IOTA: Interlinking of heterogeneous multilingual open fiscal DaTA

Fathoni A. Musyaffa[a,*], Maria-Esther Vidal[e,d], Fabrizio Orlandi[c], Jens Lehmann[a,b], Hajira Jabeen[a]

[a] *University of Bonn, Bonn, Germany*
[b] *Fraunhofer IAIS, Dresden, Germany*
[c] *ADAPT Centre, Trinity College Dublin, Ireland*
[d] *Universidad Simón Bolívar, Venezuela*
[e] *German National Library of Science and Technology, Hannover, Germany*

Abstract

Open budget data are among the most frequently published datasets of the open data ecosystem, intended to improve public administrations and government transparency. Unfortunately, the prospects of analysis across different open budget data remain limited due to schematic and linguistic differences. Budget and spending datasets are published together with descriptive classifications. Various public administrations typically publish the classifications and concepts in their regional languages. These classifications can be exploited to perform a more in-depth analysis, such as comparing similar items across different, cross-lingual datasets. However, in order to enable such analysis, a mapping across the multilingual classifications of datasets is required. In this paper, we present the framework for Interlinking of Heterogeneous Multilingual Open Fiscal DaTA (IOTA). IOTA makes use of machine translation followed by string similarities to map concepts across different datasets. To the best of our knowledge, IOTA is the first framework to offer scalable implementation of string similarity using distributed computing. The results demonstrate the applicability of the proposed multilingual matching, the scalability of the proposed framework, and an in-depth comparison of string similarity measures.

Keywords: data interlinking, budget and spending data, string similarity measure, open data, translated string matching framework, cluster computing

1. Introduction

Public governments and international bodies have increasingly started to publish open government data. This data plays an important role in improving transparency and compliance (Shadbolt et al., 2012; Tygel et al., 2016), democratic control and political participation (Huijboom & Van den Broek, 2011; Tygel et al., 2016), government efficiency and effectiveness (Tygel et al., 2016), and law enforcement (Huijboom & Van den Broek, 2011). By opening government data, not only the government-citizen barrier can be lowered, but also comparative analysis among datasets from different regions can be enabled (Tygel et al., 2016; Huijboom & Van den Broek, 2011). Budget data is one of the most important (The World Wide Web Foundation, 2017) and most published data (Open Knowledge International, 2017). Budget data has remained in the top three domains published for open data in years 2013-15. 98 countries out of observed 122 countries provided their budget data openly. OpenSpending[1] states that they host more than 3200 fiscal (budget and spending) datasets openly as of July 2019, which comprises more than 132 million fiscal records. The growing government budget and spending data makes it possible to perform cross-administration budget

*Corresponding author
Email addresses: musyaffa@cs.uni-bonn.de (Fathoni A. Musyaffa), maria.vidal@tib.eu (Maria-Esther Vidal), fabrizio.orlandi@tcd.ie (Fabrizio Orlandi), jens.lehmann@iais.fraunhofer.de (Jens Lehmann), jabeen@cs.uni-bonn.de (Hajira Jabeen)

[1] https://openspending.org/

analysis. Due to the decentralized nature of the data publication and creation, the published budget and spending data are often disparate, it is published in different structures, formats, languages, metrics (e.g., feet/meters), granularity (e.g., years/months), and possess different forms of heterogeneity (Musyaffa et al., 2018c). In addition, the data are normally found to be incomplete and of low-quality (The World Wide Web Foundation, 2017). In order to deal with this heterogeneity, open fiscal data is often published along with non-standardized classifications providing additional information about particular concepts.

Owing to the structural complexity and the size of budget data, the automated cross-linking of these datasets is a challenging task. In order to do so effectively, it is important to develop mappings of similar attributes among different datasets available in different languages. Several efficient machine translation tools(Wu et al., 2016) exist to solve the problem of multilingual data. After the translation, various string similarity measures(ref) can be used to map similar concepts. However, string comparison is a computationally expensive task, especially when there is a high volume of concepts to be compared. Therefore, this task is not feasible for large scale data using a single machine. One of the recent in-memory distributed computing framework Apache Spark (Zaharia et al., 2016), can provide a scalable solution to solve complex tasks like mappings over large data.

The above-mentioned challenges and existing technologies have inspired us to design and propose a framework that uses fiscal-data-classifications, machine-translation and string-similarity in a distributed and scalable manner for interlinking of open fiscal data. In this paper, we present the framework for Interlinking of Heterogeneous Multilingual Open Fiscal DaTA (IOTA). To the best of our knowledge, IOTA is the first approach that: (1) Uses fiscal data classifications in conjunction with machine translations to provide mappings for heterogeneous and cross-lingual data coming from different regions. Three language pairs (German-Spanish, German-French, and Spanish-French) have been tested with this approach. (2) Provides a comparative analysis of 19 different string similarity measures for fiscal data linking. (3) Uses the distributed scalable computing framework to enable complex string similarity assessment over large datasets. In short, IOTA is designed to enable multi-regional, context-free, open fiscal data interlinking in a scalable manner.

The remaining paper is organized as follows: the preliminary concepts are defined in section 2, and the use case scenario is provided in section 3, Subsequently, section 4 summarizes works related to this paper. A description of IOTA is outlined in section 5. The evaluation is provided in section 6, with experimental result, configuration as well as the discussion. Finally, this paper is concluded in section 7.

2. Preliminaries

Open budget and spending data usually contain several components for observation, such as its measures, dimensions, and attributes (Dudaš et al., 2015). A *measure*, in spending/budgeting context, contains the amount of money allocated for a particular activity. A measure may also contain information such as population or budget/spending as the percentage of GDP. The *dimension* defines the measure in more detail, e.g., the classification (elaborated later) to which the observation belongs. An observation also contains temporal dimension for the observed measure. The *attribute* provides more precise information on the observation, for example, metrics (e.g., currency: € or £), precision level, or the measurement unit (e.g., km or meter). The combination of dimensions make an observation unique, and the availability of attribute clarify the observation in more detail.

In this paper, we refer to *classification* as a set of controlled terms published by respective official bodies to categorize budget/spending items, consisting of concise textual labels. Classification can be exploited further to improve data reusability and data comprehension, or perform data interlinking. These classifications can also be referred to as *vocabulary* or *code list*. The common classifications that are published by open data initiatives include:

- *Functional classification* describes the expense usage (e.g., `Vivienda y Urbanismo` or, translated into English, `Housing and Urbanism` concept as found in the 2013 Aragon Government Budget[2]).

[2] https://opendata.aragon.es/catalogo/presupuesto-gobierno-aragon-2013

- *Administrative classification* states which administrative office is responsible for a particular budget line (e.g., `Secretaría General Técnica de Obras Públicas, Urbanismo, Vivienda y Transportes`, or in English, `General Technical Secretariat of Public Works, Urban Planning, Housing, and Transportation` found in the same Aragon datasets).

- Other classifications, include: *economic classification* (e.g., `capital transfers, real investments`), *procurement items* (e.g., `Agricultural products`, as well as `Electricity and heating`), and so on.

Some classifications are standardized by international bodies. For example, *Classification of the Functions of Government*/COFOG (United Nations Statistics Division (UNSD), 1999), a functional classification developed by the United Nations and *Common Procurement Vocabulary*/CPV (European Commission, 2008), a procurement item classification by the European Union). A detailed outlook on the differences in budget and spending data classification is presented in our earlier work on fiscal heterogeneity (Musyaffa et al., 2018c). These classifications are published by different public administration authorities, therefore the concepts may not be standardized hence leading to data integration problems. In order to deal with complex computational tasks on integrating large scale data, cluster computing is mostly utilized.

Cluster computing can be defined as a collection of computers that jointly perform a given task in distributed manner. Cluster computing software framework, such as Apache Spark RDD allows computations to be performed in-memory within large clusters in a fault-tolerant manner (Zaharia et al., 2012). In case the computed RDD does not fit in the host memory, Spark automatically performs *spill to disk* operation which moves the RDD from host RAM to host disk. Apache Spark offers internal optimizations, such as an optimized operation for Cartesian join. Since Apache Spark requires *cluster manager* as well as *distributed file system*, Apache Spark provide *Spark Standalone* as a cluster manager and *Hadoop Distributed File System* (HDFS) as a distributed file system, among others. HDFS is a file system that is known for its scalabilitity, portability, fault-tolerant, and distributed manner with a master/slave architecture.

IOTA uses a set of string similarity measures to search the effective string similarity measures to find mappings between translated concepts. The *py_stringmatching* library[3] is used for string similarity measure calculation in IOTA. The list of compared similarity measures is provided in Table 5. Five main similarity measure categories are presented and used for comparison in this paper (*py_stringmatching* Documentation, 2016):

- *Sequence-based* (see Table 2) for similarity measures in which the inputs are considered as a sequence of characters.

- *Set-based* (see Table 3) for similarity measure in which the inputs are considered as tokens (i.e. words).

- *Hybrid-based* (see Table 3) for similarity measures which combines set-based and sequence-based similarity measures.

- *Bag-based* (see Table 4) for similarity measures that collect tokens as bags in which a token in these similarity measures could appear multiple times.

- *Phonetic-based* (see Table 4) for similarity measures that mimic string pronunciation.

3. Motivating example and use case

Open fiscal concepts published by different public administrations are often multilingual and there is no indication if two words have a similar meaning. For example, in Table 1, where datasets from the Aragon government (in Spanish) and from the municipality of Thessaloniki (in Greek) does not indicate that the concepts within the table have similar or related meaning regarding the functional classification item *culture*. If a *mapping* exist between two concepts from different datasets, further analysis can be made possible.

[3] https://pypi.org/project/py-stringmatching/

		Data A	Data B
Functional Classification	Language/Adm.	ES/Aragon	EL/Thesssaloniki
	Code	45	6471
	Label	Cultura	Έξοδα πολιτιστικών δραστηριοτήτων
Similar?	Translation	Culture	Cost of cultural activities

Table 1: A motivating example: functional classifications originating from Aragon (in Spanish) and from Thessaloniki (in Greek) which actually represent a similar concept of *culture* for the public budget. Each concept typically has their own code and label in the publisher's respective language, without indication that both concepts are, in fact, similar. Both classifications are published in separate spreadsheet documents.

Integrating classifications from different datasets allows at least two use cases. First, it allows comparison of the allocated/spent budget for a particular classification item (for example, culture, public transport, and so on), even when the datasets' classifications are published in different languages. Second, the integrated classifications could be mapped to public semantic knowledge bases (e.g., Wikidata, DBpedia), to enrich the concepts with additional information (such as an instantiation within within certain ontologies and leverage word sense hierarchies). Both use cases allow a deeper understanding of the budget and spending datasets. More precisely, they allow data discovery and reusability which would provide actionable insight for public administrators, civil communities, NGOs, stakeholders and most importantly the citizens who fund the city with their taxes.

4. Related work

Our work involves several topics, ranging from Open Fiscal Data and Open Data domain, as well as multilingual datasets mapping and open budget and spending data platforms. This section briefly covers the state of the art for these topics separately.

4.1. Open fiscal data analytics and platforms

The state of the art in open fiscal data analytics and harmonization is limited. Our previous work discusses the issues and recommendations for open fiscal data quality (Musyaffa et al., 2018a). The analysis of heterogeneity within open fiscal datasets is discussed in Musyaffa et al. (2018c). These heterogeneities can be minimized if the datasets are constructed following data models that comply with the particular specification, since one of the key requirement of 'government data quality, authority and governance' is metadata specification, and data documentation standards (Bertot & Choi, 2013). For fiscal data to be more reusable, available open budget and spending data specification need to be used, such as the Open Fiscal Data Package[4] for tabular datasets, or the OpenBudgets.eu data model[5] for RDF data. Providing reusable open data can significantly reduce the costs of reuse, adaptation, and innovation for third parties. These third parties are proven to be willing to develop tools and services for consuming and analyzing government data (Ding et al., 2010).

