# JedAI: The Force Behind Entity Resolution

George Papadakis[1(✉)], Leonidas Tsekouras[2], Emmanouil Thanos[3], George Giannakopoulos[2], Themis Palpanas[4], and Manolis Koubarakis[1]

[1] University of Athens, Athens, Greece
{gpapadis,koubarak}@di.uoa.gr
[2] NCSR "Demokritos", Athens, Greece
{ltsekouras,ggianna}@iit.demokritos.gr
[3] University of Leuven, Leuven, Belgium
emmanouil.thanos@kuleuven.be
[4] Paris Descartes University, Paris, France
themis@mi.parisdescartes.fr

**Abstract.** We present JedAI, a toolkit for Entity Resolution that can be used in three different ways: as an open-source Java library that implements numerous state-of-the-art, domain-independent methods, as a workbench that facilitates the evaluation of their relative performance and as a desktop application that offers out-of-the-box ER solutions. JedAI bridges the gap between the database and the Semantic Web communities, offering solutions that are applicable to both relational and RDF data. It also conveys a modular architecture that facilitates its extension with more methods and with more comprehensive workflows.

## 1 Introduction

Linked Open Data are not as linked as they have been envisaged by the Linked Data Principles: a recent study revealed that 44% of its datasets are not connected with any other data sources [8]. To ameliorate this situation, Entity Resolution (ER) aims to interconnect the semantically equivalent resources in different datasets with `owl:sameAs` relationships. ER is manifested with two sub-tasks [1]: *(i)* Clean-Clean ER receives as input two overlapping data sources that are individually duplicate-free, $S$ and $T$, and aims to identify all pairs $<s \in S, t \in T>$ that correspond to the same real-world object. *(ii)* Dirty ER takes as input a single set of resources $S$ that contains duplicates in itself and aims to detect all pairs $\{s_i, s_j \in S : i \neq j\}$ that refer to the same real-world object.

To facilitate researchers, practitioners and simple users in applying both ER tasks, we hereby present the *Java gEneric DAta Integration Toolkit*, JedAI for short. JedAI offers a threefold functionality:

(1) JedAI constitutes an *open source library* that implements numerous state-of-the-art methods for all steps of the end-to-end ER workflow of Sect. 2. This workflow combines high time efficiency and scalability with high effectiveness [3,7]. The former aspect is accomplished by Steps 2–4, which use (meta-)blocking to significantly reduce the search space, omitting the comparison

of evidently irrelevant pairs of resources, while effectiveness emanates from Steps 5–6, which combine string similarity measures with advanced clustering methods that relax the transitivity of the equality relation. Every step exposes a well-defined API with detailed documentation. The code (in Java 8) is freely distributed through the Apache License V2.0, supporting both academic and commercial uses.

(2) JedAI constitutes a *desktop application* with an intuitive Graphical User Interface that can be used not only by experts in ER, but also by lay users. The user simply has to select a method for every step of the end-to-end workflow through an intuitive wizard. All methods are associated with a default configuration that has been experimentally verified to achieve the best performance on average [7], thus requiring no manual fine-tuning. All methods are also unsupervised, thus requiring no manual definition of link specifications. Finally, all methods operate in a schema-agnostic fashion, thus requiring no external domain knowledge (e.g., a training set or an ontology). As a result, the workflow that is formed by the user is carried out in a fully automatic way.

(3) JedAI can be used as a *workbench*, too. The number of methods that are available in every step yields more than 4,000 different possible workflows. JedAI facilitates users to compare in detail two or more executed workflows by reporting a large variety of performance measures in Step 7. Users can actually perform thorough experiments on top of most established benchmarks, since JedAI supports a wide variety of structured data formats (e.g., relational databases, CVS files) and semi-structured ones (e.g., RDF, XML, OWL files).

The code of the JedAI library along with several datasets for experimentation, the executable jar of the JedAI desktop application and videos presenting its features can be downloaded from https://github.com/scify/JedAIToolkit.

## 2 JedAI Workflow

The end-to-end ER workflow that is implemented by JedAI combines the blocking workflow proposed in [7] (Steps 2, 3 and 4) with the matching workflow used in [3] (Steps 5 and 6). In more detail, it comprises the following steps:

(1) **Data Reading** loads from the disk into main memory one (Dirty ER) or two (Clean-Clean ER) sets of resources along with the corresponding golden standard. It supports Semantic Web data contained in RDF, XML or OWL files as well as relational data contained in CSV files or SQL databases (e.g., mySQL).

