapore (2016). https://doi.org/10.1007/978-981-10-0515-2_3

Man, L., Moustafa, M.: LAHGA: Arabic dialect classifier. In: IR 2011, Colorado (2011)

Namli, D., Bouzoubaa, K., Tajmout, R., Tahir, Y., Khamar, H.: A complex Arabic stop-words list design. In: 2ème Journée Doctorale Nationale sur l'Ingénierie de la Langue Arabe (JDILA 2015), Fes (2015)

Pasha, A., Al-Badrashiny, M., ElKholy, A., Eskander, R., Diab, M., Habash, N., et al.: MADAMIRA: a fast, comprehensive tool for morphological analysis and disambiguation of Arabic. In: The Ninth International Conference on Language Resources and Evaluation, Reykjavik (2014)

Peters, C., Braschler, M., Clough, P.: Multilingual Information Retrieval: From Research To Practice. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-23008-0

Sadat, F., Kazemi, F., Farzindar, A.: Automatic identification of Arabic language varieties and dialects in social media. In: Second Workshop on Natural Language Processing for Social Media (SocialNLP), Queensland, Australia, pp. 35–40 (2014)

Salia, R.: Between Arabic and French Lies the Dialect: Moroccan Code-Weaving on Facebook. Thesis, Columbia University (2011)

Samih, Y., Maier, W.: Detecting code-switching in moroccan Arabic social media. In: SocialNLP @ IJCAI-2016, New York (2016)

Smith, T.C., Frank, E.: Introducing machine learning concepts with WEKA. In: Mathé, E., Davis, S. (eds.) Statistical Genomics. MMB, vol. 1418, pp. 353–378. Springer, New York (2016). https://doi.org/10.1007/978-1-4939-3578-9_17

Tachicart, R., Bouzoubaa, K., Jaafar, H.: Building a moroccan dialect electronic dictionary (MDED). In: 5th International Conference on Arabic Language Processing CITALA, Oujda (2014)

Tachicart, R., Bouzoubaa, K., Jaafar, H.: Lexical differences and similarities between Moroccan dialect and Arabic. In: 4th IEEE International Colloquium on Information Science and Technology (CiSt), Tanger (2016)

Zaidan, O., Callison-Burch, C.: The Arabic online commentary dataset: an annotated dataset of informal Arabic with high dialectal content. In: 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, Portland, Oregon, USA, pp. 37–41 (2011)

Arabic NLP Tools and Applications

Encoding Prototype of Al-Hadith Al-Shareef in TEI

Hajer Maraoui[1(✉)], Kais Haddar[2], and Laurent Romary[3]

[1] Faculty of Mathematical, Physical and Natural Sciences of Tunis, MIRACL Laboratory, University of Sfax, Sfax, Tunisia
hajer.maraoui@gmail.com
[2] Faculty of Science of Sfax, MIRACL Laboratory, Sfax, Tunisia
kais.haddar@yahoo.fr
[3] Inria, Team ALMAnaCH, Paris, France
laurent.romary@inria.fr

Abstract. The standardization of Al-Hadith Al-Shareef can guarantee the interoperability and interchangeability with other textual sources and takes the processing of Al-Hadith corpus to a higher level. Still, research works on Hadith corpora had not previously considered the standardization as real objective, especially for some standards such as TEI (Text Encoding Initiative). In this context, we aim at the standardization of Al-Hadith Al-Shareef on the basis of the TEI guidelines. To achieve this objective, we elaborated a TEI model that we customized for Hadith structure. Then we developed a prototype allowing the encoding of Hadith text. This prototype analyses Hadith texts and automatically generates a standardized version of the Hadith in TEI format. The evaluation of the TEI model and the prototype is based on Hadith corpus collected from Sahih Bukhari. The obtained results were encouraging despite some flaws related to exceptional cases of Hadith structure.

Keywords: Hadith text · TEI model · Standardization · Prototype

1 Introduction

The processing of Al-Hadith Al-Shareef has always been a center of academic interest. On the one hand, this interest is due to the importance of Al-Hadith Al-Shareef in Islamic law. It is the second fundamental source, after the Quran, of Islamic legislation. On the other hand, linguistic researches on Arabic language define Al-Hadith corpus as one of references for classical Arabic from the pre-Islamic era. Many studies carried out on the Al-Hadith corpus especially in linguistic analyses and information retrieval. However, the representation of such corpora poses serious structuring and text unification problems, which require their standardization (or normalization). This could lead to a new presentation of the Text based upon a descriptive and detailed annotation for each of its parts. A normalized corpus allows also the compatibility and the interchangeability between NLP applications. Indeed, the normalization of Al-Hadith Al-Shareef can bring the automatic processing of such corpus to another level.

However, normalizing Al-Hadith corpus is a complex task. In fact, it requires a specific model customized for standardizing the Hadith text. This task requires the

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 217–229, 2018.
https://doi.org/10.1007/978-3-319-73500-9_16

selection of the standardization model that harmonizes with Arabic language in general and Al-Hadith structure specifically. Moreover, to realize a normalized Hadith corpus, an automatic encoding process can facilitate this task.

Our main objective is the normalization of Al-Hadith Al-Shareef. To realize this, we start with a deep study on Hadith text structure. Furthermore, to reach the normalization of Al-Hadith Al-Shareef, we apply the Text Encoding Initiative (TEI) (http://www.tei-c.org/) guidelines. We elaborate a TEI customized model for encoding Hadith text. Also, to create a normalized Hadith corpus, we make a prototype to create automatically an encoded version of Al-Hadith text with TEI structure.

In the present paper, we begin with a state of the art on Al-Hadith Al-Shareef. Second, we continue with an overview on TEI guidelines. Third, we present our model for encoding Hadith text. Then, we present our prototype for constructing a normalization of Hadith texts with TEI structure. This section is followed by an evaluation step. We cloture our paper with a conclusion and some perspectives.

2 State of the Art on Al-Hadith Al-Shareef

Hadiths (or prophetic traditions) are narrations on the life and deeds of Prophet Muhammad (peace and blessing upon him), which report what he said or did, or of his implicit approval of something said or done and by itself define what is considered good, by providing details to regulate all aspects of life in this world and to prepare people for the beyond, clarifying the Qur'anic shades. The traditional Muslim schools of jurisprudence regarding Hadith constitute an important tool for understanding Qur'an and in all matters related to jurisprudence [1, 2].

The Hadith consists of two parts: the actual narration, which called *Matn* (المتن); and the chain of narrators who has transmitted the narration, known as *Isnad* (إسناد). The *Isnad* consists of a short or long chronological list of the narrators, each mentioning the one from whom he heard the Hadith all the way to the prime narrator of the *Matn* followed by the *Matn* itself [1].

Research in the *Isnad* is very important in the science of Hadith. Islamic scholars have agreed that *Isnad* is required to prove the accuracy and the soundness of the Hadith which means that any flaw in the chain of transmitters lead to the negation of the Hadith. In order to know whether the Hadith is authentic or not, the Hadith scholars follow clear steps in the judgment on the Hadith *Isnad* that considered as traditional methods. Sahih Bukhary and Sahih Muslim are the recognized collection of authentic assortment of the *Sunna* [1, 3–5].

Nowadays, software tools help judge the Hadith *Isnad* like electronic Hadith encyclopedias and some websites. Additionally, information retrieval and search engines that related to semantic web can used to serve in deciding the degree of the Hadith *Isnad*. Scholars such as Al-Albani have agreed and encouraged using computers and programs in serving religion and Hadith [1].

There are many projects handled with Hadith corpus. Indeed, these projects focused on several branches of researches such as Hadith ontology, linguistic analyzing, Hadith segmentation, authorship attribution, classification and the mining of information.

In [5], the researchers proposed a model for the unsupervised segmentation and the linguistic analysis of the Arabic texts of Hadith. This model is named SALAH. The model automatically segments each text unit in a transmitter chain *Isnad* and text content *Matn*. A tailored, augmented version of the AraMorph morphological analyzer (RAM) analyzes and annotates lexically and morphologically the text content. A graph with relations among transmitters and a lemmatized text corpus, both in XML format, are the final output of the system.

In [1], the author constructed an ontology-based *Isnad* Judgment System (IJS) that automatically generates a suggested judgment of Hadith *Isnad*. It based on the rules that Hadith scholars follow to produce a suggested judgment. A prototype of the approach implemented to provide a proof of concept for the requirements and to verify its accuracy.

Authors of paper [6] built a domain specific ontology (Hadith Isnad Ontology) to support the process of authenticating *Isnad*. They evaluate the ontology through Hadith example and DL-Queries.

Author of paper [7] compared the effectiveness of four different automatic learning algorithms for classifying Hadith corpus into 8 selective books depending on Sahih Bukhary. The automatic learning algorithms are Rocchio algorithm, K-NN algorithm (K- Nearest Neighbor), Naïve Bayes algorithm and SVM algorithm (Support Vector Machines).

In [2], the authors reported on a system that automatically generates the transmission chains of a Hadith and graphically display it. They involve parsing and annotating the Hadith text and identifying the narrators' names. They use shallow parsing along with a domain specific grammar to parse the Hadith content.

In [8], the author experimented author discrimination techniques between the Qur'an and the Hadith. The Qur'an is taken in its entirety, whereas for the Prophet's statements, the researcher chose only the certified texts of Sahih Bukhari. Three series of experiments are done and commented on. The author's investigation sheds light on an old enigma, which has not been solved for 14 centuries: in fact, all the results of this investigation have shown that the two books should have two different authors.

In [9], the researchers reimplemented and evaluated the methods of artificial intelligence using a single dataset. The result of the evaluation on the classification method reveals that neural networks classify the Hadith with 94% accuracy. The Hadith mining method that combines vector space model, Cosine similarity, and enriched queries obtains the best accuracy result.

3 The Text Encoding Initiative

The Text Encoding Initiative (TEI) is an international project aiming at the development of a set of standards for the preparation and the exchange of electronic texts. The TEI was founded in November 1987 by a group of international text database leaders. TEI was created officially in 1988 under the aegis of the ACH[1], the ACL[2] and the ALLC[3] [1].

[1] Association for Computers and the Humanities.

[2] Association for Computational Linguistics.

[3] Association for Literary and Linguistic Computing.

The publication of the works of the various committees' results was in the form of "Guidelines" which have been developed and are maintained by the Text Encoding Initiative Consortium [10]. The TEI guidelines recommend suitable ways to represent the features of textual resources using a set of XML elements in order to elicit the text structure and simplify its digital processing. Indeed, these guidelines present conventions of usable coding in several domains and can be applied to texts in any natural language, of any date, in any literary genre or text type [10]. Moreover, they can be applied as well to create new information that to exchange existing information.

TEI annotation is based on a "patrimonial" transcription of the text, interested in giving as much information as possible while allowing automatic processing for language searches, historical data, different versions of the document and variations level of accuracy that can be adapted to the desired searches [11]. TEI offers the possibility of multilingual uses of the structural description. The universality of the elements allows the compatibility of the analyzes, and the digitized sources within such frameworks as the BVH[4] and belong to all encyclopedic domains, in several ancient and modern languages.

The rules and recommendations made in these guidelines are expressed in terms of the extensible Markup Language (XML) so a TEI document has to comply with XML coding rules. One of the main advantages of the XML language is that it is possible to solve by encoding the very large typographic and textual variation of the documents: this flexibility should not, however, be a hindrance to the acquisition and encoding of textual corpora.

The fundamental structure of a TEI document describes the textual part of the text. In what follows, we give overviews onto the structure and the TEI representation of documents. The basic structure of a document encoded with TEI is represented as follow:

```
<TEI xmlns="http://www.tei-c.org/ns/1.0">
  <teiHeader>
    <!-- Header properties; meta-data -->
  </teiHeader>
  <text>
    <front>
      <!-- front information -->
    </front>
    <body>
      <!-- main body -->
    </body>
    <back>
      <!-- back information -->
    </back>
  </text>
</TEI>
```

[4] The laboratory LI-RFAI (director Jean-Yves Ramel) and the consortium Navidomass (ANR project 2007–2009).

Any textual document encoded with TEI includes a document header, with the <teiHeader> element, and a text part within the <text> element. TEI header contains all the information analogous to that provided by the title page of a printed document. It has four parts: a bibliographic description of the machine-readable text, a description of the way it has been encoded, a text profile, and a revision history [11–13].

A TEI document can contain five textual elements: <text>, <front>, <group>, <body> and <back>. Only <text> and <body> are the obligatory elements. The use of the <front>, <group> and <back> is optional. The <front> element is defined to cluster together the pieces located before the beginning of the text itself. The <back> element is used in case the document contains an annex in the back of the text. The <group> element is specified to include several texts in collections. The <body> is included in the text and it contains the core text of the document [11].

Personalization is a central aspect of TEI using. There are three methods of customization in TEI: TEI Lite, web application Roma and TEI ODD. TEI Lite was originally designed as a demonstration of the customization mechanism. The Roma web application was introduced to select TEI modules that manipulate the elements. As for TEI ODD language, it essentially allows the manual specification of the TEI models, allowing the modification or addition of new elements [10].