[4] http://frictionlessdata.io/specs/fiscal-data-package/
[5] https://openbudgets.eu/resources/2016/11/17/open-budgets-data-model-and-landscape/

Sequence-based Similarity Measure		
Name	Description	Formula
Bag Distance	Counts characters in each string x and y as a multiset, subtracts the difference between elements in x and y as well as between the difference of elements between y and x, and chooses the maximum element count from these numbers (Bartolini et al., 2002).	$dBD(x,y) = max(\|x-y\|, \|y-x\|)$. And can be normalized by: $sE(x,y) = 1 - \frac{dBD(x,y)}{max(\|x\|,\|y\|))}$
Levenshtein	Measures the distance of two given strings based on how much minimum edit cost (insert/delete/substitute) are needed to make two strings identical (Levenshtein, 1966). Also known as *Edit Distance*.	Given $d(x,y)$ is the edit distance between strings x , y, normalized Levenshtein similarity: $sL(x,y) = 1 - \frac{d(x,y)}{max(length(x), length(y))}$
Jaro	Counts how many common characters c are similar between strings x,y, and how many transpositions t are needed to make these common characters have a similar sequence (Jaro, 1980; Doan et al., 2012).	$sJaro(x,y) = \frac{1}{3 \times [\frac{c}{\|x\|} + \frac{c}{\|y\|} + \frac{c-\frac{t}{2}}{c}]}$
Jaro-Winkler	Improves *Jaro* by considering two extra parameters: the maximum length l of common prefix between two strings and the weight w considered for the prefix (Winkler, 1993; Doan et al., 2012).	$sJW(x,y) = (1 - l \times w) \times jaro(x,y) + l \times w$
Ratio	Utilizes parameters M as a total number of matches between elements in the strings x and y, and T as the total number of elements in both strings (Cohen, 2011). Score is normalized by dividing the result by 100.	$sR(x,y) = 2 \times \frac{M}{T} \times 100$
Partial Ratio	Compares the shorter string of length n with every sub-string of length n from the longer string. The maximum similarity score from these comparisons are provided as the partial ratio similarity score. Suppose between compared string x and y, x is the shorter string with length n. y is splitted into n-gram with length of n, resulting a set of tokens y with m member. $B_y = B_1, ..., B_m$ (Cohen, 2011). Score is normalized by dividing the result by 100.	$sPR(x,y) = max(\sum_{i=1}^{m} ratio(x, B_i))$
Partial Token Sort	Converts two strings into tokens, sorting the tokens, and calculates the partial ratio similarity score of calculated strings (Cohen, 2011). The score is normalized by dividing the result by 100.	
Token Sort	Splits two strings into tokens, sorts the tokens and calculates the ratio similarity score (Cohen, 2011). The score is normalized by dividing the result by 100.	

Table 2: An overview of eight sequence-based string similarity measures used in the experiment and their respective formula. Some similarity measures (Token Sort and Partial Token Sort) use formula from other similarity measures. Some of the similarity scores are not normalized by default, those are Bag Distance, Levenshtein, Ratio, Partial Ratio, Partial Token Sort, and Token Sort similarity.

Set-based Similarity Measure		
Name	Description	Formula
Cosine-Ochiai	Computes the intersection between two sets of tokens, divided by the square root of the multiplication between the size of both token sets. This is A derivative of cosine's algorithm known as Ochiai coefficient (Jiang et al., 2014; *py_stringmatching* Documentation, 2016).	$sC(x,y) = \frac{\lvert B_x \cap B_y \rvert}{\sqrt{\lvert B_x \rvert . \lvert B_y \rvert}}$
Dice	Also known as Sørensen-Dice coefficient. It is calculated by twice the size of the intersection between two sets of tokens, divided by the size of both token sets (Sørensen, 1948; *py_stringmatching* Documentation, 2016).	$sD(x,y) = 2 \times \frac{\lvert B_x \cap B_y \rvert}{\lvert B_x \rvert + \lvert B_y \rvert}$
Jaccard	The division between the intersection size of two sets and the union size across the sets (Jaccard, 1901; Doan et al., 2012).	$sJacc(x,y) = \frac{\lvert B_x \cap B_y \rvert}{\lvert B_x \cup B_y \rvert}$
Overlap Coefficient	Indicates the overlap between two sets by dividing the intersection size between two token sets with the minimum size from of the two sets (Vijaymeena & Kavitha, 2016).	$simOC(x,y) = \frac{\lvert B_x \cap B_y \rvert}{min(\lvert B_x \rvert, \lvert B_y \rvert)}$
Tversky Index	The division between intersection size of the token sets with: the sum of intersection between sets, the number of items only available on the first token set multiplied by a coefficient α, and the number of element only available in the second token sets multiplied by a coefficient β (Tversky, 1977).	$sT(x,y) = \frac{\lvert B_x \cap B_y \rvert}{\lvert B_x \cap B_y \rvert + \alpha \lvert B_x - B_y \rvert + \beta \lvert B_y - B_x \rvert}$
Hybrid-based Similarity Measure		
Generalized Jaccard	Calculated by 1) converting compared strings x, y into two sets B_x , B_y; 2) calculating the string similarity s between tokens across the two sets (hence Cartesian product is involved); 3) filtering the string similarity value s so that s is larger than specified threshold α. The result of this filtering is a bipartite graph mapping between B_x and B_y with similarity score $s > \alpha$ and collected into a graph M, which is then used to calculate the Generalized Jaccard similarity score; 4) getting the maximum similarity pairs s from graph M, and use the pair with maximum similarity s to calculate the final score.(On et al., 2007; Doan et al., 2012).	$sGJ(x,y) = \frac{\sum_{(x_i,y_j) \in M} s(x_i,y_j)}{\lvert B_x \rvert + \lvert B_y \rvert + \lvert M \rvert}$
Monge-Elkan	Also requires specifying a string similarity as a parameter name. Calculated with the following steps: 1) compared strings x, y is tokenized to $x = A_1, ...A_n$ and $y = B_1, ...B_m$; 2) string similarity scores are counted against each token from the other set; 3) the maximum similarity score from each set is then taken from the two sets and then averaged. String similarity function $s'()$ is the chosen string matching similarity measure parameter (Monge et al., 1996; Doan et al., 2012).	$sME(x,y) = \frac{1}{n} \sum_{i=1}^{n} \max_{j=1}^{m} s'(A_i, B_j)$
Soft TF/IDF	The calculation is done by: 1) computing a similarity score between tokens;, 2) filtering the tokens using threshold;, and 3) calculating similarity score using TF/IDF vectors along with filtered similarity score (Bilenko et al., 2003). In this experiment, Jaro is used as the secondary string similarity measure.	

Table 3: An overview of five set-based and hybrid-based string similarity measures and their respective formula used in our experiment: Ochiai as a derivative of Cosine similarity (will be referred later here as Cosine), Dice (also known as Sørensen-Dice Coefficient), Jaccard, Overlap Coefficient and Tversky Index. In the set-based similarity metrics part, B_x and B_y are tokens generated respectively from compared strings x and y. All the resulting values from these similarity measures fall in the range of [0,1].

Bag-based Similarity Measure	
Name	Description
TF/IDF	Takes into account term frequency to measure the similarity between two documents, and offset the similarity by the inverse document frequency so that commonly-appearing-terms' importance are discounted (Manning et al., 2008).
Phonetic-based Similarity Measure	
Soundex	To mimic pronunciation, Soundex replaces or removes characters from each compared strings, ended by examining processed strings. The steps are: 1) keep the first letter of each compared strings. 2) remove any occurrence of W, H, Y and vowels (A, E, I, O, U). 3) replace B, F, P, V with 1; C, G, J, K, Q, S, X, Z with 2; D, T with 3; L with 4; M, N with 5; R with 6. 4) remove any consequential identical digits (e.g., '22' to '2'). 5) keep only the first four characters but if the total length is less than four characters, the digit '0' is appended until it has four characters. 6) compare the processed strings which result in binary similarity score (Odell & Russell, 1918; Doan et al., 2012; Zobel & Dart, 1996).
Editex	Editex is a Soundex similarity modification with different letter groups to represent a more accurate pronunciation and allows some characters to be on more than one of nine letter groups: group 0 = {A, E, I, O, U, Y}, 1 = {B, P}, 2 = {C, K, Q}, 3 = {D, T}, 4 = {L, R}, 5 = {M, N}, 6 = {G, J}, 7 = {F, P, V}, 8 = {S, X, Z}, 9 = {C, S, Z}; in which {W, H} is removed. Editex utilizes a Levenshtein-like similarity measure to compare processed strings (Zobel & Dart, 1996).

Table 4: An overview of bag-based and phonetic-based string similarity metrics used in the experiment and applicable formula. Due to the complexity of some similarity measures, it is not possible to squeeze summarized formula in this table. The similarity score of these algorithms are normalized by default. Soundex yields binary decision by default, while Editex needs the similarity score to be normalized.

An important part of big data in an e-Government is the implementation of a robust architecture and data platform (Bertot & Choi, 2013). Following our previous work, a specific platform was devised for open spending and budget datasets. This platform, OpenBudgets.eu (Musyaffa et al., 2018b), provides a materialized and budget/spending-specific architecture for consuming Open Budget and Spending data. The OpenBudgets.eu platform is more concrete when it is compared to an earlier, related work, DIGO, that proposes a *conceptual* open data architecture (Machado & de Oliveira, 2011). Another work, Data-gov Wiki (Ding et al., 2010), converts raw data from US data.gov open data portal into RDF. In contrast to Data-gov Wiki, OpenBudgets.eu platform specifically integrates tools to upload, annotate, transform, visualize and analyze datasets from budget and spending domain. However, a missing component of the OpenBudgets.eu platform is a mapping tool that could map concept labels from classifications by different publishers and in different languages.

Budget comparison across different public administrators could potentially be made if there is a *mapping across labels* from different languages. This mapping is a part of *data interlinking* cycle, and according to Attard et al. (2015), data interlinking is one of the eight elements within Open Data Life Cycle, and is particularly crucial for data exploitation stage. There are often similar concepts provided in different languages, but there is a rare chance that a mapping across these labels from different languages exists (see a related survey from Musyaffa et al. (2018a)). Our work in this paper addresses the mapping challenge by experimenting with a framework consisting of machine translation, multiple similarity measures, and a cluster computing architecture to find which similarity measures are more suited to create mappings for concepts originating from different languages.

4.2. Multilingual concept mapping

4.2.1. Distributional semantics

Firth (1957) states that the words surrounding a word in question characterize its meaning. This is a basis for distributional semantics which uses the distributional properties in a large data sample for finding semantic similarities across language items. In the past few decades, there have been several distributional

semantic modelling approaches, such as Latent Semantic Analysis (LSA) (Deerwester et al., 1990) and Hyperspace Analogue to Language (HAL) (Lund, 1995). Another approach, word embedding, has been gaining popularity in the past few years. Word embedding maps words and phrases within a vocabulary to vectors of real numbers using a variety of methods. One of these methods uses a shallow neural network to learn this mapping coined as Word2Vec (Mikolov et al., 2013). Using Word2Vec, the semantic similarity between words of similar language can be computed utilizing vector values that are being compared. Joulin et al. (2016) have implemented the fast word embedding learning algorithm coined as FastTex and published the pre-trained models in multiple languages.[6] Aligning different languages into one vector space is done by MUSE (Conneau et al., 2017). The MUSE team has also published aligned word embedding vectors trained from different languages of Wikipedia.[7] These multilingual word embeddings that have been aligned into a single vector space can be further used to calculate the similarity between words from different languages. Yet, multilingual phrases similarity (instead of words) is not directly facilitated in the MUSE pre-trained model.

We have tested the aligned multilingual pre-trained vectors from MUSE for evaluating the quality of mappings between multilingual phrases. This is done by calculating the cosine similarity between averaged vectors of each concept from each language. If a phrase has multiple links, we select the phrases pair with the maximum similarity value. However, the result of using this averaged-vector approach from multilingual phrases is not satisfactory. For example, any language pair involving German language resulted in a maximum F-Measure value of 0.153 for CPV datasets. Following observations hold when working with pre-trained embeddings for specialized cases.

1. The pre-trained models are more generic, and are not fully applicable to specialized fiscal data.
2. The training of models require substantial amounts of training data, which is not widely available for fiscal data.
3. The effectiveness of word embeddings usage depends on the language. For example, in our experience, a word in German consists of several conjugated words that are not available in the publicly-available Wikipedia-based pre-trained word embedding vector index. Hence, it results in a word vector that can not be found in the vector index.
4. The published pre-trained word embedding vectors are, as the name suggests, based on words instead of phrases. Its application to the cross-lingual phrase similarity requires n-gram training from each respective language corpora. However, such data is not available in the fiscal domain.