(2) **Block Building** receives as input the data source(s) loaded by Data Reading and clusters their resources into a set of blocks that is returned as output. This is a mandatory step that drastically reduces the search space in order to ensure high time efficiency and scalability. At the moment, the user can select among 8 established methods. All of them use the schema-agnostic

blocking keys that were defined in [6]. Thus, they require no domain knowledge, placing every resource into multiple blocks in order to achieve high recall.

(3) **Block Cleaning** receives as input the blocks produced by Block Building. Given that they are overlapping (i.e., every resource participates in multiple blocks), they contain two types of unnecessary comparisons: the *redundant* ones, which repeat the same comparisons in different blocks, and the *superfluous* ones, which compare non-matching resources [6]. Block Cleaning discards both types of comparisons by enforcing constraints on the level of individual blocks. This results in significant gains in efficiency and scalability, though at the cost of slightly lower recall [7]. For this reason, this step is optional, yet it allows the user to choose one or more of the 4 methods that are currently available (i.e., all methods are complementary with each other).
(4) **Comparison Cleaning** receives as input a set of blocks and aims to clean it from its unnecessary comparisons, just like Block Cleaning. The difference is that Comparison Cleaning operates at a finer granularity, targeting individual comparisons. As a result, it takes more accurate decisions, but its performance is more time consuming than Block Cleaning [7]. Due to its cost in recall, it is an optional step, too. For now, the user can choose 1 out of 7 competitive methods.
(5) **Entity Matching** is a mandatory step that executes all comparisons that are contained in the set of block it receives as input. As output, it produces a *similarity graph*, with one node for every resource and one weighted edge for every compared pair of resources. The user can choose among 2 methods, which incorporate a plethora of established string similarity metrics [2].
(6) **Entity Clustering** is a mandatory step that receives as input the similarity graph of Entity Matching. Its goal is to partition its nodes into *equivalence clusters* such that every cluster contains all resources that correspond to the same real world object. At the moment, the user can select 1 out of 7 established methods, which have been experimentally evaluated in [3].
(7) **Evaluation & Storing**, the final step of our workflow, estimates the performance of the resulting set of equivalence clusters with respect to the established effectiveness measures (Precision, Recall and F-Measure). For this purpose, it relies on the golden standard that was given as input file in Step 1. This step also evaluates the time efficiency of the implemented workflow, assessing the overhead time of every individual step. Finally, the user is able to store the identified equivalence links in a variety of output formats, such as CSV. An example of this step's screen is illustrated in Fig. 2.

All methods included in every step are schema- and domain-agnostic, requiring no background knowledge from the user in order to apply them.

## 3 Architecture

JedAI has a *modular* architecture that is described in Sect. 2. Note that there is a separate module for every step in the workflow of Sect. 2. Each module exposes

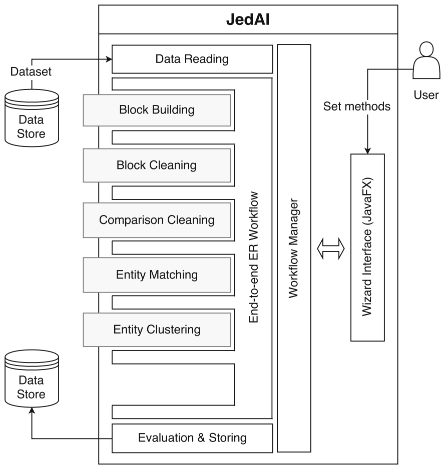
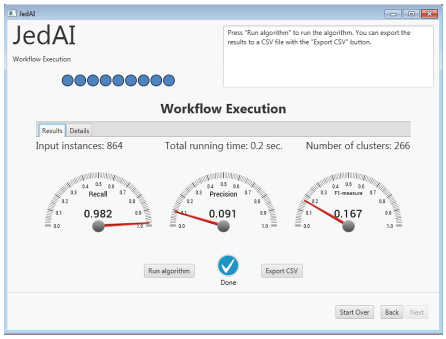


**Fig. 1.** JedAI architecture



Press "Run algorithm" to run the algorithm. You can export the results to a CSV file with the "Export CSV" button.

Workflow Execution

Input instances: 864 Total running time: 0.2 sec. Number of clusters: 266

Recall 0.982 Precision 0.091 F1-measure 0.167

**Fig. 2.** Evaluation & Storing screen.

a well-defined interface so that any method that implements it can be seamlessly integrated into that module. This makes it easy to add more ER techniques in the future, as they become available. Additionally, every interface takes special care to ensure that the functionality of each method is well-documented and that its configuration parameters are associated with a short explanation of their role along with their possible values. Finally, JedAI's architecture is *extensible* with additional modules that could add more steps to the workflow of Sect. 2. This is illustrated by the empty placeholder right after the Entity Clustering module in Fig. 1; however, a new module/step could be placed anywhere in the current workflow. An example is Ontology Matching before Entity Matching or Block Building for higher effectiveness.