Encoding quotation with TEI

Quotation marks are conventionally used to indicate certain elements appearing in a text, the most frequent case is for the quotation. However, the marking of the underlying logical element (for example, a quotation or a piece of direct speech) in the text is recommended, rather than just recording quotation marks in the text [10–12]. The following TEI elements are specified for the encoding of quotation and narration which can be adaptable with the structure of Al-hadith *Matn*:

- **<q>** (or quoted) contains material which is distinguished from the surrounding text using quotation marks or a similar method. It contains a citation or an apparent citation - the representation of a speech or thought, marked out to indicate that it is a quotation. Among the possible attributes there are:
  - The @type attribute: it can be used to indicate whether the quoted passage is pronounced or simply thought, or to characterize it more finely: possible values are: *spoken* (for the representation of direct speech, usually marked by quotation marks) and *thought* (for the representation of thoughts, for example, internal monologue).
  - The @who attribute: it identifies the speaker in the case of a direct speech passage.
- **<said>** (speech or thought) indicates passages thought or spoken aloud, whether explicitly indicated in the source or not, whether directly or indirectly reported, whether by real people or fictional characters.
- **<quote>** (quotation) contains a phrase or passage attributed by the narrator or author to some agency external to the text.

The <q> (http://www.tei-c.org/release/doc/tei-p5-doc/en/html/ref-q.html) element may be used if no further distinction beyond this is judged necessary. If it is felt necessary to distinguish such passages further, for example to indicate whether they are regarded as

speech, writing, or thought, either the type attribute or one of the more specialized elements discussed in this section may be used. For example, the element <quote> (http://www.tei-c.org/release/doc/tei-p5-doc/en/html/ref-quote.html) may be used for written passages cited from other works, or the element said for words or phrases represented as being spoken or thought by people or characters within the current work. If the distinction among these various reasons why a passage is offset from surrounding text cannot be made reliably, or is not of interest, then any representation of speech, thought, or writing may simply be marked using the q element. Quotation may be indicated in a printed source by changes in type face, by special punctuation marks (single or double or angled quotes, dashes, etc.) and by layout (indented paragraphs, etc.), or it may not be explicitly represented at all.

Encoding person name with TEI

The TEI guidelines present several models for encoding a set of types of named entities. One of these models aims to annotate person names. This can cover the annotation of all the information related with the person name (such as first name, family name, additional name, etc.). To conduct nominal record linkage or even to create an alphabetically sorted list of personal names, it is important to distinguish between a family name, a forename and an honorary title. Similarly, when confronted with a string such as 'أمير المؤمنين أبي حفص عمر ابن الخطّاب' ('*Prince of the Believers Abu Hafs Umar Ibn Al-Khattab*'), the analyst will often wish to distinguish amongst the various constituent elements present, since they provide additional information about the status, the occupation, or the residence of the person to whom the name belongs. The following TEI elements are provided for encoding person name and related purposes.

- **<persName>** (personal name) contains a proper noun or proper-noun phrase referring to a person, possibly including one or more of the person's forenames, surnames, honorifics, added names, etc.
- **<surname>** contains a family name, as opposed to a given, baptismal, or nick name.
- **<forename>** contains a forename, given or baptismal name.
- **<roleName>** contains a name component which indicates that the referent has a particular role or position in society, such as an official title or rank.
- **<addName>** (additional name) contains an additional name component, such as a nickname, epithet, or alias, or any other descriptive phrase used within a personal name.
- **<nameLink>** (name link) contains a connecting phrase or link used within a name but not regarded as part of it, such as *"de"* or *"بن"*.
- **<genName>** (generational name component) contains a name component used to distinguish otherwise similar names on the basis of the relative ages or generations of the persons named.

We were inspired by TEI models for encoding quotation and person information to build a model for Hadith text encoding. The following section illustrates this model who was improved to encode *Matn* and *Isnad* of Hadith text.

4 Proposed Model for Hadith Encoding with TEI

In this section, we present our proposed encoding model shaped for Hadith text and inspired from the TEI guidelines. To realize the standardization of Hadith corpus with TEI, we start by the selection of the necessary data categories that harmonized with Hadith text specification. The structure of the Hadith text characterized with the imbrication of the different part of the text: Hadith include *Isnad* and *Matn*, and each one of them include other parts. Also, TEI allows the imbrication and the restructuring of several models and elements.

We start with the concept of the similarity between the structure of Hadith text is similar to the basic structure of quotations and named entities: considering the *Matn* text of the Hadith as a quotation and the *Isnad* as an enchained list of person names. To achieve our initial TEI model for Hadith encoding, we select the adaptable TEI elements for *Isnad* and *Matn* from TEI model for encoding quotations and named entities [14].

Starting with the *Isnad* encoding, we consider that the structure of TEI model for encoding the person name can be developed to create a TEI model conforming with the structure of an Arabic person name. Moreover, we based on the integration and imbrication of TEI elements to represent the chain of Al-Hadith transmitters. Returning to the TEI coding of complex structures in general, person names are presented at least by a single <persName> element. The <persName> element is the one that includes all the person name information. Thus, we use this element for transmitter name annotation.

The structure of the *Matn* is quite similar to the structure of quotations. In fact, many suggestions were proposed by TEI to encode quotations or narration of the author or transmitted quote mentioned inside the document. We used a TEI model to encode *Matn* text. Figure 1 illustrates the basic elements model that we customized to adapt Al-Hadith text structure.

Figure 1 presents our adapted model respecting the main structure of an ideal Hadith text. The representation starts with the encoding of the header of the Hadith proprieties in <teiHeader> element. Then the main corpus in <body> element includes in <text> element. The body contains a <q> element to encode all the Hadith. This quotation element comprises the *Isnad* encoded in a succession of <persName> elements and the *Matn* in a <said> element. <persName> covers all the transmitter name information. The *Matn* also can include person names which will be encoded in <persName>. The *Matn* may contain some direct quotations encoded in <quote> elements.

The development of a specific TEI model for Al-Hadith texts aims to formulate a TEI structure for the encoding of Al Hadith texts. To this first objective, we select the necessary data categories and the adaptable TEI elements from TEI model for encoding quotations and named entities. Our second objective was the creation of a prototype to generate a normalized Hadith text encoded with TEI model. We present this prototype in the next section.

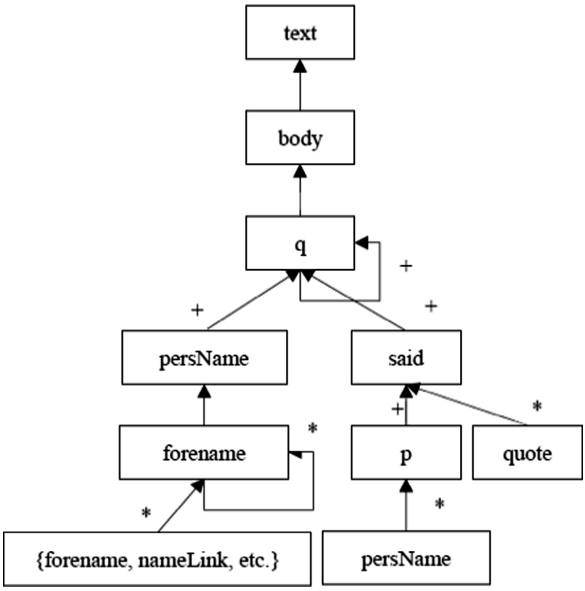


Fig. 1. Extraction from the basic TEI encoding model for Hadith text

5 Elaborated Prototype for the Automatization of Hadith Encoding in TEI

To implement the creation of the encoded Hadith corpus with our TEI model, we proposed a prototype for encoding Hadith text. This prototype is developed to generate automatically the encoded Hadith texts with TEI format. For the implementation, we used some tools and programs. First, we designed our system with UML. Then, we used Oxygen XML Editor to adapt a TEI structure for the encoded of Hadith text. After that, we developed the prototype using JAVA language and the API JDOM Library.

The creation of a normalized Hadith with our prototype can be divided into two steps. First, as an input file, the system requests the user to choose an external Hadith file path with .txt extension which contains a Hadith text. Then, the prototype reads the Hadith text and separate the *Isnad* from the *Matn*.

For the *Isnad*, the system identifies the narrators from the chains of transmitters in the *Isnad* and encodes each narrator name in a <persName> element. To keep the sequence of the transmitters in order, the system assigns for each <persName> element an "xml:id" attribute which contains as value the order of the narrator in the chains of transmitters. Furthermore, according to Hadith text, the narrator name can contain more than one forename. To organize them in order, the system attribute for each forename a "sort" attribute to sort them by number. This step allows the generation of the encoded chains of transmitters.

For the *Matn* of the chosen Hadith, the prototype identifies his structure and encloses it in a <said> element. To identify the prime narrator who is the first transmitter of the Hadith, the <said> element get a "who" attribute which contains the

```
<?xml version="1.0" encoding="UTF-8"?>
<TEI xmlns="http://www.tei-c.org/ns/1.0">
  <teiHeader>…</teiHeader>
  <text>
    <body>
      <div xml:lang="ar">
        <q>
          <!--encoding of Isnad-->
          <p>حَدَّثَنَا</p>
          <persName xml:id="p5">إِسْمَاعِيلُ</persName>
          <p>قَالَ</p>
          <p>حدثني</p>
          <persName xml:id="p4">
          <forename sort="1">مَالِكُ</forename>
          <forename sort="2" type="nasab">
            <nameLink>بْنُ</nameLink>
            <forename>أَنَسٍ</forename>
          </forename>
          </persName>
          <p>عَنْ عَمِّهِ</p>
          <persName xml:id="p3">
            <forename sort="1" type="kunya">
              <nameLink>أَبِي</nameLink>
              <forename>سُهَيْلِ</forename>
            </forename>
            <forename sort="2" type="nasab">
              <nameLink>بْنِ</nameLink>
              <forename>مَالِكٍ</forename>
            </forename>
          </persName>
          <p>عَنْ</p>
          <persName xml:id="p2">أَبِيهِ</persName>
          <p>أَنَّهُ</p>
          <p>سَمِعَ</p>
          <persName xml:id="p1">
            <forename sort="1">طَلْحَةَ</forename>
            <forename sort="2" type="nasab">
              <nameLink>بْنَ</nameLink>
              <forename>عُبَيْدِ الله</forename>
            </forename>
           </persName>
          <p>يَقُولُ</p>
        <!--encoding of Matn-->
        <said who="p1">
          <p>
```

```
جَاءَ رَجُلٌ إِلَى رَسُولِ اللَّهِ صلّى الله عليه وسلّم مِنْ أَهْلِ نَجْدٍ، ثَائِرُ الرَّأْسِ، يُسْمَعُ دَوِيُّ صَوْتِهِ،
وَلاَ يُفْقَهُ مَا يَقُولُ حَتَّى دَنَا، فَإِذَا هُوَ يَسْأَلُ عَنِ الإِسْلاَمِ فَقَالَ رَسُولُ اللَّهِ صلّى الله عليه وسلّم:
</p>
<quote>.«خَمْسُ صَلَوَاتٍ فِي الْيَوْمِ وَاللَّيْلَةِ»</quote>
<p>:فَقَالَ هَلْ عَلَيَّ غَيْرُهَا؟ قَالَ</p>
<quote>.«لاَ، إِلاَّ أَنْ تَطَوَّعَ»</quote>
<p> :قَالَ رَسُولُ اللَّهِ صلّى الله عليه وسلّم</p>
<quote>.«وَصِيَامُ رَمَضَانَ»</quote>
<p>:قَالَ هَلْ عَلَيَّ غَيْرُهُ؟ قَالَ </p>
<quote>.«لاَ، إِلاَّ أَنْ تَطَوَّعَ»</quote>
<p>:قَالَ وَذَكَرَ لَهُ رَسُولُ اللَّهِ صلّى الله عليه وسلّم الزَّكَاةَ. قَالَ هَلْ عَلَيَّ غَيْرُهَا قَالَ</p>
<quote>.«لاَ، إِلاَّ أَنْ تَطَوَّعَ»</quote>
<p> قَالَ فَأَدْبَرَ الرَّجُلُ وَهُوَ يَقُولُ وَاللَّهِ لاَ أَزِيدُ عَلَى هَذَا وَلاَ أَنْقُصُ. قَالَ رَسُولُ اللَّهِ صلّى الله
:عليه وسلّم</p>
<quote>.«أَفْلَحَ إِنْ صَدَقَ»</quote>
</said>
</q>
</div>
</body>
</text>
</TEI>
```

reference of the prime narrator as value. In Hadith text, the *Matn* is characterized by different structures, it could be narrative text with no quotations, or sort of conversations which can contain direct discourses or quotations. It can also contain other person names that need to be identified in the encoding phase. The prototype allows the identification of all these data information. The encoding of each part of the *Matn* goes line by line: the system identifies the structure of each part and encodes it in a corresponding element, for example, a narrative text identified as a narration an encoded in <p> (paragraph) element or a quotation encoded in <quote> element.

After that, the prototype generates the complete TEI encoded file and save it as an output of the system. The following XML code present an output file from our prototype covering the TEI encoding of the Hadith number 46 from the chapter of Belief from Sahih Al-Bukhari book.

Our prototype allows as to generate encoded Hadith text with TEI format. To create encoded Hadith corpus, we made an evaluation phase to test the consistency of our prototype and the adaptability and the flexibility of the TEI model for Hadith text. The evaluation phase is presented in the following section.

6 Evaluation and Discussion

To evaluate our prototype, we collected the first 1000 Arabic Hadith text from 14 chapters from Sahih Bukhari book. Consequently, the prototype generates respectively 1000 TEI files representing each Hadith encoded with TEI format. Table 1 illustrates the obtained results.

Table 1. Summary table for the prototype results.