4.2.2. Entity linking

Concept mappings can be done when concepts are represented as entities within knowledge bases. Pappu et al. (2017) performed a lightweight multilingual entity extraction and linking using an approach they coined as Fast Entity Linker (FEL). The FEL approach detects mentions and retrieves entities, utilizing compact entity embedding that captures and searches several features used for entity disambiguation (e.g., click logs). Graph algorithms and context-based retrieval on structured knowledge bases can also be utilized in detecting correct entities for multilingual settings. Moussallem et al. (2017) presents a multilingual, knowledge-base agnostic and deterministic entity linking approach (coined as Multilingual AGDISTIS or MAG) which combines context-based retrieval and graph algorithms on structured knowledge bases. MAG does not require mono-lingual models. In another work, labels surrounding a graph entity can also be used to find a matching entity from another language with the help of machine translation. Such work is done by Lesnikova (2016), in which context found in an RDF graph is used to find links between similar entities from different languages. This is done by creating virtual documents from the labels found in the neighboring nodes of compared RDF entities. Virtual documents are then translated to a similar language and then compared with string similarity measures. All the approaches summarized above require the data and the

[6] https://fasttext.cc/docs/en/crawl-vectors.html
[7] https://github.com/facebookresearch/MUSE/

context and/or published with additional information in RDF format. These approaches are however not applicable to budget and spending datasets. Budget and spending classification data tends to be published as a spreadsheet in tabular form. Classification concepts in fiscal data tend to be provided in short phrases, i.e., not provided as entities on a knowledge graph with surrounding labels and properties. Hence, attempts to interlink fiscal data concepts as done by other previous works by Pappu et al. (2017); Moussallem et al. (2017); Lesnikova (2016) are not feasible due to the lack of entity/semantics surrounding published fiscal concepts. We consequently choose to investigate the use of string similarity measures instead to create a mapping between translated open fiscal data concepts.

4.2.3. Ontology based data integration

According to Wache et al. (2001), Ontology-Based Data Integration (OBDI) refers to the use of ontologies to capture implicit knowledge from different data sources and obtain the semantic interoperability from these heterogeneous sources. Wache et al. (2001) also states that ontology can be used to integrate data with several approaches: (1) Single ontology - requires an ontology to integrate data. (2) Multiple ontologies - requires mapping of concepts across used ontologies. (3) Hybrid - uses one ontology as a base for underlying multiple ontologies used in data integration. In fiscal data context, it may be possible to integrate all these datasets multilingually using OBDI if the following is available: (a) Ontologies that able to represent different types of fiscal classifications. (b) An approach to mapping the different types of fiscal classifications into concept instantiation/assertions (i.e., ABox) based the specific ontologies. (c) A method to handle multilingualism during assertion mappings.

Ontology-based data integration for heterogeneous datasets requires a well-defined ontology for our use case, i.e., open fiscal data. Specifically for the main open fiscal data itself,[8] OpenBudgets.eu (OBEU) ontology has been developed by Dudaš et al. (2015). The OBEU ontology is based on data cube vocabulary[9] for statistical data. The ontology covers how money allocated/spent for budget/spending are represented (i.e., measure), how attributes (e.g., the currency) can be represented, and how dimensions (i.e., classifications) can be modelled. However, a specific ontology[10] for representing concepts from different open fiscal data classifications are not yet available.

Instead of publishing open fiscal data in the RDF format that adopts ontologies, open fiscal data is published in a tabular format without semantics. The dataset is accompanied by controlled vocabularies in the form of classifications. These classifications are mostly independently published by local/national public administrators. Despite the availability of standardized classifications published by interstate organizations, these classifications are not adapted by local/national public administrators (Musyaffa et al., 2018a), since adapting these vocabularies requires alignment of concepts from their business process flow. This alignment requires efforts, resources, and an approach to handle fiscal data complexity. As we know, fiscal classification deal with concrete controlled concepts, while ontologies deal more in an abstract and formal level, hence it requires more expertise to apply or even develop ontologies for this domain. This demonstrates that the creation of ontologies is not feasible for now. Moreover, since embedding semantics on publishing fiscal data requires a steep learning curve, these ontologies may not be used by the data publishers. This is because the development and application of these ontologies require training, substantial efforts, and resources which are often neither feasible activities nor a priority for these public administrations.

4.3. Data interlinking frameworks

SILK Framework (Volz et al., 2009) is a data interlinking framework which, among others, consists of different string similarity measure implementations. Some common uses of SILK Framework include (1) link generation among data items across different sources of linked data, and (2) data transformation of structured data. SILK Framework provides a visual user interface to build a data interlinking pipeline. The framework also allows the query of data items from SPARQL endpoints, as well as obtain data items from structured

[8] i.e., the datasets with stated budgeted amount

[9] https://www.w3.org/TR/vocab-data-cube/

[10] in the sense that available classes, axioms, and properties that correlates the terms within the ontology are formally defined.

data formats, such as CSV. String similarity measures are also implemented within SILK Framework as plugins, which ranges from sequence-based and set-based string similarity measures. A distance threshold can be specified in the string similarity experiment on SILK Framework. Using SILK, an experiment has been done by Karampatakis et al. (2018), utilizing string similarity measures implementation to map links between Central Product Classification (CPC, published by the United Nations) and Classifications of Products by Activity (CPA, published by the European Union and derived from CPC). Four similarity measures are used in their experiment namely: Dice, Jaro, Jaro-Winkler, and Soft Jaccard, Jaro provides the best precision value when the threshold (i.e., distance) is set at 0.0, and Dice provides the best recall with a threshold set at 0.5. Depending on the similarity measures and distance threshold configuration, the usage of SILK Framework may impose an out-of-memory problem as we experienced with our initial experiment (will be elaborated in subsubsection 6.2.2). The fiscal classification concepts may result in a large number of comparisons that require scalability, which is not covered by SILK Framework.

4.4. Open Data Related Platforms and Tools

The proposed work on mapping and interlinking open datasets can improve the existing open data platforms such as OpenSpending, Socrata,[11] Junar,[12] OpenGov,[13] and WikiBudgets.[14] OpenSpending provides a free platform to store, visualize and analyze open fiscal datasets. Fiscal data in OpenSpending is stored as a CSV file, accompanied with JSON metadata which is created via annotation done by the uploader. The combination of this CSV file and the accompanying JSON metadata follows a specification for publishing open fiscal data as Fiscal Data Package[15] (FDP). Socrata is a Data as a Service platform that allows data integration and analytics. It also supports a wide variety of services, such as governmental program and public engagement optimization, trends and governmental insight discovery, as well as open data-related services, including open budget. Wikibudgets is a platform to interactively visualize budget data from public administrations in an easy to understand manner. Junar is an open data platform to make open data easier to use for citizens, developers, and companies. Junar provides features such as visualizations, dashboards, maps, and dynamic tables. Upon the inspection of the websites of mentioned platforms, we could not find any platform listed that provides a multilingual mapping across open data. For this reason, we deem that it is important to investigate further regarding concept mapping approaches for open fiscal data domain.

5. Approach

The architecture overview of IOTA is shown in Figure 1. Given any two fiscal datasets, their labels or concepts are identified according to the provided classifications. These classifications act as blocks for comparisons and the labels belonging to different classifications are not compared, avoiding any unnecessary comparisons and optimizing the performance. Since the labels are provided in the regional languages, an essential step is to translate the concepts based on the classification pairs. We used Google Translate[16] for the translation of concepts. These translated concepts are then post-processed for case correction and stored in HDFS. The string matching module is executed within Apache Spark (Figure 3), and this module reads the data for parallel string matching from the HDFS. The parallel string similarity assessment in IOTA is achieved by using *py_stringmatching* library (*py_stringmatching* Documentation, 2016; Doan et al., 2012), and the parameters used in IOTA are detailed in Table 5. IOTA utilizes 19 generic string similarity assessment algorithms taken from five different similarity measure categories. The inspected similarity measures in this experiment are shortly described in Table 2, Table 3, Table 4, for sequence-based, set and hybrid-based, as well as bag-based and phonetic-based, respectively.

[11] https://socrata.com
[12] http://www.junar.com/
[13] https://opengov.com/
[14] https://www.wikibudgets.org/
[15] https://frictionlessdata.io/specs/fiscal-data-package/
[16] https://translate.google.com/

Another perspective of Figure 1 can be seen in Figure 2, explaining parameters used within IOTA Framework. Open fiscal data comes with respective classifications, which ideally should be aligned before integrating the data. However, this is not the case. For linking the similar concepts, we use and benchmark several similarity measures after string preprocessing and machine translation steps. A minimum threshold of γ is set to filter the overall similarity values that are allowed to be inspected as links. To investigate the optimum similarity threshold values, we set an iterative threshold $t(i)$ where $\gamma \leq t(i) \leq 1$. The filtered result is then analyzed and evaluated, which are then interlinked via the RDF SKOS ontology for interlinking similar multilingual concepts.

The translated and processed set of concepts stored in HDFS for the two data sets are represented internally as RDDs as shown in Figure 3. The RDD is a parallel collection of records that can be processed in a distributed, parallel manner to achieve scalability. The entities presented in the two RDDs are then cross-compared with each other for the similarity assessment. For concept matching, we use a regular expression before applying the selected similarity assessment to extract the exact location of the labels of interest from each label. The output of this parallel operation is an RDD with the entity pairs and their similarity score. The final step is to filter out the scores in the distributed RDD, based on the provided threshold. The filtered result is stored back to HDFS, which is used for evaluating the performance. The use of classification based translation, matching and parallel in-memory processing differentiates IOTA from other state of the art string comparison frameworks.

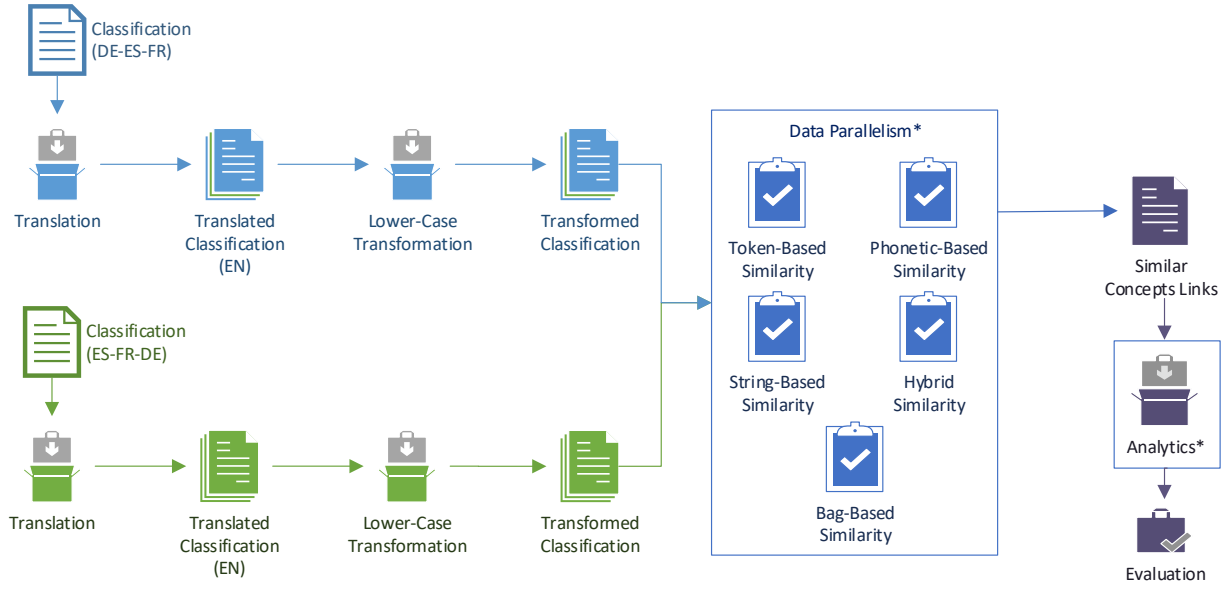

Figure 1: Our IOTA pipeline to map similar concept from translated classification. Preprocessed, translated classifications from different language and public administration are measured for their similarity scores. *The similarity measure comparison and analytics process utilizes Apache Spark for scalability.