## 4 Demonstration

The goal of our demonstration is to showcase how JedAI can apply a series of ER workflows to several established benchmark datasets in a straightforward way, i.e., without requiring any manual fine-tuning or domain knowledge. To this end, we will use JedAI's desktop application to load some established CSV datasets from [4] and RDF datasets from the Ontology Alignment Evaluation Initiative (OAEI). All datasets will be loaded in their original format, without any preprocessing. Then, the user will be able to form an end-to-end ER workflow with a couple of clicks in the screen of every step. Finally, a screen similar to Fig. 2 will present the performance of the formed workflow. Different workflows or different configurations will be applied to the same dataset(s) with the same 7-step procedure in order to compare the resulting performances.

## 5 Related Work

Entity Resolution has been extensively studied in the literature as a special case of the more general task of Link Discovery [5]. In fact, OAEI organizes a special track for Instance Matching and several frameworks have been proposed for facilitating Link Discovery, such as RiMOM, Silk and LIMES. A recent survey can be found in [5]. These frameworks are similar to JedAI in the sense that they combine methods for high efficiency (e.g., blocking) with methods for high effectiveness (e.g., thresholds on string similarities).

However, JedAI goes beyond these frameworks in the following ways: *(i)* Most frameworks for Link Discovery focus exclusively on Clean-Clean ER, whereas JedAI covers Dirty ER, too. *(ii)* Link Discovery frameworks take as input only RDF data, while JedAI is able to process structured data, too, supporting CSV files and relational databases. *(iii)* JedAI offers out-of-the-box ER solutions, due to the schema-agnostic functionality of its methods and the default configuration that is associated with every one of them. Thus, it requires no manual fine-tuning, unlike most Link Discovery frameworks. *(iv)* JedAI is suitable for both lay and power users. The former can simply use the intuitive GUI, while the latter can use it as a library, too, through its well-defined API. In contrast, all Link Discovery frameworks require some domain knowledge from the user. *(v)* JedAI works as a workbench, too, allowing users to compare the performance of more than 4,000 different ER workflows that can be formed by combining state-of-the-art methods from every step. *(vi)* There is small overlap in the methods offered by JedAI and those offered by the Link Discovery frameworks. The former currently focuses on schema-agnostic methods, while the latter cover ontology-based methods.

## 6 Conclusions

We present JedAI, a toolkit that can be used as Java library, a desktop application and a workbench for domain-independent, out-of-the-box ER solutions. Its capabilities will be demonstrated by applying it to a series of established relational and RDF datasets, without requiring any manual fine-tuning or background knowledge. In the future, we plan to participate in the OAEI Instance Matching task and to integrate more methods that are based on ontologies for even better performance. We also plan to parallelize JedAI using Apache Spark.

**Acknowledgments.** This work has been supported by the project "Your Data Stories", which is funded by EU Horizon 2020 programme under grant agreement No. 645886. We would also like to thank Oktie Hassanzadeh for sharing with us the implementation in C of the clustering algorithms examined in [3].

## References

1. Christophides, V., Efthymiou, V., Stefanidis, K.: Entity Resolution in the Web of Data. Morgan & Claypool, San Rafael (2015)
2. Cohen, W., Ravikumar, P., Fienberg, S.: A comparison of string distance metrics for name-matching tasks. In: IIWeb, pp. 73–78 (2003)
3. Hassanzadeh, O., Chiang, F., Miller, R., Lee, H.: Framework for evaluating clustering algorithms in duplicate detection. PVLDB **2**(1), 1282–1293 (2009)
4. Köpcke, H., Thor, A., Rahm, E.: Evaluation of entity resolution approaches on real-world match problems. PVLDB **3**(1), 484–493 (2010)
5. Nentwig, M., Hartung, M., Ngomo, A., Rahm, E.: A survey of current link discovery frameworks. Semant. Web **8**(3), 419–436 (2017)
6. Papadakis, G., Alexiou, G., Papastefanatos, G., Koutrika, G.: Schema-agnostic vs schema-based configurations for blocking methods on homogeneous data. PVLDB **9**(4), 312–323 (2015)
7. Papadakis, G., Svirsky, J., Gal, A., Palpanas, T.: Comparative analysis of approximate blocking techniques for entity resolution. PVLDB **9**(9), 684–695 (2016)
8. Schmachtenberg, M., Bizer, C., Paulheim, H.: Adoption of the linked data best practices in different topical domains. In: Mika, P., et al. (eds.) ISWC 2014. LNCS, vol. 8796, pp. 245–260. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-11964-9_16