Evaluated hadith	Hadith chapter	Total encoded	Encoded correctly	Encoded incorrectly
1000 Hadith	**14 chapter of Sahih Bukhari**	**982**	**846**	**136**
007 Hadith	Chapter of revelation of Sahih Bukhari	007	006	01
051 Hadith	Chapter of belief of Sahih Bukhari	051	042	09
076 Hadith	Chapter of knowledge of Sahih Bukhari	075	064	11
113 Hadith	Chapter of ablutions (wudu') of Sahih Bukhari	112	096	16
046 Hadith	Chapter of bathing (ghusl) of Sahih Bukhari	046	039	07
040 Hadith	Chapter of menstrual periods of Sahih Bukhari	038	032	06
015 Hadith	Chapter of ablution with dust of Sahih Bukhari	014	009	05
172 Hadith	Chapter of prayer (salat) of Sahih Bukhari	164	150	14
082 Hadith	Chapter of time of the prayer of Sahih Bukhari	081	066	15
273 Hadith	Chapter of call to prayer of Sahih Bukhari	270	234	36
066 Hadith	Chapter of Friday prayer of Sahih Bukhari	066	056	10
006 Hadith	Chapter of fear prayer of Sahih Bukhari	006	005	01
042 Hadith	Chapter of the two festivals of Sahih Bukhari	041	037	04
011 Hadith	Chapter of the Witr prayer of Sahih Bukhari	011	010	01

Table 1 shows that sometimes for particular Hadith texts, we can obtain erroneous encoding. The total number of obtained encoded Hadith texts is 982. The program succeeded to produce 846 Hadith encoded correctly. However, we found 136 Hadith incorrect or incomplete encoded which require some rectification on our TEI model to

obtains more encoding coverage for some particular part in some types of Hadith text. This problem of incorrect or incomplete encoding is related with some exceptions and particular Hadith forms such as irregularity in *Matn* and *Isnad* position or combination between two or more chain of transmitters referring a same *Matn*. Also, the prototype misses the encoding of some Hadith because of their exceptional forms: some Hadiths refer directly to prefix Hadiths and came without *Isnad* or at least they refer to the *Isnad* of their prefix Hadiths. Indeed, we estimate the quality of our work manually. Table 2 illustrates the obtained values of precision, recall and F-score.

Table 2. Summary table of the precision, recall and F-score.

Hadith corpus	Precision	Recall	F-score
1000 Hadith	0,86	0,85	0,85

According to the value of precision, we conclude that the value of precision is worth 0.86. Also, the recall value is 0.85. These values provide an F-measure equal to 0.85.

Consequently, we conclude that the obtained results are encouraging. Besides, we can say that this prototype is flexible and easy to maintain because it is based on an object-oriented programming language. However, we handled some problems. Some of them are related with the particular Hadith forms which need to integrate more specificity and to develop our TEI model to cover the encoding of the particular and the complex forms of Hadith text.

7 Conclusion and Perspectives

The normalization of Al-Hadith Al-Shareef can take the automatic processing of such corpus to another level. In this work, to attain our main objective, we based on TEI guidelines to elaborate a TEI module customized for Hadith structure. To achieve that, first, we started with a deep study of Hadith text structure. Second, we identified the data categories from the TEI standard register which harmonized with the language specification. After that, we elaborated a prototype for the automatic processing of the encoding of Hadith text with our TEI model. Then, we tested our prototype with a 1000 Hadith text from 14 chapters from Sahih Bukhari book. The elaborated prototype allowed us to generate normalized Hadith texts. As mentioned, the obtained values of measures show that the results obtained from our prototype are encouraging. These results can be used in others levels of analyses.

As perspectives, we want improve our TEI modeling of Al-Hadith Al-Shareef by incorporating other criteria and specifications for deeper encoding of the fundamental fragments of the Hadith. Also, we need to integrate more TEI elements in the Hadith model to reach a deep description for the exceptional part in Hadith structure. Besides that, we want to improve our prototype to generalize our method to cover complex Hadith structures.

References

1. Dalloul, Y.M.: An ontology-based approach to support the process of judging Hadith Isnad, Thesis Submitted in Partial Fulfillment of the Requirements for the Degree of Master in Information Technology, March 2013
2. Azmi, A., Badia, N.: iTree – automating the construction of the narration tree of Hadith. IEEE (2010)
3. Alrfaai S.: عناية العلماء بالاسناد وعلم الجرح والتعديل و أثر ذلك في حفظ السّنة النبوية, AlMadina Almonawara (2004)
4. Alaskalani A.: تقريب التهذيب. Society AlRisala, Bayrout, Labunan (2008)
5. Boella, M., Romani, F.R., Al-Raies, A., Solimando, C., Lancioni, G.: The SALAH project: segmentation and linguistic analysis of ḥadīṯ Arabic texts. In: Salem, M.V.M., Shaalan, K., Oroumchian, F., Shakery, A., Khelalfa, H. (eds.) AIRS 2011. LNCS, vol. 7097, pp. 538–549. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25631-8_49
6. Baraka, R.: Building Hadith ontology to support the authenticity of Isnad. Int. J. Islam. Appl. Comput. Sci. Technol. **2**(1), 25–39 (2014)
7. Alkhatib M.: Classification of Al-Hadith Al-Shareef using data mining algorithm, October 2010
8. Sayoud, H.: Author discrimination between the Holy Quran and Prophet's statements. Lit. Linguist. Comput. **27**(4), 427–444 (2012)
9. Saloot, M.A., et al.: Hadith data mining and classification: a comparative analysis. Artif. Intell. Rev. **46**, 113–128 (2016)
10. Burnard, L., Sperberg-McQeen, C.M.: TEI P5: Guidelines for Electronic Text Encoding and Interchange. Text Encoding Initiative Consortium, Version 3.0.0. revision 89ba24e (2016)
11. Dufournau, N., Demonet, M.-L., Uetani, T.: Manuel d'encodage XML-TEI Renaissance et temps modernes Imprimés-manuscrits, Version Beta, UMR 6576 (2008)
12. Ide, N., Veronis, J.: Une application de la TEI aux industries de la langue: le Corpus Encoding Standard. Cahiers GUTenberg **24**, 166–169 (1996)
13. Burnard, L., Sperberg-McQeen, C.M.: La TEI Simplifiée: Une introduction au codage des textes électroniques en vue de leur échange. Cahiers GUTenberg **24**, 23–151 (1996)
14. Maraoui, H., Haddar, K., Romary, L.: Modeling of Al-Hadith Al-Shareef with TEI. In: ICEMIS Conference (2017)

A New Tool for Benchmarking and Assessing Arabic Syntactic Parsers

Younes Jaafar(✉) and Karim Bouzoubaa

Mohammadia School of Engineers, Mohammed Vth University, Rabat, Morocco
younes.jfr@gmail.com, karim.bouzoubaa@emi.ac.ma

Abstract. This work aims to develop a Natural Language Processing (NLP) tool for benchmarking and assessing Arabic syntactic parsers. This tool is integrated within the Software Architecture For Arabic language pRocessing (SAFAR). Indeed, SAFAR contains several ANLP tools from simple preprocessing up to the semantic level. The benchmarking tool will take advantage of the available basic tools in addition to the flexibility and reusability of SAFAR. The benchmark process takes as input an evaluation corpus and one/several syntactic parsers implementations. As a result, it outputs the most common metrics used for evaluation namely: precision, recall, accuracy and F-measure. We introduced also a new metric called G_p-score which takes into account the execution time besides the accuracy. The execution time is very crucial for some tasks such as real-time automatic translators or in the context of processing huge data. This benchmarking solution will help researchers in comparing their parsers against each other; it will help as well other researchers in selecting the appropriate parser to use within their high level projects. Two Arabic syntactic parsers are evaluated to give a concrete example of this tool: The Stanford parser and the ATKS parser.

Keywords: Arabic NLP · Syntactic parsers · Evaluation · Benchmark

1 Introduction

The world of the internet has known a huge and continuous growth regarding the Arabic content: texts, videos or images, etc. The statistics show that this content has increased by 2501% between the year of 2000 and 2011 [1]. As a result, developing Arabic Natural Language Processing (ANLP) tools is required to process this huge and growing content. In fact, there are many applications that have been already developed for Arabic such as search engines [2] and machine translation systems [3]. Many of these applications rely on syntactic parsers in order to define the structure of the sentences and prepare it for high level processing stages. The syntax level describes how to organize tokens in order to build statements. It is the scientific study of sentence construction and it has been a major area of research within computational linguistics for decades. Indeed, the syntactic analysis represents an essential step in processing natural languages; it serves as a tool for particular high level applications and is preliminary to further analysis, such as semantic processing. The purpose of the syntactic analysis is to find all the syntactic structures of a sentence. Parsers use several formalisms, such as Head-driven Phrase

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 230–243, 2018.
https://doi.org/10.1007/978-3-319-73500-9_17

Structure Grammar (HPSG) in order to parse sentences. The accuracy of an application is affected directly by the accuracy of the parser it uses. The more the parser is accurate, the more the results of the application are accurate as well, and vice versa. Hence, it is very important for researchers to search for the most suitable syntactic parser for their research projects.

However, ANLP researchers can be confused when selecting the appropriate parser according to their accuracies. In fact, authors usually use different custom corpora to present the accuracy of their parsers, which may lead to different results using other corpora. That is to say, one parser may get better results for one corpus but not for another one. Thus, all parsers should be compared using several common corpora, or at least using one common corpus in order to be fair in their evaluation. Hence, to help researches making the choice of the parser to use, comparative tools should be developed in order to present advantages and drawbacks of each parser according to the same corpus. To our knowledge, no such comparative tools have been addressed so far for the context of Arabic. One should note that we have previously developed similar benchmarking tools for Arabic stemmers [4] as well as for Arabic morphological analyzers [5, 6].

Therefore, our objective in this article is to present a new automatic tool for benchmarking and evaluating Arabic syntactic parsers. This tool is generic, reusable and compares the results of parsers according to an evaluation corpus. This benchmark is designed to check the accuracy, precision, recall, F-measure of the results and also compare the parsers with each other according to their execution time. Indeed, when evaluating and benchmarking NLP tools, researchers usually use metrics related to the accuracy of results only. However, it is very difficult to use syntactic parsers that do not optimize their execution time to deal with the huge content of the Arabic language on the internet. It is then necessary to take into consideration the execution time when comparing and evaluating two parsers.

To provide concrete examples about the effectiveness of our solution of the benchmark, we have selected two syntactic parsers that are freely available and which are widely used within the ANLP community, namely: Stanford [7] and ATKS [8] parsers. These two Arabic parsers are the only free ones and ready to use that we have found; they will serve as examples to present the benchmarking process. It should be mentioned that our benchmarking tool is not limited to these two parsers, it can be easily extended to compare new parsers and/or to use new evaluation corpora.

The rest of this paper is organized as follows. The next section presents related works concerning the comparison and the benchmark of syntactic parsers. In Sect. 3 we outline some challenges that should be overcome when benchmarking parsers. Section 4 provides the steps of developing our benchmark solution and the metrics used. The results of experiments are described and commented in Sect. 5. Finally, some conclusions are given in Sect. 6.

2 Related Works

In this section, we present algorithms and approaches that have been developed for comparing syntactic trees and parsers. In fact, comparing parsers is based on comparing

their output syntactic trees. One should note that we have found no works addressing the evaluation of Arabic syntactic parsers; all the following works concern mainly the English language.

Pawlik and Augsten presented AP-TED [9], a new memory efficient algorithm for calculating the similarity between trees and providing the TED (Tree Edit Distance) metric. AP-TED calculates the minimal-cost sequence of node edit operations that transforms one tree into another. Authors claim that their solution runs at least as fast as RTED [10] without trading in memory efficiency. Pawlik and Augsten provide the tree edit distance implementation as a runnable Java JAR file that can be downloaded from their official website[1].

Atwell [11] presented a comparative evaluation of grammatical annotation models. In his article, Atwell focuses mainly on the differences in parsing schemes, and discusses how these differences should be taken into account in comparative evaluation of parsers. He presented two solutions for this: the first one is to convert parsers outputs to a dependency structure, and the second one consists of mapping parses onto simple context-free constituency structure trees. Both solutions present problems and challenges that should be taken into consideration. Atwell does not provide any tool for comparing and evaluating parsers.

Tsarfaty et al. developed TedEval [12], which is a heuristics-free framework for cross-experiment parse evaluation. It is used to evaluate parsing results for different annotations schemas. TedEval calculates the parse error relative to the common gold standard. According to its developers, TedEval can be used to evaluate both functional trees and dependency trees by providing the TED metric. TedEval can be downloaded via Tsarfaty's website[2].

Black et al. [13] presented the comparison of 10 parsers using constituent boundaries. The conducted evaluation was done against a reference build from a majority vote out of the output parses consisting of 14 sentences. Two metrics were provided for each parser: (1) the number of crossing parenthesis and (2) the recall. The average scores obtained were respectively of 4% crossing-brackets and 94% recall. In the literature, this type of evaluations is usually referred to as "Parseval" [14].

Lin [15] presented a dependency-based method for evaluating broad-coverage parsers. The author claims that his method offers several advantages over previous methods that are based on phrase boundaries. He introduced the error count score which is relevant to semantic interpretation. Lin also presented an algorithm to transform constituency trees into dependency trees in order to be able to evaluate them using his dependency-based method.

Kummerfeld et al. [16] proposed a new approach for detecting error types in parsers outputs. Authors remarked that the F-score for constituency parsing evaluation gives a useful measure of overall performance; however it does not provide any information about the nature, or relative importance, of the remaining errors. To remedy this, they

[1] http://tree-edit-distance.dbresearch.uni-salzburg.at/.
[2] http://www.tsarfaty.com/unipar/download.html.

proposed a new method of error classification using tree transformations. Their tool can be downloaded via GitHub website[3].