6. Experiment and evaluation

6.1. Dataset and evaluation metrics

For the experiment, we use the European Union official procurement classification,CPV classification (European Commission, 2008). CPV is published in 24 different European languages. This dataset is comprised of 9454 concepts. Each concept in any language is associated with a unique key. Hence, the key can be used to identify a proper match between concepts.

The experiment starts with translating concepts from different languages using Google Translate. Three datasets originally from German, Spanish and French are translated into English. The translation result is

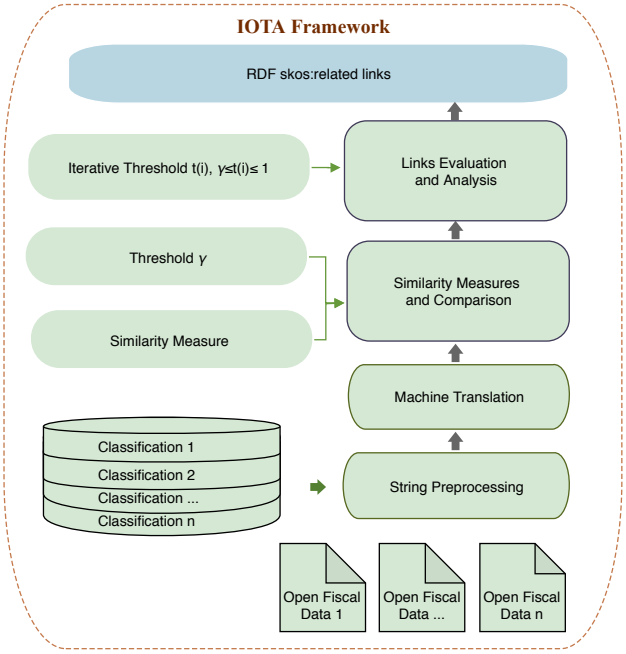

Figure 2: IOTA Framework takes out classification labels from different languages, as well as specific similarity measures and minimum threshold that can limit the similar string estimation. Later, we iterate from the minimum passing similarity threshold from γ to 1, to check which thresholds yield the highest F-Measure.

Type	Similarity measure	Norm. range	Score ranging from [0-1] by default?	Non-norm. Score Range	Extra param. required?	Type of extra parameter	Default value (respectively)
Bag-based	TF/IDF	[0,1]	Yes	-	Yes	Corpus (list containing lists), Dampening (true/false)	None (only use tokens from compared strings), True
Hybrid-based	Generalized Jaccard	[0,1]	Yes	-	Yes	Similarity measure, Similarity threshold	Jaro, 0.5
	Monge-Elkan	[0,1]	Yes	-	Yes	Similarity measure	Jaro-Winkler
	Soft TF/IDF	[0,1]	Yes	-	Yes	Corpus (list containing lists), Similarity measure, Similarity threshold	None (only use tokens from compared strings), Jaro, 0.5
Phonetic-based	Soundex	[0/1]	Yes	-	No	-	-
	Editex	[0,1]	No	Integer	Yes	Match cost (weight when the correct char match), Group cost (weight when the char is in the same Editex group), Mismatch cost (weight when the char match incorrect), Local variant	0, 1, 2, False
Sequence-based	Bag distance	[0,1]	No	Integer	No	-	-
	Jaro*	[0,1]	Yes	-	No	-	-
	Jaro-winkler*	[0,1]	Yes	-	Yes	Prefix weight (weight for the prefix)	0.1
	Levenshtein*	[0,1]	No	Integer	No	-	-
	Partial Ratio	[0,1]	No	[0, 100]	No	-	-
	Ratio	[0,1]	No	[0, 100]	No	-	-
	Partial token sort	[0,1]	No	[0, 100]	Yes	Force ASCII (boolean to remove non-ASCII characters), Full process (boolean for preprocessing such as lower case transformation as well as removing leading/trailing white spaces)	True, True
	Token Sort	[0,1]	No	[0, 100]	Yes	Force ASCII (boolean to remove non-ASCII characters), Full process (boolean for preprocessing such as lower case transformation as well as removing leading/trailing white spaces)	True, True
Set-based	Cosine	[0,1]	Yes	-	No	-	-
	Dice	[0,1]	Yes	-	No	-	-
	Jaccard	[0,1]	Yes	-	No	-	-
	Overlap Coefficient	[0,1]	Yes	-	No	-	-
	Tversky Index	[0,1]	Yes	-	No	-	-

Table 5: The list of different similarity measures used within the IOTA framework. Similarity measures marked with asterisks (*) indicate a cythonized implementation in the *py_stringmatching* library that speeds up the performance. The similarity score range from 0 to 1 for most similarity scores, except for Soundex similarity, which provides a true or false decision. Most of our experiments use default parameter values provided by the library, except for TF-IDF and Soft TF-IDF, in which we are using a corpus from the whole translated words instead of only compared, translated words.

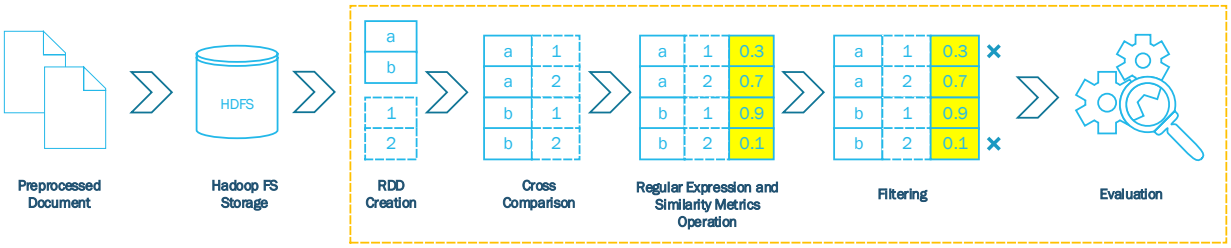

Figure 3: Distributed processing pipeline that we perform in our IOTA experiment. Preprocessed classification documents are stored within Hadoop FS, then Apache-Spark operations follow the next step: creating RDD data types out of stored documents, performing cross computation, getting similarity score between concepts, filtering the scores and finished by evaluating the result.

then paired in three language pairs as shown in Table 6: German-Spanish, German-French, and Spanish-French.

Language Pairs	
German	Spanish
Spanish	French
French	German

Table 6: Language pairs used for our experiment. The pairs are chosen based on the availability in the datasets (Common Procurement Vocabulary by European Union) and how wide the EU languages are used.

The mappings between the two classifications are evaluated using recall, precision, and F-measure. To compute these measures, we calculate true positive, false positive, false negative and true negative values by comparing the assigned and original values.

The basic combinations of actual vs assigned data category is a well-known concept in binary classification. *True positive* (tp) indicates the number of retrieved information that classified as correct and actually belongs to the correct result. In our case, the possible number of true positives is equal to the number of concepts in the datasets we are experimenting with. *False positive* (fp) indicates the number of the incorrect result but are classified as a correct result. *False negative* (fn) indicates information that is classified as false but it is not actually false. False negative is computed based on the possible number of true positive links minus true positive links that are found, so

$$fn = |concept| - tp.$$

True negatives (tn) are the number of classes that are classified as false and are actually false. True negative is a result of the subtraction of Cartesian product cardinality between two sets in the compared concepts with the sum of true positives, false positives, and false negatives, so

$$tn = (|concept_1| \times |concept_2|) - (tp + fp + fn).$$

F-Measure is then calculated as a harmonic mean of precision and recall.

$$F - Measure = 2 * \frac{precision * recall}{precision + recall}$$

In this paper, we attempt to answe rthe following four research questions ($\mathcal{RQ}$):

- $\mathcal{RQ}1$. Which string similarity measures provide the highest F-Measure in interlinking fiscal classification concepts?
- $\mathcal{RQ}2$. What is the impact of applying a similarity threshold for interlinking concepts between translated classification?

- $\mathcal{RQ}3$. How robust is the similarity measure performance when the language pairs are changed?
- $\mathcal{RQ}4$. Different similarity measures have different computational performance. Which similarity measures have faster computational performance, and is there any trade-off between faster computational performance and the resulting F-Measure?

6.2. Experimental configuration and result

6.2.1. SILK framework experiment configuration

In the initial experiment, we use German and Spanish concept from CPV classification. Computing similarity between translated string is done at first by utilizing SILK Framework which implements sequence-based similarity matching (*Jaro*, *Jaro Winkler*, *Levenshtein*, *Normalized Levenshtein*, *qGrams* and *Substring*) as well as set-based similarity matching (*Token-wise*, *Soft-Jaccard*, *Dice* and *Jaccard*). Other than performing tokenization for set-based similarity measure and changing the distance threshold, we use default parameters in SILK Framework. Comparison of strings leads to a *distance* threshold which is defined as the maximum distance two strings allowed to have. The more distance threshold value is set, the more links can be found, but there are more false-positive links discovered. The experiment result is then stored as an ontology alignment XML format[17], which later converted to CSV and then processed for analytics using a python script. We use the latest stable version of SILK Framework v2.7.1, at the time of our experiment.

6.2.2. SILK framework experiment result

From our experiment using the SILK Framework, the similarity measure that yields the biggest F-Measure score is Substring, with 0.501 F-Measure scores as the distance threshold is set to 0.2. In our particular use case, other similarity measures that provide a relatively good F-Measure score are qGrams (F-Measure = 0.453, distance threshold = 0.4) and Soft Jaccard (F-Measure = 0.446, distance threshold = 0.4). The result of the experiment using SILK framework is provided in detail on Table 7. The corresponding F-Measure chart for the SILK experiment is provided in Figure 4.

On low distance thresholds (i.e. 0.0), it is fast to use SILK Framework for most similarity measures, except for some similarity measures that are failed at the 0.0 distance threshold. On Table 7, the fields that are marked as an asterisk (*) in the table indicate that those fields yield out of memory error during the experiment, hence an increase in the higher distance threshold can not be done. On the other hand, there is a problem with Cosine similarity measure during our experiment, and we can not proceed with any of the thresholds, which is indicated with dash (-) in Table 7. This limitation prevents our further experiment using higher distance thresholds. As a result, in similarity experiments using SILK Framework, only limited thresholds can be presented. For this reason, we deviated from using SILK Framework for further experiments and continue with our IOTA framework for the experiment as we described in section 5.

6.2.3. IOTA Framework Evaluation Configuration

The evaluation is conducted on a cluster of three workstations, each consists of 256 GB RAM, and each has four AMD Optheron^{TM} 6376 2.3 GHz processors. Each processor has 16 cores, totaling 64 cores in each workstation. One workstation is used as a Spark driver, and two others are used as Spark workers. We use Apache Spark 2.3.1 on our cluster during our experiment.