There are many other works that have been conducted for evaluating English parsers and syntactic trees that is not possible to cite them all. Many other works have also been accomplished for other languages such as French [17]. However, for the context of Arabic, there are few works concerning parsers themselves; which justifies the lack of researches for evaluating and benchmarking Arabic syntactic parsers. This situation makes the benchmark of Arabic parsers a challenging task and poses several problems as presented in the next section.

3 Syntactic Parsers Benchmark: Challenges and Solutions

Evaluating Arabic syntactic parsers using an evaluation corpus highlights several challenges that can be categorized into two types: (1) The lack of Arabic parsers and evaluation corpora, and (2) the lack of standards in parsers outputs and schemas.

3.1 Limited Number of Arabic Parsers

Researches on Arabic syntactic parsers have not reached an advanced stage compared to other languages such as English in which many parsers have already been developed [18–21]. Indeed, most of the existing Arabic tools focus mainly on the morphology rather than the syntax. Hence, we have a very limited number of Arabic parsers that we can rely on in order to present our benchmarking solution with several examples. We have found only two Arabic syntactic parsers that are freely available and widely used within the ANLP community to serve as example for our tests and experiments with the benchmark solution, these parsers are:

Stanford parser [7]: is one of the most used syntactic parsers of the Arabic language within the ANLP community, it is written in Java and can be run either through a graphical interface or using a command line.

ATKS parser [8]: is an Arabic syntactic parser developed by Microsoft in the Advanced Technology Lab in Cairo. This parser is included within the Arabic Toolkit Service (ATKS). It should be mentioned also that the ATKS parser is integrated into several Microsoft products and services such as Windows, Office and Bing, and it can be exploited only as a web service.

3.2 Limited Number of Syntactic Evaluation Corpora

In order to perform the syntactic benchmark, the results returned by a syntactic parser must be compared to results of an annotated corpus. This corpus should contain a maximum amount of different sentences with their possible syntactic trees. There are several corpora for Arabic such as the Penn Arabic Treebank (PATB) [22], the Prague Arabic Dependency Treebank (PADT) [23], the Columbia Arabic Treebank (CATiB)

[3] https://github.com/jkkummerfeld/berkeley-parser-analyser.

[24] and the Quranic Arabic Dependency Treebank (QADT) [25]. All these corpora contain both morphological annotation of individual words, and syntactic parses of sentences in either constituency phrase structure grammar or dependency grammar. However, they are not available for large public. They are either not available for download or not free. This is because building these kinds of corpora is time consuming and requires experts with linguistic knowledge in Arabic language. It has been realized nowadays that the effort needed to build such corpora may exceed largely the effort needed to build tools that exploit them. This justifies the lack of such free gold standards for benchmarking Arabic syntactic parsers.

To remedy this, we used OntoNotes [26] as our syntactic evaluation gold standard. OntoNotes is a free annotated corpus whose development was supported under the GALE program of the Defense Advanced Research Projects Agency. OntoNotes includes approximately 1.5 million words of English, 800 K of Chinese, and 300 K of Arabic. The Arabic portion of OntoNotes 5.0 includes 300 K of Arabic An-Nahar newswire, with Treebank, word sense, proposition, coreference, and named entity annotation layers. The newswire data is taken from the 400 K Arabic Treebank Part 3. V3.1 (ATB P3 V3.1). OntoNotes is the only freely available evaluation corpus for the syntax that we have found for Arabic.

3.3 No Standards for Tagsets

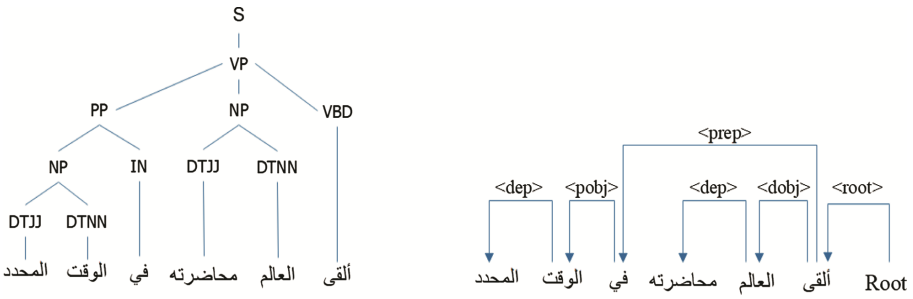
The POS tags used in OntoNotes corpus and in Stanford and ATKS parsers are heterogeneous and not identical. For example, OntoNotes uses the tag "ADV" for Adverb, while Stanford and ATKS parsers use the "RB" tag. Therefore, prior developing the syntactic benchmarking tool, it was necessary to map all tagsets used to a unified and simpler tagset consisting only of the major POS categories listed in Table 1. This mapping was based on the scheme suggested in [27]. All tags are converted automatically to this simplified tagset before executing the benchmarking process.

Table 1. Reduced tagset used fort the benchmark

Tag name	Gloss
CC	Coordinating conjunction
DT	Determiner
IN	Preposition
JJ	Adjective
NN	Noun
NNP	Proper noun
NNS	Plural noun
PRP	Pronoun
RB	Adverb
RP	Particle
VBD	Perfect verb
VBN	Passive verb
VBP	Imperfect verb

3.4 Constituency vs Dependency Structures

There are two major varieties of syntactic annotation: a phrase structure (also known as constituency) and a dependency representation. Parsers may output either one of these structures or both. However, most of the parsers do not output the dependency structures [11]. There are certainly other grammar models such as HPSG but less used compared to dependency and constituency structure, especially when it comes to annotating corpora. The dependency Grammar is a class of modern syntactic theories that are all based on the dependency relation. Dependency is the notion that linguistic units, e.g. words, are connected to each other by directed links. In a dependency tree, each word of the sentence is a modifier of exactly one other word. On the other side, the constituency grammar describes the structural categories and hierarchical structure of the whole sentence rather than dealing with word-word relations. Figure 1 gives an example of dependency and constituency structures for the sentence: "محاضرته في الوقت المحدد ألقى العالم".



a. Constituency structure

b. Dependency structure

Fig. 1. Examples of dependency and constituency structures

The heterogeneity of these two representations makes their comparison very difficult without transforming one structure to the other. For our case, Stanford parser outputs both types while ATKS parser outputs only the constituency representation. It goes as well for OntoNotes which is annotated using the constituency representation only. Therefore, we cannot evaluate these parsers according to the dependency structure. To remedy this, we have limited the benchmark in this article to constituency representation only until new Arabic parsers produce dependency structures that we can rely on.

4 Developing the Benchmark Solution

The main objective of this benchmark is to develop an automatic parser evaluation tool, which assesses the Arabic output syntactic trees according to an evaluation corpus (OntoNotes) [26] and then calculates metrics related to the accuracy and the execution time. This section describes all steps for creating and developing this benchmarking tool.

4.1 Using SAFAR Framework

We had two possibilities when developing our tool, either develop it as a separate program or integrate it within a platform for ANLP. Since we have already developed similar benchmarking solutions for Arabic stemmers [4] and morphological analyzers [4, 5], we decided to follow the same approach and integrate our parser benchmark solution into SAFAR (Software Architecture For Arabic language pRocessing) [28–30]. SAFAR is a framework dedicated to ANLP written in Java. It is cross platform, modular, extensible and flexible. Therefore, our benchmarking solution can be easily executed within SAFAR with few lines of code, and will benefit from its advantages of extensibility and reusability. Moreover, a researcher will be able to integrate new parsers within SAFAR in order to be compared with the existing ones via the same process without any modification in the benchmarking tool. This integration can be done via code or using web services. New evaluation corpora can be also used instead of OntoNotes. Concerning the execution time of parsers, SAFAR can easily calculate the consumed time of each parser according to its executed code.

As specified in Fig. 2, SAFAR has several layers. The "Utilities" layer includes a set of technical services, the "Resources" layer provides services for consulting language resources such as lexicon, the "Basic layer" contains the three regular layers (morphology, syntax and semantics), the "Application" layer contains high-level applications that use the layers listed above such as "Resources" and "Utilities". Finally, the "Client" tier that provides users (especially, non programmer users such as linguists) the opportunity to exploit available resources and applications from all other layers. Our benchmark solution is integrated within the "Utilities" layer. Stanford and ATKS parsers are integrated within the "Syntax" layer. As for OntoNotes, it is integrated within the "Resources" layer.

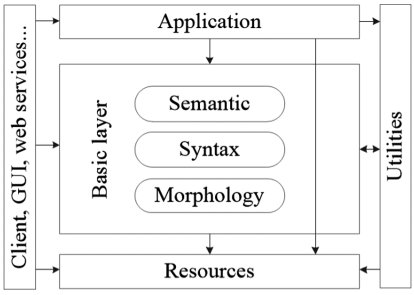


Fig. 2. SAFAR framework architecture

4.2 Preprocessing the Evaluation Corpus

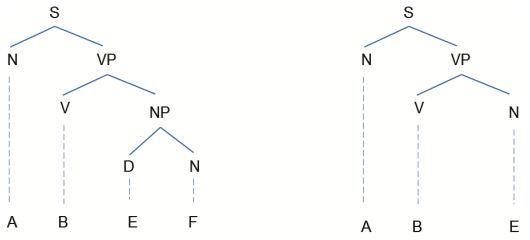
The OntoNotes trees and the trees generated using Stanford and ATKS parsers follow basically the same schema rules concerning the used tags. However, there are still some differences that should be normalized before performing the evaluation process. This normalization consists of removing any additional labels that are used in OntoNotes but not in any of the other parsers, such as words that are labeled with –NONE– tag. In

addition, all detailed tags in OntoNotes are reduced to a common basic annotation. For example, the tag "NOUN+NSUFF_FEM_SG+CASE_DEF_GEN" is reduced to "NOUN" since the remaining part contains detailed information that is not available in parsers outputs, therefore we cannot evaluate parsers based on that detailed part.

Moreover, we have noticed that small sentences are always parsed correctly while long ones are not. Therefore, in order to evaluate parsers according to the length of sentences and show how long sentences can affect the performance and results, we have divided the OntoNotes corpus into three lists of sentences. List 1 contains all small sentences with 10 words or less. List 2 contains all sentences with a number of words between 10 and 20. And finally, List 3 contains all sentences with 30 words or more. Benchmarking results is provided for each of these sentences lists.

4.3 Calculating the Evaluation Metrics

The Arabic syntactic parsers benchmark process consists of returning a list of metrics on which researchers can rely to measure the performance of a given syntactic parser. To measure this performance, we used the usual evaluation metrics[4]: the precision, recall, accuracy and F-measure. Other metrics can also be applied such as Tree Edit Distance (TED) and similarity; however we have initially considered the use of classical metrics only since they give a global overview of the relevance of returned results. Calculating the evaluation metrics is performed as follows. First, the evaluation process finds all constituents in both OntoNotes and the parser output. For every constituent, we search for its label and span. Figure 3 gives an example of how we get labels and spans.



a. Gold standard tree b. Parser tree

Fig. 3. Example of two different syntactic trees from the Gold and a parser

[4] http://en.wikipedia.org/wiki/Precision_and_recall.

The constituents for each tree in Fig. 3 are given in Table 2 in the form label:span.

Table 2. Constituents of the Gold and the parser trees according to Fig. 3

Corpus constituents	Parser constituents
N:A	N:A
V:B	V:B
D:E	N:E
N:F	VP:BE
NP:EF	S:ABE
VP:BEF	
S:ABEF	

Once we get the constituents in the form label:span, the evaluation metrics are then calculated for each syntactic tree using the parameters presented in Table 3:

Table 3. Parameters used to calculate the benchmark metrics

	Positive (P)	Negative (N)
True (T)	Total of correct constituents returned by the parser	Total of incorrect constituents identified by the parser
False (F)	Total of incorrect constituents returned by the parser	Total of correct constituents not returned by the parser

Using the parameters TP_s, FP_s, TN_s and FN_s, where "s" refers to "sentence", we can calculate our main metrics as follows:

$$Precision = \frac{\sum TP_s}{\sum TP_s + \sum FP_s}$$

The precision of results returned by a syntactic parser expresses the total number of correct constituents compared to the total number of all constituents returned by the parser. The precision can be less than 100% even if the parser returns all possible correct constituents of the sentence; this means that it returns some incorrect constituents in addition to the correct one. Precision can also be equal to 100% even if the parser does not return all possible correct constituents for that sentence; this means that all returned constituents are correct.

$$Recall = \frac{\sum TP_s}{\sum TP_s + \sum FN_s}$$

The recall of results expresses the total number of correct constituents returned by a parser for this sentence compared to the total number of all constituents that should be returned. In contrast to the precision, the recall may be equal to 100% even if the parser returns additional incorrect constituents since the recall does not take into account the FP_s parameter.

$$Accuracy = \frac{\sum TP_s}{\sum TP_s + \sum TN_s + \sum FP_s + \sum FN_s}$$

The accuracy of results returned by a parser expresses the proportion of the constituents that are false. In contrast to the precision and recall, the accuracy is equal to 100% only if the parser returns all possible correct constituents, and in addition to that, there are no additional constituents that are incorrect within its results. If the accuracy is equal to 100%, this means that the parser results are perfect.

$$F\text{-}measure = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

The F-measure combines both precision and recall in one single metric. It can be interpreted as a weighted average of the precision and recall.