6.2.4. IOTA Framework Experiment Result

IOTA Framework provides several experiment results. Execution time for each language pair in the cluster is compared in Figure 5. We present the result of F-Measure values from each language pairs experiment in Table 8, Table 9, and Table 10, respectively The more intense the color of the cell within those tables, the higher the F-Measure values are. The summarized top-10 F-Measure score for each similarity measure and the filter is summarized in Table 11. The charts for these F-Measure scores from Table 8 to Table 10 are provided in Figure 7, Figure 8, and Figure 9. The radar chart of aggregated average score

[17] http://alignapi.gforge.inria.fr/format.html

Similarity Type	Similarity Measure	Distance Threshold	Found	TP	FP	FN	TN	Precision	Recall	F-Measure
Set-based	Dice	0.0	2,207	1,995	212	7,459	89,368,450	0.904	0.211	0.342
		0.2	3,952	2,671	1,281	6,783	89,367,381	0.676	0.283	0.398
		0.4	31,543	4,315	27,228	5,139	89,341,434	0.137	0.456	0.211
		0.6	417,618	6,531	411,087	2,923	88,957,575	0.016	0.691	0.031
	Jaccard	0.0	2,207	1,995	212	7,459	89,368,450	0.904	0.211	0.342
		0.2	2,372	2,114	258	7,340	89,368,404	0.891	0.224	0.358
		0.4	5,479	3,017	2,462	6,437	89,366,200	0.551	0.319	0.404
		0.6	46,383	4,597	41,786	4,857	89,326,876	0.099	0.486	0.165
	Soft Jaccard	0.0	2,844	2,432	412	7,022	89,368,250	0.855	0.257	0.396
		0.2	3,179	2,658	521	6,796	89,368,141	0.836	0.281	0.421
		0.4	7,237	3,722	3,515	5,732	89,365,147	0.514	0.394	0.446
		0.6	63,112	5,545	57,567	3,909	89,311,095	0.088	0.587	0.153
	Token Wise	0.0	*	*	*	*	*	*	*	*
	Cosine	0.0	-	-	-	-	-	-	-	-
Sequence-based	Jaro	0.0	2,179	1,968	211	7,486	89,368,451	0.903	0.208	0.338
		0.2	47,324	4,324	43,000	5,130	89,325,662	0.091	0.457	0.152
		0.4	*	*	*	*	*	*	*	*
	Jaro Winkler	0.0	2,179	1,968	211	7,486	89,368,451	0.903	0.208	0.338
		0.2	255,495	5,376	250,119	4,078	89,118,543	0.021	0.569	0.041
		0.4	*	*	*	*	*	*	*	*
	Levenshtein	0.0	2,179	1,968	211	7,486	89,368,451	0.903	0.208	0.338
		0.2	2,179	1,968	211	7,486	89,368,451	0.903	0.208	0.338
		0.4	2,179	1,968	211	7,486	89,368,451	0.903	0.208	0.338
		0.6	2,179	1,968	211	7,486	89,368,451	0.903	0.208	0.338
	Normalized Levenshtein	0.0	2,179	1,968	211	7,486	89,368,451	0.903	0.208	0.338
		0.2	4,922	3,066	1,856	6,388	89,366,806	0.623	0.324	0.427
		0.4	63,124	4,674	58,450	4,780	89,310,212	0.074	0.494	0.129
		0.6	*	*	*	*	*	*	*	*
	qGrams	0.0	2,192	1,981	211	7,473	89,368,451	0.904	0.210	0.340
		0.2	3,001	2,635	366	6,819	89,368,296	0.878	0.279	0.423
		0.4	9,836	4,365	5,471	5,089	89,363,191	0.444	0.462	0.453
		0.6	136,028	6,500	129,528	2,954	89,239,134	0.048	0.688	0.089
	Substring	0.0	2,262	2,043	219	7,411	89,368,443	0.903	0.216	0.349
		0.2	7,907	4,347	3,560	5,107	89,365,102	0.550	0.460	**0.501**
		0.4	35,881	5,791	30,090	3,663	89,338,572	0.161	0.613	0.255
		0.6	150,071	6,790	143,281	2,664	89,225,381	0.045	0.718	0.085

Table 7: Different similarity measures performance for mapping concepts originally from German and Spanish datasets using SILK Framework. Asterisk (*) sign indicates out of memory error, hence these algorithms are not scalable, while dash (-) indicates other error during the experiment. Several similarity measures here are not robust to the change of distance thresholds.

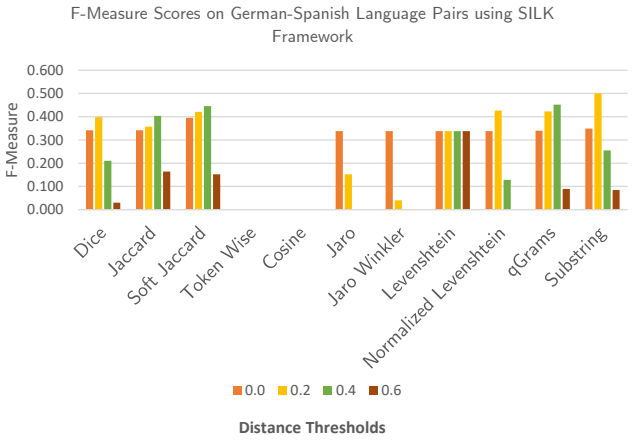

Figure 4: F-Measure chart of different similarity measures and *distance* thresholds experimented using SILK Framework. A blank space in the diagram indicates the unavailability of the F-Measure value for that particular similarity measure/filter mostly due to scalability reasons. In this comparison, *Substring* yields the highest F-Measure score, followed by *qGrams* and *Jaccard*.

is provided in Figure 10, and broken down into (1) bag-based, hybrid-based, and phonetic-based similarity measures; (2) sequence-based similarity measures; (3) set based similarity measures. These figures and tables are used to highlight the findings from our experiment.

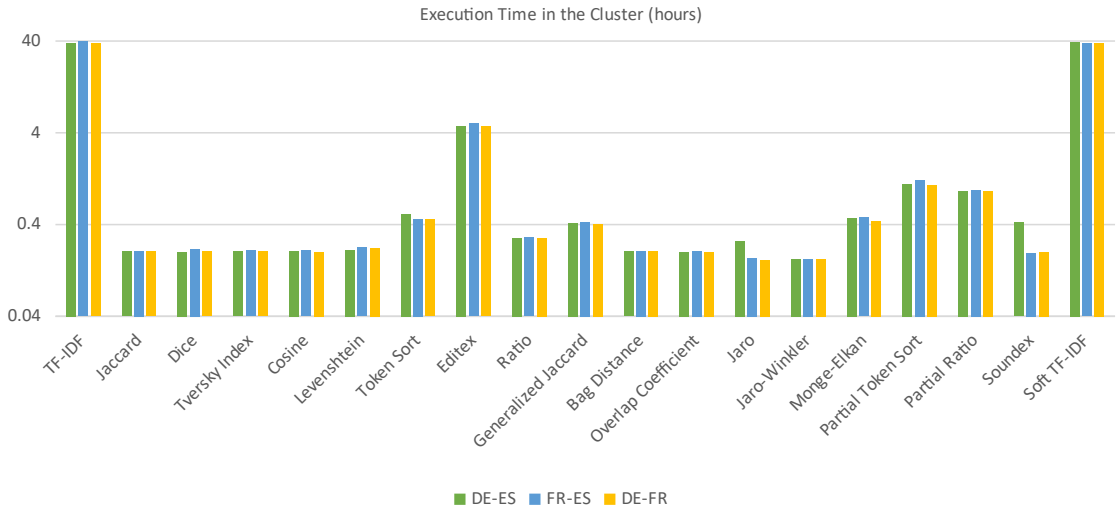

Figure 5: Execution time (hours, on a logarithmic scale) in our cluster. The cluster performs more than 89 million string comparisons. TF-IDF and Soft TF-IDF similarity measure have the longest execution time due to their complexity. Most of the other similarity measures provide decent computational performance.

6.3. Discussion

Our experiments with IOTA framework show that finding similar concept is reliable and scalable for all threshold, even though some similarity measures used within IOTA need more time for the computation. Token Sort provides the IOTA framework the highest F-Measure score when the similarity threshold is estimated properly. TF-IDF provides a high average F-Measure score which is robust across similarity threshold change, yet TF-IDF needs significant computational resources which we discuss in detail on the following subsections.

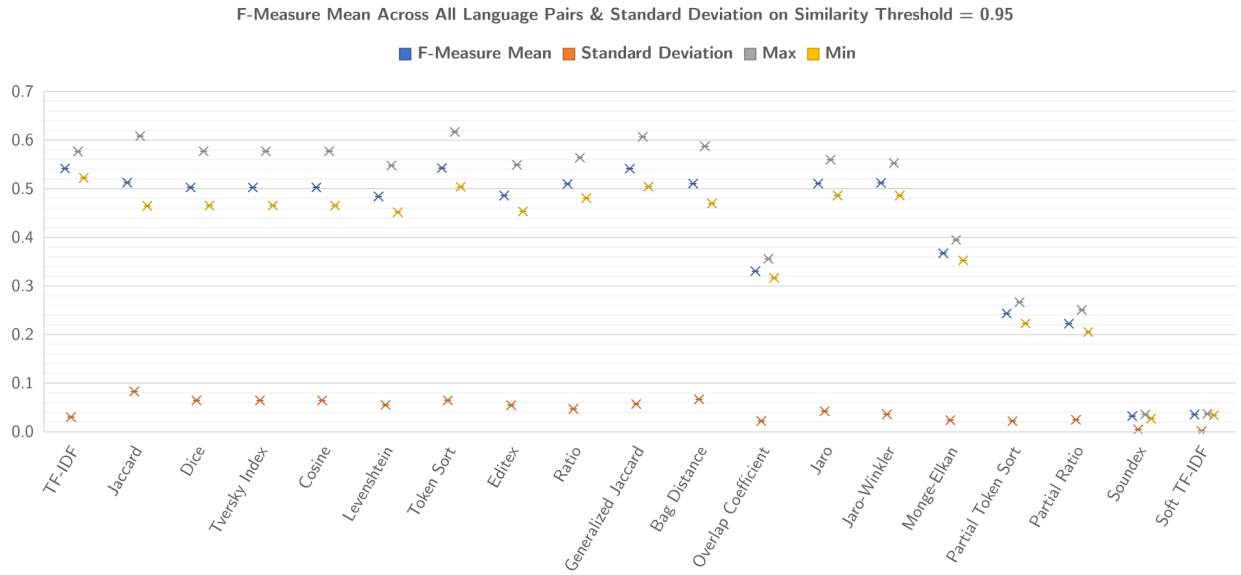

Figure 6: The plot for average F-Measure score, minimum F-measure score, maximum F-measure score and sample standard deviation for each language as similarity threshold set to 0.95. Even though the TF-IDF similarity score takes a long time to compute, it has the minimum standard deviation with a relatively good F-Measure score compared to other similarity measures. On the other hand, Token Sort yields the maximum F-Measure and needs much less computational time compared to TF-IDF, but it has a high standard deviation.

German - Spanish											
Similarity Threshold	**0.50**	**0.55**	**0.60**	**0.65**	**0.70**	**0.75**	**0.80**	**0.85**	**0.90**	**0.95**	**1.00**
TF-IDF	0.280	0.337	0.388	0.437	0.479	0.511	0.532	0.543	0.540	0.522	0.455
Jaccard	0.241	0.364	0.424	0.487	0.496	0.501	0.494	0.476	0.466	0.465	0.465
Dice	0.067	0.152	0.191	0.241	0.363	0.424	0.487	0.501	0.482	0.465	0.465
Tversky Index	0.067	0.152	0.191	0.241	0.363	0.424	0.487	0.501	0.482	0.465	0.465
Cosine	0.092	0.145	0.205	0.240	0.335	0.422	0.498	0.501	0.482	0.465	0.457
Levenshtein	0.040	0.075	0.122	0.202	0.292	0.391	0.466	0.493	0.486	0.452	0.441
Token Sort	0.011	0.023	0.048	0.093	0.178	0.311	0.460	0.545	**0.548**	0.507	0.459
Editex	0.014	0.031	0.059	0.109	0.183	0.285	0.409	0.472	0.482	0.453	0.441
Ratio	0.012	0.023	0.043	0.077	0.143	0.244	0.376	0.478	0.510	0.485	0.441
Generalized Jaccard	0.003	0.004	0.009	0.021	0.047	0.100	0.219	0.377	0.482	0.513	0.465
Bag Distance	0.001	0.001	0.002	0.004	0.011	0.036	0.148	0.415	0.503	0.474	0.455
Overlap Coefficient	0.015	0.078	0.079	0.092	0.182	0.191	0.252	0.315	0.317	0.317	0.317
Jaro	0.000	0.000	0.001	0.004	0.019	0.058	0.133	0.296	0.463	0.487	0.441
Jaro-Winkler	0.000	0.000	0.001	0.003	0.010	0.023	0.035	0.070	0.249	0.486	0.441
Monge-Elkan	0.000	0.000	0.001	0.001	0.003	0.008	0.021	0.061	0.171	0.352	0.360
Partial Token Sort	0.001	0.003	0.006	0.011	0.022	0.036	0.078	0.164	0.221	0.223	0.216
Partial Ratio	0.001	0.003	0.006	0.011	0.021	0.032	0.068	0.141	0.195	0.205	0.202
Soundex	0.027	0.027	0.027	0.027	0.027	0.027	0.027	0.027	0.027	0.027	0.027
Soft TF-IDF	0.001	0.001	0.002	0.002	0.003	0.005	0.008	0.013	0.021	0.034	0.042

Table 8: F-Measure values of different string similarity measures for mapping concepts originally from German and Spanish datasets. TF-IDF, Jaccard, and Dice have the best F-Measure scores when it is averaged by the similarity thresholds. Token-Sort provides the best F-Measure score as the similarity threshold is set to 0.90.