When evaluating a syntactic parser, researchers often use these usual metrics that are related to the accuracy of results. However, Arabic data in the digital world has become so large that it becomes impossible to neglect the execution time of parsers. That is why we propose a new metric called G_p-score (for Global parser score) that takes into consideration the accuracy of results as well as the execution time, it is calculated as follows:

$$G_P - score = \frac{\sum T_s}{Accuracy}$$

Where T_s is the time taken by the parser to parse the sentence "s". For calculating the G_p-score, we have used the accuracy instead of precision or recall because these latter do not take into consideration all the constituents returned (or that must be returned) by the parser. In the other side, the accuracy takes into account correct constituents of the parser, its incorrect constituents and correct constituents that are not returned by the parser. It should be noted that our G_p-score metric is considered better when its value tends to zero and worse when it tends to a big number. Researchers can rely on this new metric to measure the performance of a parser when the execution time is a crucial parameter for their projects.

4.4 Summary of All Steps

All steps of the execution of our benchmarking tool are summarized in Fig. 4:

In step 0, the program converts all textual trees in the evaluation corpus into memory trees objects according to SAFAR API[5]. Indeed, using memory objects leads to fast evaluation. In step 1, each parser processes all sentences of the evaluation corpus. The results of each parser are then retrieved as memory trees in step 2. In step 3, the program compares the resulting trees of each parser with those of the evaluation corpus and calculates the evaluation metrics.

[5] http://arabic.emi.ac.ma/safar/javadoc.

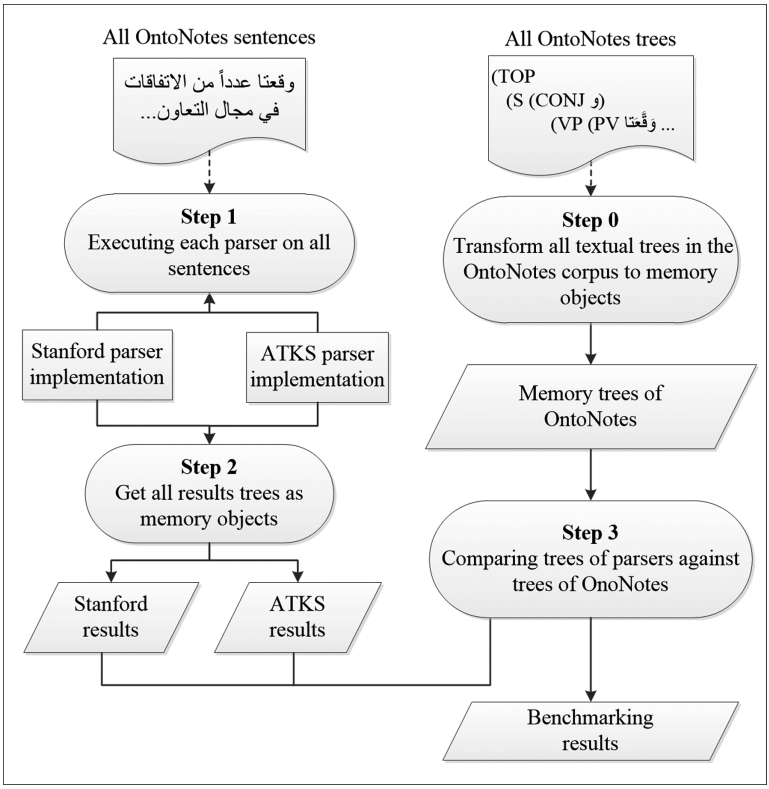


Fig. 4. Steps for executing the parsers benchmark solution

5 Experiments and Results

Tables 4 and 5 show the results of our experiments. We have calculated all metrics for each list of sentences based on their lengths (see Sect. 4.2). Table 4 concerns Stanford parser while Table 5 concerns ATKS parser results. These experiments were performed on a computer having the following characteristics: CPU = Core i7 @2.4 GHz, RAM = 8 GO, Operating System = Win 10, 64 bits.

Table 4. Results of evaluating Stanford parser according to the three categories of sentences

Metrics	List 1	List 2	List 3
Precision	71.48	49.54	32.42
Accuracy	58.48	46.28	31.15
Recall	76.28	87.54	88.85
F-measure	73.8	63.27	47.51
Execution time	2.13	12.50	49.68
G_p-score	0.036	0.270	1.594

Table 5. Results of evaluating ATKS parser according to the three categories of sentences

Metrics	List 1	List 2	List 3
Precision	68.93	48.35	32.42
Accuracy	62.35	47.33	32.1
Recall	86.72	95.75	97.07
F-measure	76.81	64.25	48.61
Execution time	18.54	42.72	88.74
G_p-score	0.297	0.902	2.76

As it is shown in Tables 4 and 5, the precision, accuracy, recall and F-measure vary from roughly 31% to 97% and are in all cases rather close for both parsers. However, the experiments show clearly that Stanford parser processes the sentences in less time than the ATKS parser, which may be due to the remote calls of web services of ATKS. Indeed, ATKS needs to call its server each time the syntactic parsing is executed. Gp-score varies from 0.04 to 2.8. Stanford parser gets highest scores of G_p-score due to its execution time.

Results show also that both parsers get the highest scores when parsing small sentences (List 1) and get the worst scores with longer sentences (List 3). The more sentences are long, the more parsers performances are affected on both accuracy and execution time. This is because the sentences become more complex and therefore lead to more errors in the parsing process.

6 Conclusion

In this paper, we presented a new automatic tool for benchmarking Arabic syntactic parsers. We outlined some challenges to take into consideration when developing a system for comparing Arabic parsers. Then we presented our reusable solution to solve this problem. This solution is integrated within the Software Architecture For Arabic language pRocessing (SAFAR) framework. Indeed, SAFAR represents for us a way to standardize the various aspects shared by Arabic processing tools in order to promote interoperability and flexibility. We selected two freely available Arabic syntactic parsers (Stanford and ATKS) in order to compare their results and give an example of use of this solution. We used OntoNotes as evaluation corpus to perform the benchmark of parsers. This corpus includes 300 K of Arabic An-Nahar newswire, with Treebank, word sense, proposition, coreference, and named entity annotation layers. The newswire data is taken from Treebank Part 3. V3.1. We have divided this corpus into three sub lists of sentences according to their lengths in order to study the impact of long sentences on the parsing process.

Our benchmark process consists of returning a list of metrics on which researchers can rely to measure the performance of a given syntactic parser. We used the usual evaluation metrics namely: the precision, recall, accuracy and F-measure. In addition, we introduced a new evaluation metric called G_p-score (for Global parser score) that combines the accuracy of parsers as well as their execution time. Indeed, the execution time is an important element for many researchers; it may affect the decision of using a

tool in their projects or not. Experiments show that results of accuracy are close for both parsers. However, Stanford parser is faster than ATKS parser in parsing sentences. Experiments show also that the accuracy of parsers is affected by the lengths of sentences. Shortest sentences get highest accuracies and longest sentences get lowest accuracies.

As future work, we plan to create a specific corpus containing sentences by categories: verbal, nominal, etc. This classification will be more suitable to evaluate the parser performance according to the syntactic phenomena, not only to the sentence length. We also plan to standardize all the used tags either by parsers or by evaluation corpora according to ISO 12620[6] guidelines.

References

1. Miniwatts Marketing Group (2001). http://www.internetworldstats.com
2. Hattab, M., Haddad, B., Yaseen, M., Duraidi, A., Shmais, A.A.: Addaall Arabic search engine: improving search based on combination of morphological analysis and generation considering semantic patterns. In: The 2nd International Conference on Arabic Language Resources & Tools (2009)
3. Ittycheriah, A., Roukos, S.: A maximum entropy word aligner for Arabic-English machine translation. In: Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing (2005)
4. Jaafar, Y., Namly, D., Bouzoubaa, K., Yousfi, A.: Enhancing Arabic stemming process using resources and benchmarking tools. J. King Saud Univ. Comput. Inf. Sci. **29**(2), 164–170 (2016)
5. Jaafar, Y., Bouzoubaa, K.: Benchmark of Arabic morphological analyzers: challenges and solutions. In: 9th International Conference on Intelligent Systems: Theories and Applications (SITA 2014), Rabat, Morocco (2014)
6. Jaafar, Y., Bouzoubaa, K., Yousfi, A., Tajmout, R., Khamar, H.: Improving Arabic morphological analyzers benchmark. Int. J. Speech Technol. **19**(2), 259–267 (2016)
7. Green, S., Manning, C.D.: Better Arabic parsing: baselines, evaluations, and analysis. In: The 23rd International Conference on Computational Linguistics (COLING 2010), Beijing (2010)
8. Microsoft: Arabic Toolkit Service (ATKS). https://www.microsoft.com/en-us/research/project/arabic-toolkit-service-atks/. Accessed 01 Mar 2017
9. Pawlik, M., Augsten, N.: Tree edit distance. Inf. Syst. **56**(C), 157–173 (2016)
10. Pawlik, M., Augsten, N.: RTED: a robust algorithm for the tree edit distance. In: Proceedings of the VLDB Endowment (2011)
11. Atwell, E.: Comparative evaluation of grammatical annotation models. Ind. Parsing Software Manuals **17**, 25–46 (1996)
12. Tsarfaty, R., Nivre, J., Andersson, E.: Cross-framework evaluation for statistical parsing. In: Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics (2012)
13. Black, E., Abney, S., Flickenger, D., Gdaniec, C., Grishman, R., Harrison, P., Hindle, D., Ingria, R., Jelinek, F., Klavans, J., Liberman, M., Marcus, M., Roukos, S., Santorini, B., Strzalkowski, T.: A procedure for quantitatively comparing the syntactic coverage of English grammars. In: Proceedings of the Fourth DARPA Speech and Natural Language Workshop, Pacific Grove (1991)

[6] https://www.iso.org/standard/37243.html.

14. Harrison, P., Abney, S., Black, E., Flickinger, D., Gdaniec, C., Grishman, R., Hindle, D., Ingria, R., Marcus, M., Santorini, B., Strzalkowski, T.: Evaluating syntax performance of parser/grammars. In: Proceedings of the Natural Language Processing Systems Evaluation Workshop, Berekely (1991)
15. Lin, D.: A dependency-based method for evaluating broad-coverage parsers. Nat. Lang. Eng. **4**(02), 97–114 (1998)
16. Kummerfeld, J.K., Hall, D., Curran, J.R., Klein, D.: Parser showdown at the wall street corral: an empirical investigation of error types in parser output. In: Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, Jeju Island (2012)
17. Seddah, D., Candito, M., Crabbé, B.: Cross parser evaluation and tagset variation: a French Treebank study. In: Proceedings of the 11th International Conference on Parsing Technologies (2009)
18. Hall, D., Berg-Kirkpatrick, T., Klein, D.: Sparser, Better, Faster GPU Parsing. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore (2014)
19. Bikel, D.: Design of a multi-lingual, parallel-processing statistical parsing engine. In: Proceedings of the Second International Conference on Human Language Technology Research (2002)
20. Charniak, E.: A maximum-entropy-inspired parser. In: Proceedings of the 1st North American Chapter of the Association for Computational Linguistics Conference (2000)
21. Collins, M.J.: A new statistical parser based on bigram lexical dependencies. In: Proceedings of the 34th Annual Meeting on Association for Computational Linguistics (1996)
22. Maamouri, M., Bies, A., Buckwalter, T., Mekki, W.: The Penn Arabic Treebank: building a large-scale annotated Arabic corpus. In: NEMLAR Conference on Arabic Language Resources and Tools, vol. 27, pp. 466–467 (2004)
23. Smrž, O., Bielický, V., Jakub, I.K.: Prague Arabic dependency treebank: a word on the million words. In: Proceedings of the Workshop on Arabic and Local Languages (LREC 2008), Marrakech (2008)
24. Habash, N., Roth, R.M.: CATiB: The Columbia Arabic Treebank. In: Proceedings of the ACL-IJCNLP 2009 Conference Short Papers (2009)
25. Dukes, K., Buckwalter, T.: A dependency treebank of the Quran using traditional Arabic grammar. In: 2010 The 7th International Conference on Informatics and Systems (INFOS), Cairo (2010)
26. Pradhan, S.S., Hovy, E., Marcus, M., Palmer, M., Ramshaw, L., Weischedel, R.: Ontonotes: a unified relational semantic representation. Int. J. Semant. Comput. **1**(04), 405–419 (2007)
27. Bies, A.: catalog.ldc.upenn.edu, 28.1.2003. https://catalog.ldc.upenn.edu/docs/LDC2003T06/arabic-POStags-collapse-to-PennPOStags.txt. Accessed 6 June 2017
28. Jaafar, Y., Bouzoubaa, K.: SAFAR: Software Architecture For Arabic language pRocessing. http://arabic.emi.ac.ma/safar/. Accessed 6 June 2017
29. Souteh, Y., Bouzoubaa, K.: SAFAR platform and its morphological layer. In: Eleventh Conference on Language Engineering, ESOLEC 2011, Cairo (2011)
30. Jaafar, Y., Bouzoubaa, K.: Arabic natural language processing from software engineering to complex pipeline. In: First International Conference on Arabic Computational Linguistics (ACLing), Egypt, Cairo (2015)

Syntactic Parsing of Simple Arabic Nominal Sentence Using the NooJ Linguistic Platform

Said Bourahma, Samir Mbarki(✉), Mohammed Mourchid, and Abdelaziz Mouloudi

MISC Laboratory, Faculty of Science, Ibn Tofail University, Kenitra, Morocco
saidbrh@yahoo.fr, mbarkisamir@hotmail.com, mourchidm@hotmail.com, mouloudi_aziz@hotmail.com

Abstract. Natural Language Processing (NLP) applications such as machine translation, question answering, knowledge extraction, and information retrieval require parsing process as an essential step. In this paper, we present a parser to analyze simple Arabic nominal sentences using the NooJ platform. Hence, we propose a well-classified NooJ dictionary that includes most syntactic, and semantic features. We also present the rule describing the Arabic sentence. Then, we implement the parser that recognizes, and annotates all possible grammatical structures of simple Arabic nominal sentence. We implement a set of transducers modeling Arabic lexical, and syntactic constraints, these constraints reduce parsing ambiguity. Our parser is tested on many sentences extracted from real texts. These experimental results show the effectiveness of the proposed parser for analyzing simple Arabic nominal sentences.

Keywords: Natural Language Processing · Arabic language parser
Syntactic analysis · NooJ linguistic platform

1 Introduction

Natural Language is the language spoken by humans. Any language is based on a vocabulary which consists of a set of words. This group of words must match a set of grammatical rules. A sequence of words, from the vocabulary, form a text, and the set of all possible texts defines the language. NLP is a subfield of Artificial Intelligence and linguistic, devoted to make computers understand statements written in natural language [7, 13]. In fact, NLP employs computational techniques for the purpose of learning, understanding, and producing natural language content. Actually the natural language processing requires relevant information about the language at different levels. Therefore, we may be able to use four knowledge levels about the language: morpho-lexical, syntactic, semantic, and pragmatic. These levels overlay each other. Each level only focuses on a given issue related to that level.

In the NooJ linguistic platform, syntactic grammars are very useful to describe the words sequence which has a meaning [14]. Hence we can use them in order to focus on various kinds of simple nominal sentences. NooJ guarantees high integration of all levels of description thanks to compatible notations and a unified representation for all linguistic analysis results, enabling different analyzers at different linguistic levels to

A. Lachkar et al. (Eds.): ICALP 2017, CCIS 782, pp. 244–257, 2018.
https://doi.org/10.1007/978-3-319-73500-9_18

communicate with one another [13, 16]. The aim of our work is to develop a syntactic parser of simple Arabic nominal sentences. This parser is based on a set of structural grammars. These grammars are implemented in the NooJ platform. In general, Arabic texts are not diacritized. So these texts become ambiguous. That is why the disambiguation of the sentence components is also expected in this work.

The rest of this paper is organized as follows: Sect. 2 is dedicated to related work, Sect. 3 describes our contribution; we have three subsections in this part: the lexicon classification, the disambiguation, and mapping between the lexicon classes and the nominal sentence components. In Sect. 4, we present the main NooJ platform functionalities. Section 5 explains the implementation of our Simple Arabic nominal sentence syntactic parser. Section 6 is devoted to the running and the test of our parser on different sentences. Finally, the last part will present the conclusion and the future work.

2 Related Work

In literature, many approaches were applied to design and implement a syntactic analyzer for parsing Arabic sentences [1, 6, 9, 10]. Actually there are three main approaches: linguistic, statistical, and hybrid. The linguistic methods are based on lexicon and grammars. This approach lacks of resources, for instance, the Arabic grammars do not cover all sentences' types. It is often said that linguistic methods are costly to implement because they require the construction of dictionaries and grammars. However, statistical methods also require a great deal of work to manually construct their reference corpora. The hybrid approach incorporates linguistic rules and corpora-based statistics. So the strengths of both linguistic and statistical approaches to NLP can be combined in a single framework. The other shortcoming of statistical methods is that it relies on reference corpora. So, if the reference corpora contain so many errors, we cannot expect reliable results. Regarding the Arabic language, most of syntactic analyzer developed are based on statistical approach.

3 Methodology and Contribution

This section is devoted to our contribution. The aim of our work is to develop a syntactic parser for simple Arabic nominal sentences. As described in Fig. 1, our methodology is completely based on a linguistic approach. Therefore, we apply three main steps: lexicon classification, disambiguation, and grammar modeling regarding the simple Arabic nominal sentence structure. We have already defined a dictionary [4]. But these entries are not accurately classified. That is why we carry on a new lexicon classification. This classification is very helpful in the last step. In a previous work [8], we also implemented morpho-syntactic rules for processing agglutination using the NooJ platform. The morphological analysis result leads to multiple annotations for the same word. Hence, the disambiguation is required. Finally, from the simple Arabic nominal sentence structure, we map the sentence classes with the nominal sentence components. All these steps are described below:

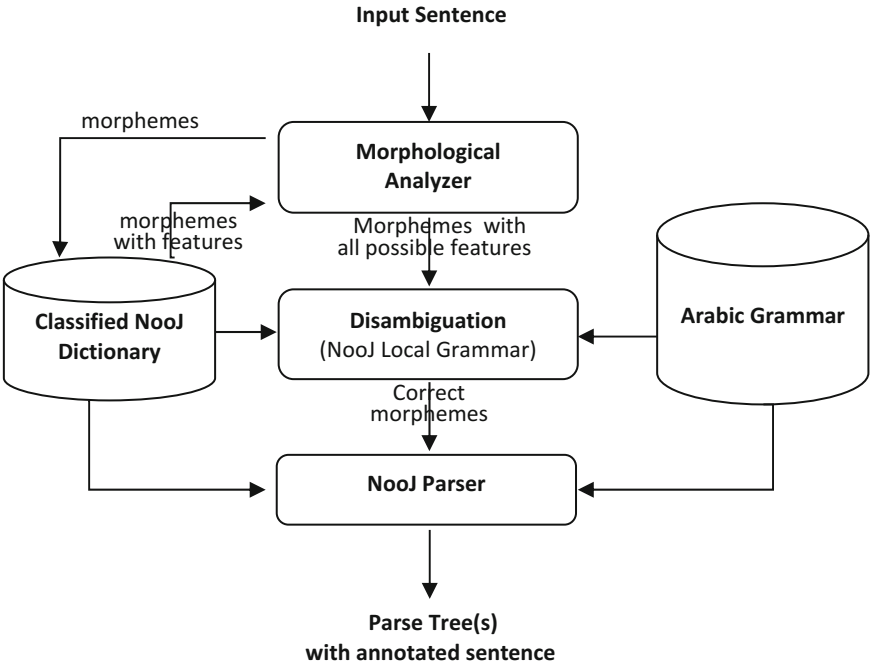


Fig. 1. Analysis steps of Arabic sentence.

3.1 Lexicon Classification

The Arabic language words are divided into three main classes: nouns, verbs, and particles [2, 3, 11, 12, 17, 20–22]. We can eventually add an extra class named "others" or "residuals". This class includes "borrowed words", "numbers", etc. Each class is in turn divided into sub-classes. For example, "complete noun" (الإسم التام, al-'ism al-ttām) and "incomplete noun" (الإسم الناقص, al-'ism al-nnāqiṣ) are two sub-classes for the noun.

A noun in Arabic language is defined as a word which has a meaning without being connected with the notion of time. This class includes pronouns. It also contains "adjectives", "verbal nouns", "noun of place", "proper nouns", etc. as sub-classes. The sub-class "adjectives" has also sub-classes: "resembling adjective" (الصفة المشبهة, al-ṣṣifah al-mušabbahah), "active participles" (اسم الفاعل,'ism al-fā'il), "passive participles" (اسم المفعول,'ism al-maf'ūl), etc.

A verb in Arabic language is a word with two features: action and time. When it is used in a specific context, the verb, prefixed by some particles such as futurity and interrogation particles, get more information about tense, form and meaning. In fact, the "verbs" class is divided into two main sub-classes: "complete verbs" (الفعل التام, al-fi'l al-ttām) and "incomplete verbs" (الفعل الناقص, al-fi'l al-nnāqiṣ). These features are

related to the flexion of the verb. We can also classify verbs regarding many features. One of them could be syntactic feature which involves deeming important property "verb transitivity". Hence a verb could be either "transitive" or "intransitive". The transitive verbs in Arabic handle from one to three accusative forms. Besides the syntactic classification, we can classify verbs regarding semantic features.

Unlike nouns and verbs, particles do not have a meaning regardless of nouns, verbs or particles. The particle can assume the role of a linker between sentences, a linker between words (verbs and nouns), a prefix or suffix, or a sentence modify. They can modify the sentence tense or the sentence meaning. Arabic grammarians divide the "particle" class into three sub-classes: those which are related to the verb (سوف, sawfa, will), "prepositions" are related to the noun, and those which are related to both of them such as "conjunctions" [18, 19]. We can also classify these sub-classes. The sub-sub-classes highlight the grammatical function of the particle.

3.2 Disambiguation

As result of the morphological analysis, a word can have many annotations. For example, in the sentence: silver and money are in the case (فضة و مال في الحقيبة, fiḍḍatun wa mālun fī al-haqībati). As the sentence is not diacritized, the third word could be annotated as the name money (مَال, māl) or as the verb tilt (مَالَ, māla). The disambiguation here is obvious because silver (فضة, fiḍḍah) is a name and the preposition "and" (و, wa) cannot link a name with a verb. In addition, the word placed after prepositions must be a noun. And the word placed after particles affecting verbs must be a verb. In the sentence: رجل قوية, the first word can be either the name man (رَجُل, raǧul) or the name feet (رِجْل, riğlun). Regarding the attributive and predicative adjectives agreement, the predicative adjective "strong" (قوية, qawiyaah) is feminine. So رجل must be feet (رِجْل, riğl) that is also feminine. The syntactic rules enable us to do automatic disambiguation [5, 15].

Our disambiguation approach is based on cooperation between the morphological analyzer and the parser. The morphological analyzer produces all possible interpretations of the textual Arabic word. Disambiguation would be resolved by applying certain types of constraints that are defined with the grammar rules (See Fig. 2). These constraints lead to a correct parse, it could resolve the ambiguity.

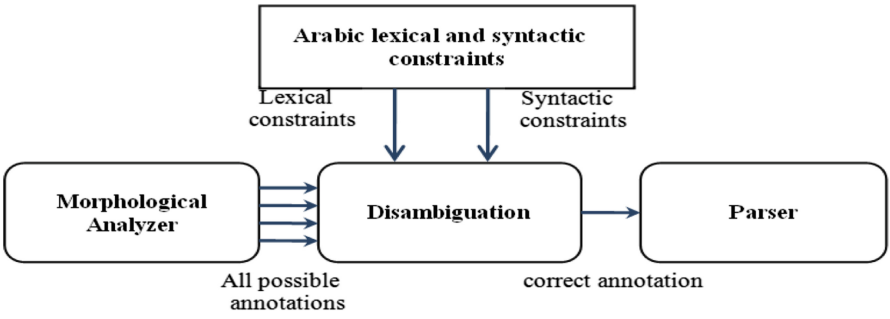


Fig. 2. Disambiguation schema.

3.3 Mapping Between the Lexicon Classes and the Nominal Sentence Components

All natural languages share the same structure which mainly consists of a nuclear predication with eventually extra elements (complement). The nuclear predication is mandatory. In Arabic, the attribution (الإسناد, al-'isnād) is the predication relation which holds between two syntagms in a sentence. The Arabic sentence is comprised of two required components: the predicate and the subject (المسند و المسند إليه, al-musnad wa al-musnad-'ilayh), which affect the sentence meaning [2, 3, 20, 21]. The predicate may precede or follow the subject whether in the nominal or the verbal sentence. Pre-position (التقديم, a-ttaqdīm) and post-position (التأخير, a-tta'ẖīr) are restricted by some conditions (أحكام التقديم و التأخير, aḥkām al-ttaqdīm wa al-tta'ẖīr). In the case of a nominal sentence, the predicate is the comment (الخبر, al-ẖabar) and the subject is the topic (المبتدأ, al-mubtada'). When the subject follows the predicate, we have to do with a pre-posed comment (خبرمقدم, ẖabar muqaddam) and post-posed topic (مبتدأ مؤخر, mubtada' mu'aḫẖar). Some modifiers may come before them regardless of their position, and consequently affect their diacritization. As example, let us consider the following sentence: the boy is assiduous (الولد مجتهد, al-waladu muğtahidun). If we begin the sentence with the particle (إنَّ,'inna, indeed), it changes the topic to accusative form. But if we replace the particle (إنَّ,'inna, indeed) with the particle (كَانَ, kāna, was) in the same sentence, it changes the comment to accusative form [7].

The Arabic grammarians have established the following rule describing the general structure of a sentence:

$$\textbf{الجملة} = [\text{ الصدر }] \ (\textbf{ المسند و المسند إليه }) \ [\text{ الفضلة }] \tag{1}$$

The sentence, al-ğumlah = [the head, al-ṣṣadr] **(the predicate, al-musnad and, wa the subject, al-musnad'ilayh)** [the complement, al-faḍlah]

Both of the predicate and the subject are mandatory in the Arabic sentence. In the other hand, the complement and the head are optional. In the context of simple Arabic Nominal sentence and regarding the lexicon classification, the head could be an incomplete verb, an interrogation particle, etc. The predicate could be an adjective, a prepositional phrase, etc. The subject is always a noun phrase. The complement is usually an incomplete noun or a prepositional/locative phrase. Therefore, a simple nominal sentence cover six kind of simple Arabic nominal sentence: when the comment could be either resembling adjective, derivative adjective (الصفة المشتقة, al-ṣṣefah al-mušttaqah(, verbal noun (المصدر, al-maṣdar), indefinite noun, prepositional phrase, or locative phrase (See Fig. 7). The Fig. 3 presents an example of simple Arabic nominal sentence.