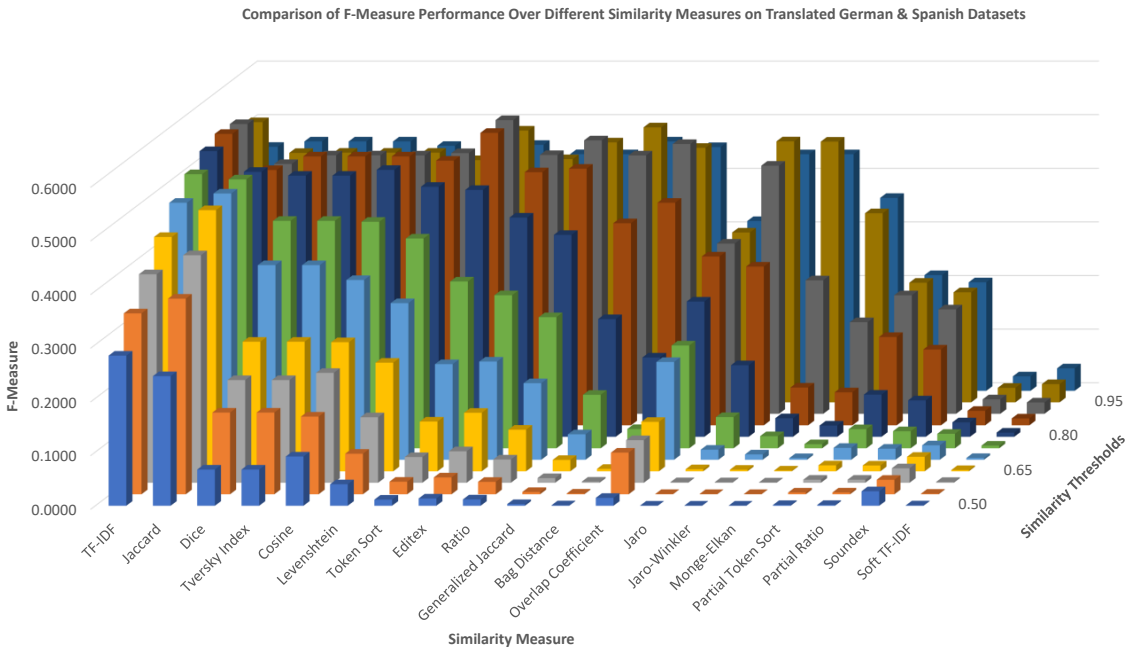

Figure 7: F-Measure chart of different similarity measures and filters for matching strings between translated German and Spanish datasets, as shown in Table 8. The performance reaches a peak as the similarity threshold is set to 0.90.

German - French											
Similarity Threshold	**0.5**	**0.55**	**0.6**	**0.65**	**0.7**	**0.75**	**0.8**	**0.85**	**0.9**	**0.95**	**1**
TF-IDF	0.283	0.340	0.393	0.442	0.485	0.517	0.537	**0.548**	0.544	0.526	0.456
Jaccard	0.282	0.413	0.468	0.513	0.511	0.512	0.497	0.478	0.466	0.465	0.465
Dice	0.073	0.178	0.234	0.282	0.412	0.468	0.513	0.512	0.484	0.466	0.465
Tversky Index	0.073	0.178	0.234	0.282	0.412	0.468	0.513	0.512	0.484	0.466	0.465
Cosine	0.106	0.169	0.242	0.281	0.371	0.466	0.515	0.512	0.485	0.466	0.456
Levenshtein	0.042	0.080	0.131	0.220	0.311	0.404	0.470	0.492	0.482	0.453	0.445
Token Sort	0.011	0.024	0.049	0.095	0.182	0.319	0.466	**0.548**	0.544	0.504	0.459
Editex	0.013	0.032	0.061	0.115	0.194	0.298	0.418	0.475	0.480	0.455	0.445
Ratio	0.012	0.023	0.045	0.081	0.152	0.263	0.395	0.486	0.502	0.481	0.445
Generalized Jaccard	0.003	0.004	0.009	0.021	0.048	0.102	0.229	0.394	0.486	0.504	0.465
Bag Distance	0.001	0.001	0.002	0.005	0.011	0.037	0.155	0.425	0.503	0.470	0.454
Overlap Coefficient	0.015	0.091	0.093	0.104	0.215	0.224	0.268	0.319	0.319	0.319	0.319
Jaro	0.000	0.000	0.001	0.004	0.020	0.073	0.170	0.345	0.483	0.486	0.445
Jaro-Winkler	0.000	0.000	0.001	0.003	0.011	0.027	0.046	0.100	0.309	0.497	0.445
Monge-Elkan	0.000	0.000	0.001	0.001	0.003	0.008	0.022	0.067	0.187	0.355	0.363
Partial Token Sort	0.001	0.003	0.006	0.011	0.022	0.034	0.080	0.184	0.260	0.266	0.261
Partial Ratio	0.001	0.003	0.005	0.010	0.019	0.030	0.067	0.154	0.232	0.251	0.249
Soundex	0.034	0.034	0.034	0.034	0.034	0.034	0.034	0.034	0.034	0.034	0.034
Soft TF-IDF	0.001	0.001	0.002	0.002	0.003	0.005	0.008	0.013	0.022	0.035	0.043

Table 9: F-Measure values of different string similarity measures for mapping concepts originally from German and French datasets. Token Sort remains the similarity measure that yields the highest F-Measure, although the optimum similarity threshold is 0.85 instead instead of 0.90.

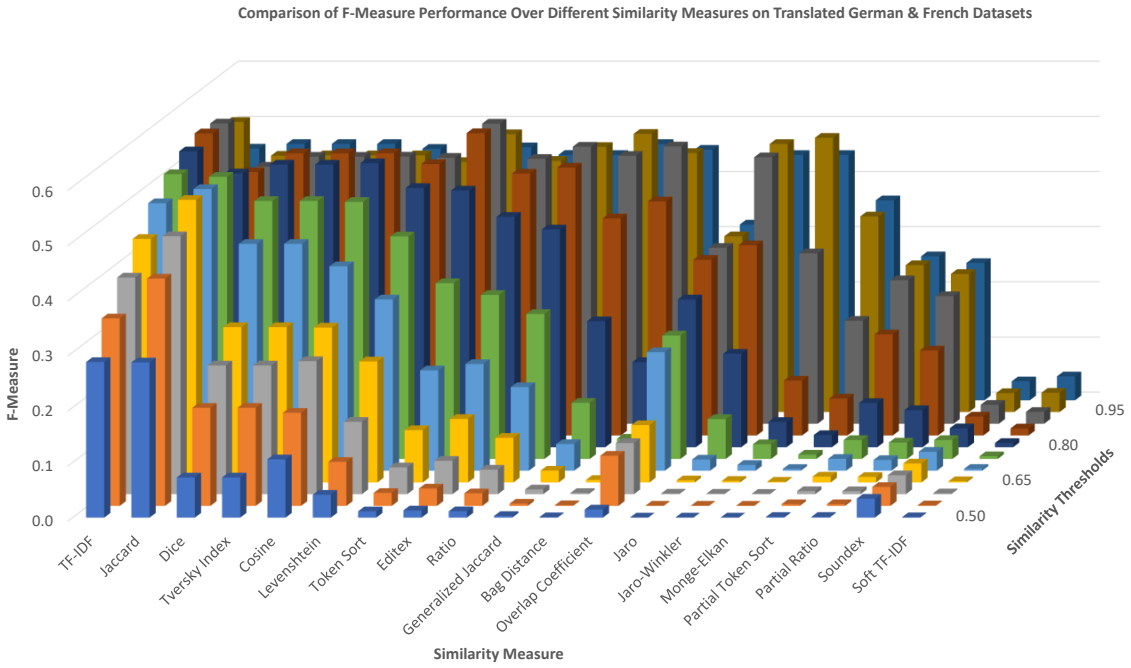

Figure 8: F-Measure chart of different similarity measures and filters for matching strings between translated German and French datasets, as shown in Table 9. There is no significant difference as compared to the chart from German-Spanish dataset (Figure 7).

French - Spanish											
Similarity Threshold	**0.50**	**0.55**	**0.60**	**0.65**	**0.70**	**0.75**	**0.80**	**0.85**	**0.90**	**0.95**	**1.00**
TF-IDF	0.297	0.360	0.420	0.474	0.523	0.566	0.598	0.615	0.618	0.608	0.564
Jaccard	0.227	0.396	0.452	0.556	0.595	0.605	0.605	0.587	0.578	0.577	0.577
Dice	0.055	0.135	0.178	0.227	0.395	0.452	0.556	0.605	0.592	0.577	0.577
Tversky Index	0.055	0.135	0.178	0.227	0.395	0.452	0.556	0.605	0.592	0.577	0.577
Cosine	0.080	0.128	0.192	0.226	0.356	0.450	0.568	0.605	0.593	0.577	0.562
Levenshtein	0.030	0.060	0.102	0.177	0.266	0.375	0.493	0.559	0.564	0.548	0.536
Token Sort	0.009	0.018	0.036	0.068	0.132	0.249	0.426	0.585	**0.645**	0.617	0.571
Editex	0.010	0.023	0.046	0.091	0.160	0.259	0.407	0.521	0.560	0.549	0.536
Ratio	0.009	0.017	0.032	0.061	0.117	0.214	0.357	0.501	0.571	0.564	0.536
Generalized Jaccard	0.003	0.004	0.009	0.021	0.044	0.093	0.213	0.397	0.558	0.607	0.577
Bag Distance	0.001	0.001	0.002	0.004	0.011	0.034	0.141	0.442	0.598	0.587	0.566
Overlap Coefficient	0.011	0.072	0.073	0.085	0.179	0.187	0.286	0.355	0.356	0.356	0.356
Jaro	0.000	0.000	0.001	0.003	0.018	0.069	0.154	0.321	0.520	0.560	0.536
Jaro-Winkler	0.000	0.000	0.001	0.003	0.010	0.026	0.047	0.103	0.286	0.553	0.536
Monge-Elkan	0.000	0.000	0.001	0.001	0.003	0.007	0.018	0.056	0.172	0.395	0.424
Partial Token Sort	0.001	0.003	0.005	0.009	0.018	0.030	0.068	0.151	0.227	0.240	0.234
Partial Ratio	0.001	0.003	0.005	0.008	0.016	0.025	0.053	0.114	0.189	0.211	0.208
Soundex	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036	0.036
Soft TF-IDF	0.001	0.001	0.002	0.002	0.003	0.005	0.008	0.014	0.023	0.037	0.051

Table 10: F-Measure chart of different similarity measures and filters for matching strings between translated French and Spanish datasets. The French - Spanish language pair yields the highest F-Measure (0.645) compared to previous language pairs (0.548 for both of previous language pairs). Token Sort remains the best performing algorithms when the similarity threshold is properly set.

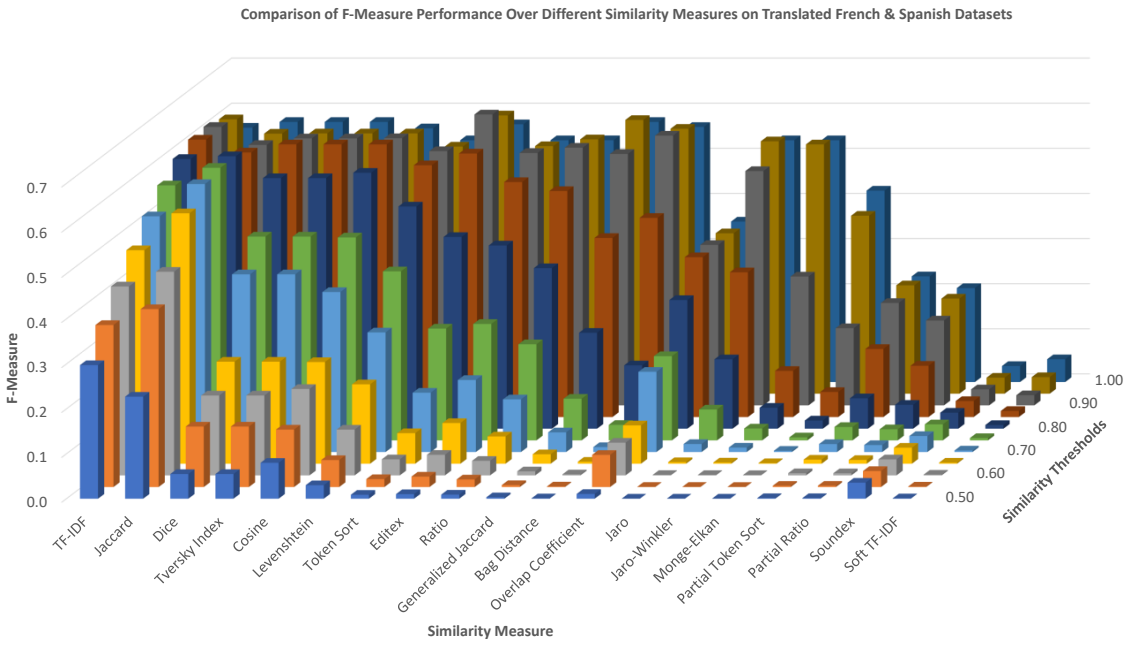

Figure 9: F-Measure chart of different similarity measures and filters for matching strings between translated French and Spanish datasets, as shown in Table 10. It consistently has similar patterns with Figure 7 and Figure 8 regarding similarity thresholds and similarity measures.

French-Spanish						German-French						German-Spanish					
Rk.	FM	Prec.	Rec.	Similarity Measure	Thres.	Rk.	FM	Prec.	Rec.	Similarity Measure	Thres.	Rk.	FM	Prec.	Rec.	Similarity Measure	Thres.
1	0.6450	0.8316	0.5269	Token Sort	0.90	1	0.5483	0.6501	0.4741	Token Sort	0.85	1	0.5476	0.8258	0.4096	Token Sort	0.90
2	0.6175	0.8763	0.4767	TF-IDF	0.90	2	0.5479	0.7684	0.4257	TF-IDF	0.85	2	0.5452	0.6425	0.4735	Token Sort	0.85
3	0.6170	0.9150	0.4654	Token Sort	0.95	3	0.5444	0.8486	0.4008	TF-IDF	0.90	3	0.5432	0.7802	0.4166	TF-IDF	0.85
4	0.6153	0.8048	0.4980	TF-IDF	0.85	4	0.5442	0.8187	0.4076	Token Sort	0.90	4	0.5402	0.8554	0.3948	TF-IDF	0.90
5	0.6084	0.9171	0.4552	TF-IDF	0.95	5	0.5366	0.6626	0.4508	TF-IDF	0.80	5	0.5325	0.6684	0.4425	TF-IDF	0.80

Table 11: Top five F-Measure (FM), Precision (Prec.), Recall (Rec.) scores and their corresponding similarity measure and thresholds (Thresh.). Token Sort and TF-IDF yield the highest F-Measure scores when the similarity thresholds is set from 0.80 upwards.

6.3.1. Performance evaluation

In the discussion section, we categorize the similarity measures into three main categories: (1) bag-based, hybrid-based and phonetic-based similarity measures (2) sequence-based similarity measures and (3) set-based similarity measures. For each category, the F-Measure score is averaged by language pairs.

Hybrid similarity measures are designed to consider misspelled tokens (Doan et al., 2012). In the hybrid similarity measure category, Generalized Jaccard provides the best F-Measure but it is sensitive to similarity threshold values. In this experiment, we use default parameter values in the *py_ stringmatching* library (see Table 5), which uses Jaro as secondary string similarity measure and 0.5 as similarity threshold. The result can potentially be improved if a better performing, secondary, string similarity measure is chosen, for example, Levenshtein instead of Jaro (see Tables and Diagrams in subsubsection 6.2.4). The same tuning can also be made for Monge-Elkan and Soft TF-IDF to improve their result. The only bag-based similarity measure, TF-IDF, provides better F-Measure value than Generalized Jaccard and other hybrid-based similarity measures. TF-IDF can capture insignificant words from the set of given corpus, and use it to make the result more relevant.

Phonetic-based algorithms are intended to match similarly sounding words. Intuitively, phonetic algorithms do not fit in our particular use case for matching translated multilingual concepts. As can be seen in the result section, Soundex has low F-Measure scores across language pairs due to its binary distinction of similar concepts. Editex, despite belongs to the phonetic algorithm category, provides much higher F-Measure score as compared to Soundex for phonetic similarity measures. This might be because Editex improves the character grouping on Soundex and combines it with Levenshtein-like similarity measure (see Table 4), making the Soundex similarity measures are much less restrictive. The difference between the two are highlighted in Figure 10(a), which shows a low average F-Measure score for Soundex, and in contrast, Editex is surprisingly decent for this task.

The best F-Measure score for the sequence-based similarity measure category is provided by Token Sort, as illustrated in Figure 10(b). In our use case, this makes sense because the result of the translation may end up as similar words, but arranged in a different phrase. After removing non-ASCII characters, removing trailing white spaces, and sorting these phrases, ratio similarity measure counts the ratio between matching elements and total elements from both compared strings. In overall algorithm category, Token Sort provides the best F-Measure score across all the language pairs we have experimented with, as shown in Table 11, as well as the diagrams in Figure 7, Figure 8, and Figure 9.

In the set-based similarity measure category, many similarity measures share close results as can be seen from Figure 10(c). This may be due to the similar formulation of the set-based similarity measures, which are mostly based on the size of shared tokens. The highest average F-Measure value across three language pairs, 0.54, is shared by many set-based similarity measures, such as Cosine-Ochiai, Dice and Tversky Index, when the filter configuration is set to 0.85. The same F-Measure value (0.54) is also found on Jaccard when the filter value is set to be 0.75.

6.3.2. Similarity thresholds

Generally, the effective similarity threshold values, i.e., the filter that provides a consistently good F-Measure value could not be determined as it can be seen from Table 8, Table 9, and Table 10. The values of F-Measure score in these tables show that some similarity measures are sensitive to similarity threshold values, except, for example, TF-IDF and Jaccard which still provide a relatively high F-Measure score across different similarity thresholds values, and Soft TF-IDF which provides low F-Measure score values across similarity thresholds. Figure 6 shows that Token Sort has a high standard deviation and sensitive to similarity threshold change, but also yield the highest F-Measure on three language pairs. Token Sort is sensitive to change on similarity thresholds. It is also observed that highest filter score does not guarantee that the F-Measure will be higher, but the F-Measure score tends to be higher with the higher filter score.

6.3.3. Language pairs

There is a difference on F-Measure performance across different translated language pairs. The best language pair in our experiment is FR-ES, followed by DE-FR and at last DE-ES. The correlation across

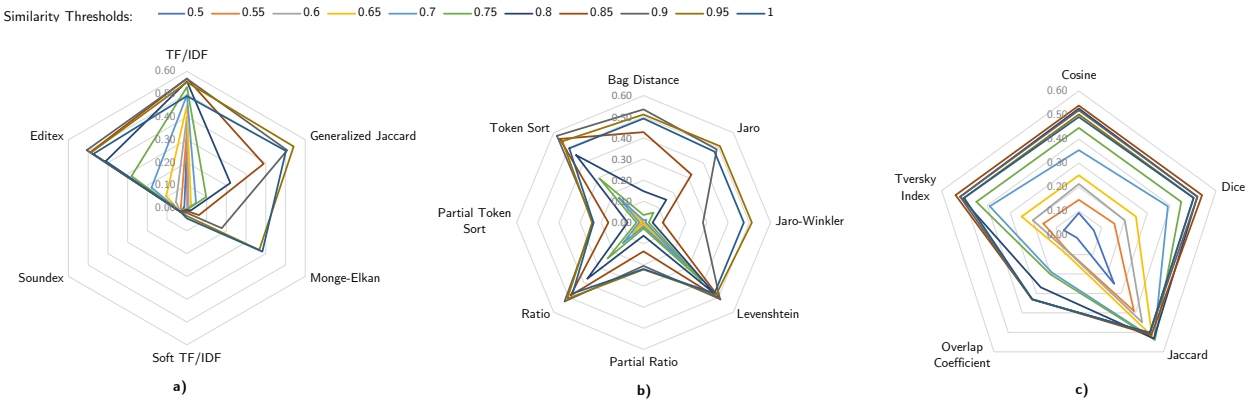

Figure 10: Average F-Measure on different similarity measures and filters across three language pairs. The further the threshold line from the center of the polygon, the more F-Measure score it has. (a) Bag-/Hybrid-/Phonetic-based similarity measures have various performance. TF-IDF provides a robust performance against most similarity thresholds while Editex and Generalized Jaccard also provide a decent performance but not robust to threshold change. (b) Sequence-based similarity measures are sensitive to threshold change, with Token Sort, provides the highest F-Measure. (c) Set-based similarity measures yield similar performance, and are also sensitive to threshold change.

Sim. Threshold: 0.85

	DE-ES	DE-FR	FR-ES
DE-ES	1.00	1.00	0.96
DE-FR	1.00	1.00	0.96
FR-ES	0.96	0.96	1.00

(a)

Sim. Threshold: 0.9

	DE-ES	DE-FR	FR-ES
DE-ES	1.00	0.92	0.90
DE-FR	0.92	1.00	0.91
FR-ES	0.90	0.91	1.00

(b)

Sim. Threshold: 0.95

	DE-ES	DE-FR	FR-ES
DE-ES	1.00	1.00	0.88
DE-FR	1.00	1.00	0.88
FR-ES	0.88	0.88	1.00

(c)

Sim. Threshold: 1

	DE-ES	DE-FR	FR-ES
DE-ES	1.00	1.00	0.98
DE-FR	1.00	1.00	0.98
FR-ES	0.98	0.98	1.00

(d)

Figure 11: Spearman Correlation between different language pairs for different thresholds: a) 0.85, b) 0.90, c), 0.95, d) 1.00. Each language pairs are positively and strongly correlated to each other, with the lowest value of correlation score 0.882 between German-French and French-Spanish when the similarity threshold is set to 0.95.

different language pairs are very high, as illustrated in four different similarity threshold in Figure 11(a), Figure 11(b), Figure 11(c), and Figure 11(d) for similarity threshold = 0.85, 0.90, 0.95 and 1.00 respectively. The lowest Spearman correlation value between all the language pairs is between German-French and French-Spanish with the value of 0.882 when the similarity threshold is set to 0.95. This indicates that regardless of language pairs, using the language pairs we experiment with, the resulting F-measure score from the combination of similarity measures and threshold matrix stays highly correlated.

6.3.4. Execution time

Execution time for each similarity varies to a large difference, depending on the complexity of the measures. There are 89.3 million comparisons performed to map the concepts in our experiment, hence the computation process required could take a long time. This can be seen in the logarithmic chart on Figure 5. The execution time in the cluster highlights that two of the tested similarity measures, TF-IDF and Soft TF-IDF, performed very slow compared to the other similarity measures. TF-IDF and Soft TF-IDF build a corpus first and use the corpus along with compared labels, instead of directly use the strings or tokens from the compared labels done by other similarity measures. Despite the slow performance, TF-IDF provides great F-Measure values and those values are robust to the change of filter values since, by nature, TF-IDF discounts the importance of less-determining words. Jaro and Jaro-Winkler are implemented using Cython[18] within the library (indicated with an asterisk in Table 5) so, in theory, these similarity measures should have a faster performance compared to pure-Python implementation. Cython is designed to approach the performance of C as a compiled programming language, instead of the Python as an interpreted programming language.

[18] https://cython.org/

However, cythonized implementation on those similarity measures do not significantly perform differently compared to other similarity measures that are not implemented with Cython. Most of the similarity measures took about only several minutes or less than an hour to run in the cluster, especially set-based similarity measures.

The robustness of TF-IDF can be used as a default choice for linking between multilingual concepts. However, the computational complexity for TF-IDF can be an issue if the datasets to be linked is high in volume. Token sort yields the highest F-Measure score in our experiment, and computational complexity is far less costly compared to TF-IDF, but a proper similarity threshold needs to be carefully approximated for Token Sort to yield the best result.

7. Conclusion and future work

In this paper, we have presented the IOTA framework that interlinks multilingual fiscal data by making use of the fiscal classifications. IOTA is designed using the distributed in-memory scalable platform (Apache Spark) to deal with the complex task of string similarity assessment for a large number of concepts. IOTA provides nineteen different measures to assess the similarities of the concepts. We have tested the performance of IOTA over data containing the three translated language pairs. We found that the best similarity measure with the relatively low computational cost is Token Sort. It provides the highest F-Measure score when the similarity threshold is properly approximated. TF-IDF also provides a high F-Measure across different similarity thresholds at the expense of significantly longer execution time. The correlation between language pairs shows a consistently high and positive correlation. IOTA can be easily adapted to be used for other use cases and domains.