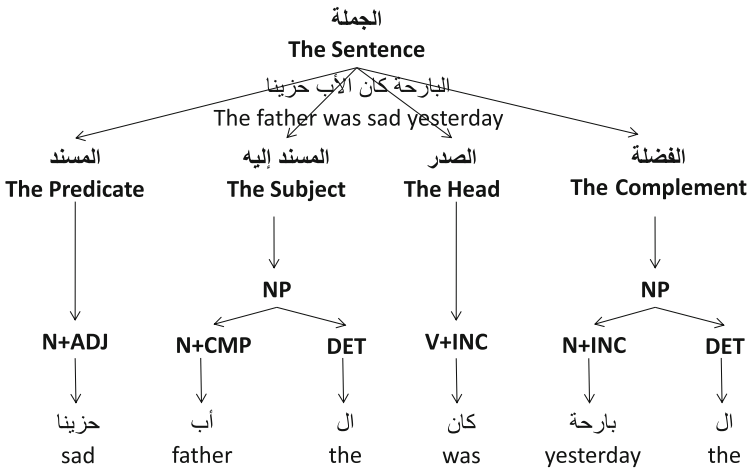


Fig. 3. Example of simple sentence structure.

4 The NooJ Linguistic Platform

NooJ is a development environment used to construct large-coverage formalized descriptions of natural languages, and apply them to large corpora, in real time [13, 16]. The descriptions of natural languages are formalized as electronic dictionaries, as grammars represented by organized sets of graphs. NooJ supplies tools to describe inflectional and derivational morphology, terminological and spelling variations, vocabulary (simple words, multi-word units and frozen expressions), semi-frozen phenomena (local grammars), syntax (grammars for phrases and full sentences) and semantics (named entity recognition, transformational analysis). In fact, NooJ allows linguists to combine in one unified framework Finite-State descriptions such as in XFST, Context-Free grammars such as in GPSG, Context-Sensitive grammars such as in LFG and unrestricted grammars such as the ones developed in HPSG.

NooJ is also used as a corpora processing system: it allows users to process sets of (thousands of) text files. Typical operations include indexing morpho-syntactic patterns, frozen or semi-frozen expressions (e.g. technical expressions), lemmatized concordances and performing various statistical studies of the results. NooJ is a freeware, linguistic engineering development environment used to formalize various types of textual phenomena (orthography, lexical and productive morphology, local, structural and transformational syntax) using a large gamut of computational devices (from Finite-State Automata to Augmented Recursive Transition Networks). NooJ includes tools to construct, test, debug, maintain and accumulate large sets linguistic resources, and can apply them to large texts [16].

5 Implementation

In a previous work, we have already implemented an Arabic dictionary in the NooJ platform. It is based on root and pattern properties. This dictionary consists of 160.000 lexical entries which are also obtained from flexional and derivational models [4]. However, this dictionary lacks a fine-grained classification allowing a correct syntactic analysis. Therefore, before the implementation in the NooJ platform, we must add new syntactic and semantic properties allowing a successful Arabic sentence parsing. Table 1 summarizes new defined properties holding the lexicon classification discussed in Sect. 3.1.

Table 1. Lexicon classification.

Property/ sub-property	Code	Example
- Noun, إسم, ’ism	N	
- Complete Noun, الإسم التام	N+CMP	قلم, pen, qalam
- Incomplete Noun, الإسم الناقص	N+INC	يوم, day , yawm
- Pronoun, الضمير, al-ddamīr	N+PRO	هو, he, howa
- Adjective, الصفة, al-ṣṣifah	N+ADJ	
- Resembling adjective الصفة المشبهة, al-ṣṣifah al-mušabbahah	N+ADJ +ARP	عظيم, great, ‘aẓīm
- Active Participle إسم الفاعل, ’ism al-fā´il	N+ADJ +AAP	كاتب, writer, kātib
- Passive Participle إسم المفعول. ’ism al-maf´ūl	N+ADJ +APP	مكتوب, wroten, maktūb
- Noun of Place, إسم المكان, ’ism al-makān	N +PLC	مطبخ, kitchen, maṭbaḫ
- Noun of Time, إسم الزمان, ’ism al-zzamān	N +TIM	مغرب, sunset, maḡrib
- Verb, فعل, fi‘l	V	
- Complete Verb, الفعل التام, al-fi‘l al-tām	V+CMP	
- Transitive 1	V+TR1	طلب, ṭalaba, to request
- Transitive 2	V+TR2	أعطى, ’aʿṭā, to give
- Transitive 3	V+TR3	أرى, ’arā, to show
- Intransitive	V+ITR	مات, to dead, māta
- Incomplete Verb, الفعل الناقص	V+INC	كان, to be, kāna
- Particle, حرف, ḥarf	PART	
- Annulling Particle	PART+ANN	لعل, la´alla, might
- Vocative Particle	PART+VOC	أيا, ’ayā, وا, wā: oh

After that, we implement some disambiguation rules that include three constraint types: lexical constraints, syntactic constraints, and agreement constraints. Theses constraints are implements as local grammars using the NooJ platform. Each analyzed sentence is matched with these grammars in a sequential mode in order to overcome

meaningless tags. The following local grammars (Figs. 4, 5 and 6) respectively summarize lexical, syntactic, and agreement constraints.

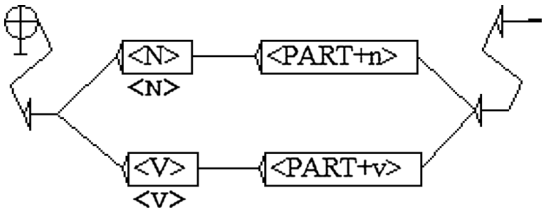


Fig. 4. Lexical constraint.

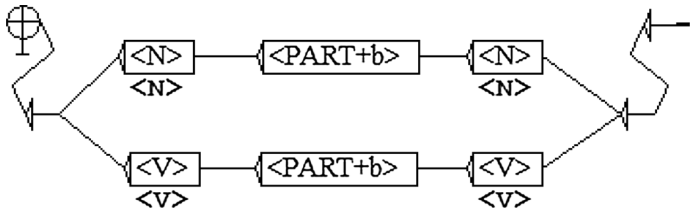


Fig. 5. Syntactic constraint.

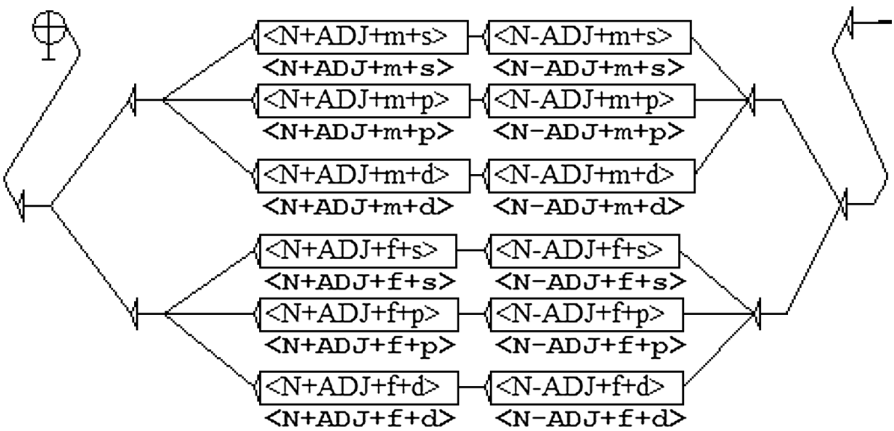


Fig. 6. Agreement constraint.

The final step in our contribution is the implementation of the simple Arabic nominal sentence structure. All in all, thirty structural grammar graphs, with seven levels of nesting, were implemented. All of them cover six kind of simple Arabic nominal sentence: Nominal sentence when the comment could be resembling and derivative adjective, a verbal noun, an indefinite noun, a prepositional phrase, or a locative phrase. Some of these grammar graphs are presented in Figs. 7 and 8.

In the NooJ linguistic platform, we implement a set of syntactic rules. These rules are based on the formula (1) describing the simple Arabic sentence structure.

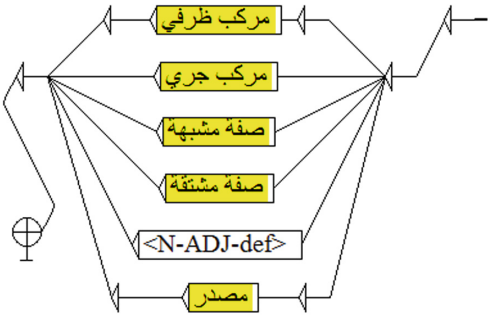


Fig. 7. Simple Arabic nominal sentence predicate values.

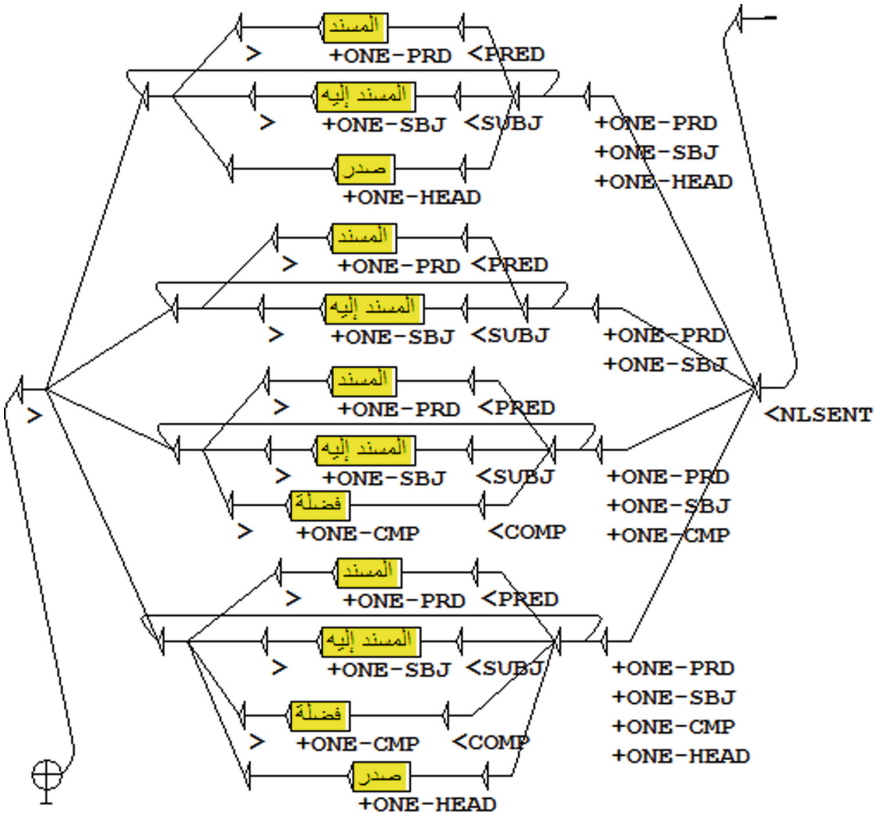


Fig. 8. First level of our grammar.

Our implementation takes advantage of the lexicon classification presented above. Figure 8 shows the first level of our grammar.

The graph shown in Fig. 8 handle and annotate the main components of the input sentence, and produce as output annotated parse tree(s) (see Sect. 3.3). our grammar is able to parse any nominal sentence regardless of the order of its components.

Table 2. Sentence components annotations

Abbreviation	Full form
COMP	Complement
INC. VERB	Incomplete verb
NLSENT	Nominal sentence
PRED	Predicate
SUBJ	Subject

Table 2 presents the list of abbreviations used in the sentence annotations produced by the first level of our grammar.