In the future, the work could be extended in several dimensions: First, the performance of other machine translation tools could be compared. Second, it could be evaluated if additional preprocessing tasks improve the performance and accuracy of the mappings (such as stop words removal, in which additional stop words can be in specifically for open fiscal data domain). Third, a combination of multiple assessment methods can be exploited. Fourth, for hybrid similarity measures, it can be assessed which specific string similarity measure combination pairs would yield the best F-measure. Lastly, it is intended to integrate classifications to public semantic knowledge bases (e.g., DBpedia), so that concepts published by public administrators could be mapped to linked open data. We will also perform more experiments for integrating non-controlled translated concepts.

Acknowledgements

The first author would like to express gratitude to the German Academic Exchange Service (DAAD) for supporting him with a PhD scholarship.

8. References

Attard, J., Orlandi, F., Scerri, S., & Auer, S. (2015). A systematic review of open government data initiatives. *Government Information Quarterly*, *32*, 399–418. URL: http://dblp.uni-trier.de/db/journals/giq/giq32.html#AttardOSA15.

Bartolini, I., Ciaccia, P., & Patella, M. (2002). String matching with metric trees using an approximate distance. In A. H. F. Laender, & A. L. Oliveira (Eds.), *SPIRE* (pp. 271–283). Springer volume 2476 of *Lecture Notes in Computer Science*. URL: http://dblp.uni-trier.de/db/conf/spire/spire2002.html#BartoliniCP02.

Bertot, J. C., & Choi, H. (2013). Big data and e-government: issues, policies, and recommendations. In *Proceedings of the 14th Annual International Conference on Digital Government Research* (pp. 1–10). ACM.

Bilenko, M., Mooney, R. J., Cohen, W. W., Ravikumar, P., & Fienberg, S. E. (2003). Adaptive name matching in information integration. *IEEE Intelligent Systems*, *18*, 16–23. URL: http://dblp.uni-trier.de/db/journals/expert/expert18.html#BilenkoMCRF03.

Cohen, A. (2011). Fuzzywuzzy: Fuzzy string matching in python. URL: https://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/.

Conneau, A., Lample, G., Ranzato, M., Denoyer, L., & Jégou, H. (2017). Word translation without parallel data. *CoRR*, *abs/1710.04087*. URL: http://dblp.uni-trier.de/db/journals/corr/corr1710.html#abs-1710-04087.

Deerwester, S., Dumais, S., Furnas, G., Landauer, T., & Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, *41*, 391–407.

Ding, L., DiFranzo, D., Graves, A., Michaelis, J. R., Li, X., McGuinness, D. L., & Hendler, J. (2010). Data-gov wiki: Towards linking government data. In *2010 AAAI Spring Symposium Series*.

Doan, A., Halevy, A. Y., & Ives, Z. G. (2012). *Principles of Data Integration*. Morgan Kaufmann. URL: http://research.cs.wisc.edu/dibook/.

Dudaš, M., Horáková, L., Klímek, J., Kučera, J., Mynarz, J., Sedmihradská, L., Zbranek, J., & Dong, T. (2015). *Deliverable 1.4: (OpenBudgets.eu Data Model) User documentation*. Technical Report OpenBudgets.eu Consortium. URL: http://openbudgets.eu/assets/deliverables/D1.4.pdf.

European Commission (2008). Information system for european public procurement: Common procurement vocabulary. URL: https://simap.ted.europa.eu/cpv.

Firth, J. (1957). *A synopsis of linguistic theory 1930-1955*. Studies in Linguistic Analysis, Philological. Longman.

Huijboom, N., & Van den Broek, T. (2011). Open data: an international comparison of strategies. *European journal of ePractice*, *12*, 4–16.

Jaccard, P. (1901). Étude comparative de la distribution florale dans une portion des alpes et des jura. *Bulletin del la Société Vaudoise des Sciences Naturelles*, *37*, 547–579.

Jaro, M. A. (1980). *UNIMATCH, a Record Linkage System: Users Manual*. Bureau of the Census.

Jiang, Y., Li, G., Feng, J., & Li, W.-S. (2014). String similarity joins: An experimental evaluation. *Proceedings of the VLDB Endowment*, *7*, 625–636.

Joulin, A., Grave, E., Bojanowski, P., & Mikolov, T. (2016). Bag of tricks for efficient text classification. *CoRR*, *abs/1607.01759*. URL: http://dblp.uni-trier.de/db/journals/corr/corr1607.html#JoulinGBM16.

Karampatakis, S., Bratsas, C., Zamazal, O., Filippidis, P. M., & Antoniou, I. (2018). Alignment: A hybrid, interactive and collaborative ontology and entity matching service. *Information*, *9*. URL: http://www.mdpi.com/2078-2489/9/11/281. doi:10.3390/info9110281.

Lesnikova, T. (2016). *Liage de données RDF : évaluation d'approches interlingues. (RDF Data Interlinking : evaluation of Cross-lingual Methods)*.. Ph.D. thesis Grenoble Alpes University, France.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady* (pp. 707–710). volume 10.

Lund, K. (1995). Semantic and associative priming in high-dimensional semantic space. In *Proc. of the 17th Annual conferences of the Cognitive Science Society, 1995*.

Machado, A. L., & de Oliveira, J. M. P. (2011). Digo: An open data architecture for e-government. In *EDOCW* (pp. 448–456). IEEE Computer Society. URL: http://dblp.uni-trier.de/db/conf/edoc/edoc2011w.html#MachadoO11.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press. URL: http://www-csli.stanford.edu/%7Ehinrich/information-retrieval-book.html.

Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). Efficient estimation of word representations in vector space. In Y. Bengio, & Y. LeCun (Eds.), *ICLR (Workshop Poster)*. URL: http://dblp.uni-trier.de/db/conf/iclr/iclr2013w.html#abs-1301-3781.

Monge, A. E., Elkan, C. et al. (1996). The field matching problem: Algorithms and applications. In *KDD* (pp. 267–270).

Moussallem, D., Usbeck, R., Röder, M., & Ngomo, A.-C. N. (2017). Mag: A multilingual, knowledge-base agnostic and deterministic entity linking approach. In O. Corcho, K. Janowicz, G. Rizzo, I. Tiddi, & D. Garijo (Eds.), *K-CAP* (pp. 9:1–9:8). ACM. URL: http://dblp.uni-trier.de/db/conf/kcap/kcap2017.html#MoussallemURN17.

Musyaffa, F. A., Engels, C., Vidal, M.-E., Orlandi, F., & Auer, S. (2018a). Experience: Open fiscal datasets, common issues, and recommendations. *J. Data and Information Quality*, *9*, 19:1–19:10. URL: http://dblp.uni-trier.de/db/journals/jdiq/jdiq9.html#MusyaffaEVOA18.

Musyaffa, F. A., Halilaj, L., Li, Y., Orlandi, F., Jabeen, H., Auer, S., & Vidal, M. (2018b). Openbudgets.eu: A platform for semantically representing and analyzing open fiscal data. In *Web Engineering - 18th International Conference, ICWE 2018, Cáceres, Spain, June 5-8, 2018, Proceedings* (pp. 433–447). URL: https://doi.org/10.1007/978-3-319-91662-0_35. doi:10.1007/978-3-319-91662-0_35.

Musyaffa, F. A., Orlandi, F., Jabeen, H., & Vidal, M.-E. (2018c). Classifying data heterogeneity within budget and spending open data. In *ICEGOV* (pp. 81–90). ACM. URL: http://dblp.uni-trier.de/db/conf/icegov/icegov2018.html#MusyaffaOJV18.

Odell, M., & Russell, R. (1918). Patent numbers 1261167 (1918) and 1435663 (1922). *Washington, DC: US Patent Office*, .

On, B.-W., Koudas, N., Lee, D., & Srivastava, D. (2007). Group linkage. In R. Chirkova, A. Dogac, M. T. Özsu, & T. K. Sellis (Eds.), *ICDE* (pp. 496–505). IEEE Computer Society. URL: http://dblp.uni-trier.de/db/conf/icde/icde2007.html#OnKLS07.

Open Knowledge International (2017). Place overview: Global open data index. URL: http://index.okfn.org/place/.

Pappu, A., Blanco, R., Mehdad, Y., Stent, A., & Thadani, K. (2017). Lightweight multilingual entity extraction and linking. In M. de Rijke, M. Shokouhi, A. Tomkins, & M. Zhang (Eds.), *WSDM* (pp. 365–374). ACM. URL: http://dblp.uni-trier.de/db/conf/wsdm/wsdm2017.html#PappuBMST17.

py_stringmatching Documentation (2016). User manual for py_stringmatching. URL: http://anhaidgroup.github.io/py_stringmatching/v0.4.x/index.html.

Shadbolt, N., O'Hara, K., Berners-Lee, T., Gibbins, N., Glaser, H., Hall, W., & Schraefel, M. C. (2012). Linked open government data: Lessons from data.gov.uk. *IEEE Intelligent Systems*, *27*.

Sørensen, T. (1948). A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on danish commons. *Biol. Skr.*, *5*, 1–34.

The World Wide Web Foundation (2017). Open data barometer 4th edition — global report. URL: https://opendatabarometer.org/doc/4thEdition/ODB-4thEdition-GlobalReport.pdf.

Tversky, A. (1977). Features of similarity. *Psychological Review*, *84*.

Tygel, A. F., Attard, J., Orlandi, F., Campos, M. L. M., & Auer, S. (2016). How much? is not enough: an analysis of open budget initiatives. In *Proceedings of the 9th ICEGOV* (pp. 276–286). ACM.

United Nations Statistics Division (UNSD) (1999). Classification of the functions of government (cofog). URL: https://unstats.un.org/unsd/iiss/Classification-of-the-Functions-of-Government-COFOG.ashx.

Vijaymeena, M., & Kavitha, K. (2016). A survey on similarity measures in text mining. *Mach. Learn. Appl. Int. J*, *3*, 19–28.

Volz, J., Bizer, C., Gaedke, M., & Kobilarov, G. (2009). Silk - a link discovery framework for the web of data. In C. Bizer, T. Heath, T. Berners-Lee, & K. Idehen (Eds.), *LDOW*. CEUR-WS.org volume 538 of *CEUR Workshop Proceedings*. URL: http://dblp.uni-trier.de/db/conf/www/ldow2009.html#VolzBGK09.

Wache, H., Vögele, T., Visser, U., Stuckenschmidt, H., Schuster, G., Neumann, H., & Hübner, S. (2001). Ontology-based integration of information - a survey of existing approaches. In A. Gómez-Pérez, M. Gruninger, H. Stuckenschmidt, & M. Uschold (Eds.), *OIS@IJCAI*. CEUR-WS.org volume 47 of *CEUR Workshop Proceedings*. URL: http://dblp.uni-trier.de/db/conf/ijcai/ois2001.html#WacheVVSSNH01.

Winkler, W. E. (1993). *Improved Decision Rules in the Fellegi-Sunter Model of Record Linkage*. Technical Report Statistical Research Division, U.S. Census Bureau, Washington, DC.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., & Dean, J. (2016). Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, *abs/1609.08144*. URL: http://dblp.uni-trier.de/db/journals/corr/corr1609.html#WuSCLNMKCGMKSJL16.

Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauly, M., Franklin, M. J., Shenker, S., & Stoica, I. (2012). Resilient distributed datasets: A fault-tolerant abstraction for in-memory cluster computing. In S. D. Gribble, & D. Katabi (Eds.), *NSDI* (pp. 15–28). USENIX Association. URL: http://dblp.uni-trier.de/db/conf/nsdi/nsdi2012.html#ZahariaCDDMMFSS12.

Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M. J., Ghodsi, A., Gonzalez, J., Shenker, S., & Stoica, I. (2016). Apache spark: A unified engine for big data processing. *Commun. ACM*, *59*, 56–65. URL: http://doi.acm.org/10.1145/2934664. doi:10.1145/2934664.

Zobel, J., & Dart, P. (1996). Phonetic string matching: Lessons from information retrieval. In *Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* SIGIR '96 (pp. 166–172). New York, NY, USA: ACM. URL: http://doi.acm.org/10.1145/243199.243258. doi:10.1145/243199.243258.