6 Results

To test the parser and the disambiguation local grammar on corpora, we have to segment corpora text into sentences. This task requires a particular processing which is not the aim of this work. So our test is applied on a text containing one hundred and

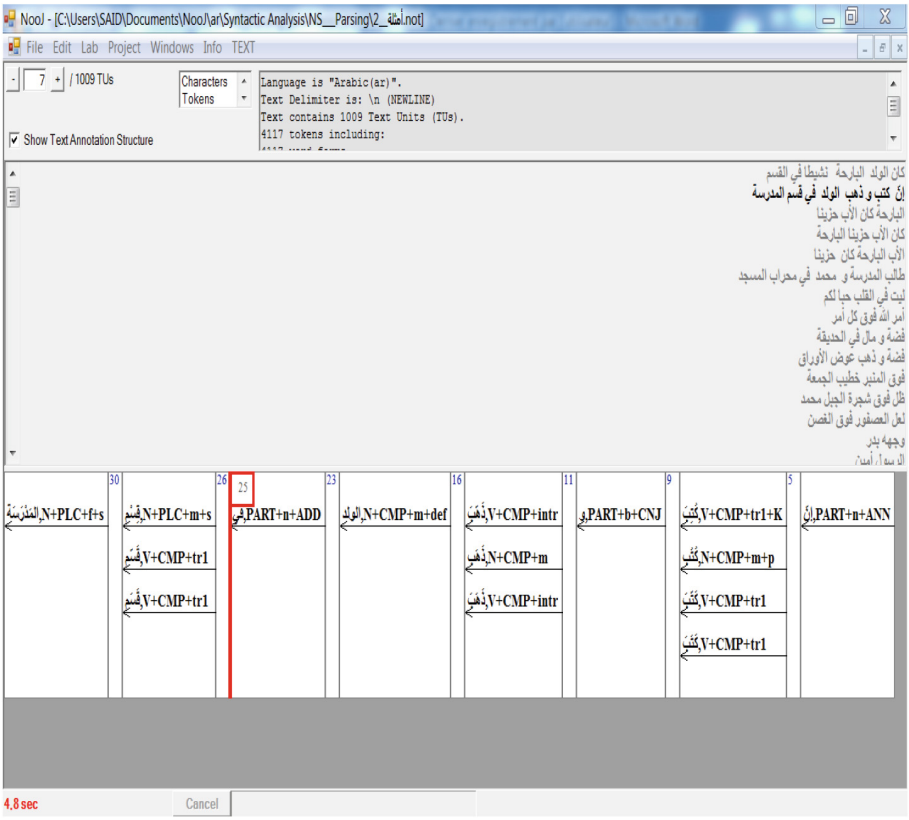


Fig. 9. Sentence annotations before disambiguation

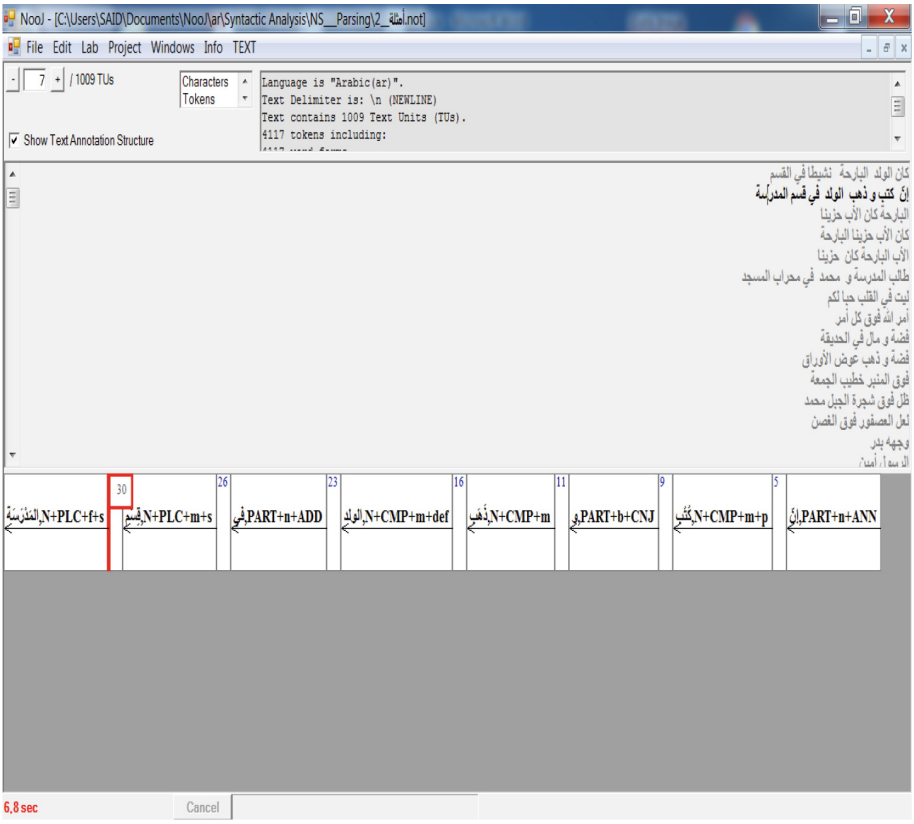


Fig. 10. Sentence annotations after disambiguation.

twenty nominal sentences having various structures. The rate of disambiguation is around eighty-six percent. This can be explained because some constraints are not yet implemented. However, we still have ambiguity in some sentences due to ambiguity feature of Arabic language. This issue could be solved once we implement a semantic analyzer beside the syntactic analyzer. Regarding the parsing task, the success rate of our analyzer is around ninety-five percent. This is obvious since some grammars are not yet implemented. Figure 9 presents the tagging of a sentence just before the disambiguation step. Figure 10 presents the tagging of the same sentence just after the disambiguation step and before the parsing step. Figures 11 and 12 contain the syntactic analysis result for the same sentence. We notice the success of the analysis even though the words order is not the same in each of them.

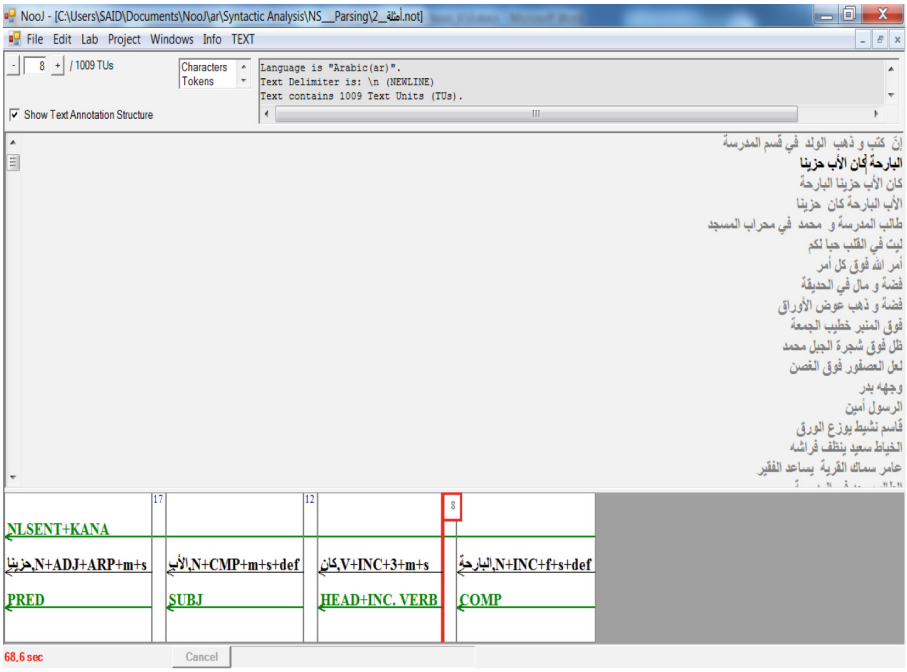


Fig. 11. Sentence annotations after the parsing step (1).

The NooJ Linguistic engine analyzes all possible interpretations for a not diacritized word, and add all possible morphological annotations in the NooJ Table Annotation Structure (TAS). Figure 9 shows an ambiguous NooJ TAS, result of the NooJ linguistic analysis, before applying our disambiguation local grammar.

After applying disambiguation local grammar, all impossible morphological annotations are filtered out from the NooJ TAS. Figure 10 presents a disambiguated NooJ TAS, which can be used for an efficient syntactic parsing and generation.

After the disambiguation step, we obtain a disambiguated sentence which is the input of our parser. The parser has to match parse tree(s) to the input sentence. Figures 11 and 12 show the NooJ TAS after the parsing step applied on two sentences. These sentences are similar but the order of their components is different. The parser returns the same syntactic annotations of the sentence components in the two proposed cases.

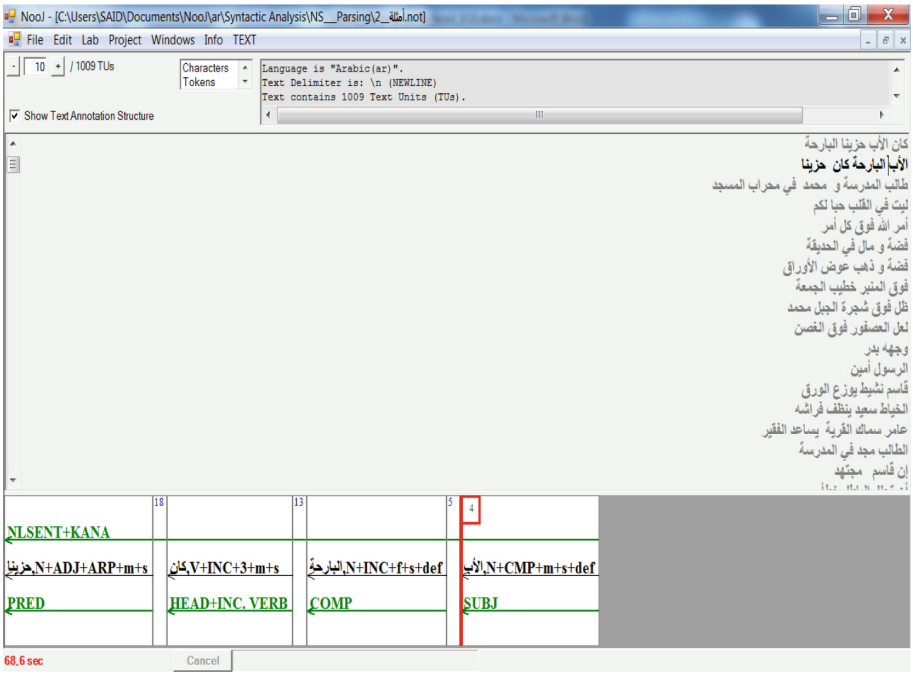


Fig. 12. Sentence annotations after the parsing step (2).

7 Conclusion and Perspectives

In this paper we presented our methodology of simple Arabic sentence parsing. This methodology consists in classifying the entries of an Arabic electronic dictionary regarding the category, implementing disambiguation rules, and creating syntactic grammars. The implementation is performed using the NooJ platform. If the platform NooJ allows to process all the stages of analysis (morphological, syntactic and semantic), our work is focused on the stage of syntactic analysis, with a preliminary stage of disambiguation.

Thus, we have implemented many transducers modeling a set of lexical and syntactic constraints in Arabic language. These transducers are applied sequentially. After that, with our structural grammars, we have analyzed several simple Arabic nominal sentences, disambiguated it automatically and generate their syntactic trees. These graphs add syntactic annotations.

Our method will not be limited to the simple nominal sentence, we will extend it to other types of Arabic sentence. So we will be able to syntactically analyze different text and corpora thereafter.

Once the Arabic analyzer is done, many issues could be solved such as automatic diacritics, Arabic sentences correction, and accurate translation. Also, other disambiguation rules could be implemented when the semantic analysis can be used.

References

1. Al Daoud, E., Basata, A.: A framework to automate the parsing of arabic language sentences. Int. Arab J. Inf. Technol. **6**(2), 191–195 (2009)
2. Assamirai, S.F.: Composition and Types of Arabic Sentence, 2nd edn. dar al kitab, Bagdad (2007)
3. Alsuhaibani, S.O.: The Verbal Sentence in Written Arabic. Thesis for the degree of Doctor of philosophy, University of Exeter, Ukraine (2012)
4. Blanchete, I., Mourchid, M., Mouloudi, A., Mbarki, S.: Formalizing Arabic inflectional and derivational verbs based on root and pattern approach using NooJ platform. In: Proceedings of the International NooJ Conference, NooJ 2017, Kenitra-Rabat, Morocco (2017)
5. Bourahma, S., Mbarki, S., Mourchid, M., Mouloudi, A.: Disambiguation and annotation of Arabic simple nominal sentences using NooJ platform. In: Proceedings of the International NooJ Conference, NooJ 2017, Kenitra-Rabat, Morocco (2017)
6. Fashwan, A., Alansary, S.: SHAKKIL: an automatic diacritization system for modern standard Arabic texts. In: The Third Arabic Natural Language Processing Workshop, Valencia, Spain (2017)
7. Habash, N.: Introduction to Arabic Natural Language Processing. Morgan & Claypool Publishers Series, San Rafael (2010)
8. Kassmi, R., Mourchid, M., Mouloudi, A., Mbarki, S.: Processing agglutination with a morpho-syntactic graph using NooJ. In: Proceedings of the International NooJ Conference, NooJ 2017, Kenitra-Rabat, Morocco (2017)
9. Khoufi, N., Aloulou, C., Belguith, L.H.: ARSYPAR: a tool for parsing the Arabic language based on supervised learning. In: The International Arab Conference on Information Technology, ACIT, University of Science & Technology, Sudan (2013)
10. Marton, Y., Habash, N., Rambow, O.: Improving Arabic dependency parsing with lexical and inflectional morphological features. In: Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages, Los Angeles, California, pp. 13–21 (2010)
11. Mourchid, M., Elfaddouli, N., Amali S.: Development of lexicons generation tools for Arabic: case of an open source conjugator. Int. J. Nat. Lang. Comput. IJNLC **5**(2), 13–25 (2016)
12. Mourchid, M.: Génération morphologique et applications. Thèse de Spécialité de 3ème Cycle, Université Mohammed V (1999)
13. Silberztein, M.: Formalizing Natural Languages The NooJ Approach. ISTE Editions, London (2016)
14. Silberztein, M.: Syntactic parsing with NooJ. In: The International NooJ Conference, Tozeur, Tunisia (2009)
15. Silberztein, M.: Disambiguation tools for NooJ. In: The International Nooj Conference, Budapest, Hungary (2008)
16. Silberztein, M.: NooJ Manual (2003). www.nooj4nlp.net
17. ابن الناظم : شر ح ابن عقيل على ألفية ابن مالك,الجزء الأول, الجزء الثاني, دار التراث, القاهرة, مصر (1980)
18. أبو زيد المقرئ الإدريسي : حروف المعاني في اللغة العربية دراسة تركيبية ودلالية, مؤسسة الإدريسي, الدار البيضاء (2016)
19. الحسين بن قاسم, الجنى : الداني في حروف المعاني, دار الآفاق الجديدة, بيروت (1983)
20. الرازي فخر الدين : نهاية الإيجاز في دراية الإعجاز, مطبعة الآداب ، القاهرة (1317 ه)
21. سيبويه: الكتاب, بولاق, القاهرة, (1316 ه)
22. نبيل علي : اللغة العربية والحاسوب, تعريب, القاهرة (1988)

Author Index