Learning Advanced Similarities and Training Features for Toponym Interlinking

Giorgos Giannopoulos¹(✉), Vassilis Kaffes^{1,2}, and Georgios Kostoulas¹

¹ IMSI/Athena Research Center, Marousi, Greece
giann@athenarc.gr

² University of the Peloponnese, Tripoli, Greece

Abstract. Interlinking of spatio-textual entities is an open and quite challenging research problem, with application in several commercial fields, including geomarketing, navigation and social networks. It comprises the process of identifying, between different data sources, entity descriptions that refer to the same real-world entity. In this work, we focus on toponym interlinking, that is we handle spatio-textual entities that are exclusively represented by their name; additional properties, such as categories, coordinates, etc. are considered as either absent or of too low quality to be exploited in this setting. Toponyms are inherently heterogeneous entities; quite often several alternative names exist for the same toponym, with varying degrees of similarity between these names. State of the art approaches adopt mostly generic, domain-agnostic similarity functions and use them as is, or incorporate them as training features within classifiers for performing toponym interlinking. We claim that capturing the specificities of toponyms and exploiting them into elaborate meta-similarity functions and derived training features can significantly increase the effectiveness of interlinking methods. To this end, we propose the *LGM-Sim* meta-similarity function and a series of novel, similarity-based and statistical training features that can be utilized in similarity-based and classification-based interlinking settings respectively. We demonstrate that the proposed methods achieve large increases in accuracy, in both settings, compared to several methods from the literature in the widely used Geonames toponym dataset.

Keywords: Interlinking · Machine learning · Toponym · Spatio-textual entities · Feature extraction · String similarity

1 Introduction

Interlinking (alt. deduplication, entity matching/linking, record linkage), in its most common form, is the task of identifying, from two entity sources, pairs of entity descriptions that correspond to the same real world entities. Interlinking is a crucial task in several domains, since it is quite often the case that real world entities are modeled, represented and gathered by different stakeholders, following different schemas, procedures and quality standards. As a result, multiple

databases might exist for representing the same groups of real world entities, in heterogeneous ways. Examples include: person names, product names and spatio-textual entities (toponyms, POIs, addresses) in different data providers.

In this paper, we examine the problem of interlinking spatio-textual entities, based solely on their name, i.e. we handle the problem of *toponym interlinking*. A toponym might refer to a broad range of spatio-textual entities, from small places, to countries. In our problem setting, the name of a spatio-textual entity/toponym, is its only *reliable* attribute that can be used for identifying same entities; the other attributes, such as spatial coordinates, categories, extended textual descriptions, are either non-existent or of too low quality/accuracy to be used for interlinking. Consider the scenario of a toponym data provider that maintains a proprietary toponym database, and periodically enriches/extends it with toponym entities extracted from user check-ins in social media. It is quite often the case that a check-in regarding a specific place is performed in locations that are considerably distant from the actual place. In this case, the coordinates extracted for the specific place are inaccurate and might even hurt the interlinking process, e.g. by leading to the rejection of a link between two toponyms that are actually the same, but appear to have distant locations. The authors of [4] elegantly describe the problem of dealing with *extremely noisy location coordinates* in the Facebook database. In another scenario, the recognition and extraction of toponyms might be performed on documents (e.g. travel guides), where the coordinates of toponyms are non-existent. Further, in either scenario, no other properties of the toponyms, such as categories or extended textual descriptions, can generally be retrieved for the majority of the extracted toponyms.

Competitive approaches from the literature utilize generic string similarity measures to solve the problem and limit their contributions to tuning their parameters or using them as training features in machine learning (ML) algorithms for classification. We take a different approach, claiming that domain knowledge is a critical factor for toponym interlinking that needs to be captured and incorporated within the interlinking process. To this end, we analyse a large toponym dataset, Geonames¹, which contains, among other toponym metadata, alternative names for millions of toponyms. Based on the insights we gain, we build an elaborate meta-similarity function, **LGM-Sim**, that takes into account and incorporates within its processing steps the specificities of toponym names. Additionally, we derive training features from **LGM-Sim**, that can be used for interlinking via classification. We demonstrate the superiority of the proposed models in two settings: similarity-based and classification-based toponym interlinking.

The rest of the paper is organized as follows. Section 2 discusses related work on interlinking, emphasizing on spatio-textual data. Section 3 defines the toponym interlinking problem and presents the two generic methodologies for solving it and, further, briefly discusses our findings and insights on the domain specificities of toponyms. Section 4 presents our proposed methods that incorporate the aforementioned insights, including domain specific similarity functions and training features, for toponym interlinking. Section 5 evaluates the effectiveness of

¹ <https://www.geonames.org/>.

the proposed methods in two different interlinking settings: similarity-based and classification-based. Finally, Sect. 6 concludes the paper.

2 Related Work

Considering the more general problem of name matching, various methods are proposed in the literature, with several previous studies [2,3] performing thorough evaluations of the most prominent ones in several datasets and demonstrating that there are no distinctively better methods that surpass all others in all settings/datasets. A generic framework for named entity interlinking is presented in [10], aiming to properly handle a variety of generic tasks where accuracy is not the most important criterion. In this frame, an enhancement of the Soft-TFIDF measure, combined with the Levenshtein similarity, is proposed.

Metrics specifically designed for toponym interlinking that mostly correspond to variations of the procedures used for generic name matching are proposed in [5,8]. The DAS similarity measure [8] comprises a hybrid, three-stages method that combines features from token-based and edit-based approaches. The meta-similarity proposed in [5] takes into account accentuation and other language-specific aspects of toponym names, in a four-stages process. The set of algorithms evaluated in [2] were assessed in the toponym interlinking problem by [12]. The authors experimented on place names listed in the GEOnet Names Server, that contains romanized toponyms from 11 different countries. Based on their study, no similarity measure achieves the highest accuracy in all datasets. In [4], the problem of business places deduplication is studied, taking a different approach. In the proposed solution, certain words (*core terms*) are identified, that are of higher significance in the name deduplication process. Based on these terms, a name model is constructed and properly combined with a spatial context model, using unsupervised learning algorithms.

Several works apply supervised machine learning approaches by extracting training features on name, coordinates, category, as well as textual, topological and semantic similarity, and utilizing them within classifiers [1,9,15,16]. A framework for improving duplicate detection using learnable text distance functions is presented in [1], however, in this work too, generic string similarity measures are used for the feature scores computation. The authors of [15] extract features for location name, coordinates, location type and demographic information. Then, machine learning algorithms are used to weight all features to solve the spatial entity matching problem. The work in [16] proposes a machine learning based approach to detect duplicate location entities. For this, a proposed metric is calculated for each key feature consisting of name, address and category similarity that describe entity pairs. Then, these extracted features are fed in a classification model to decide whether two entities are duplicates or not. Support Vector Machine (SVM) and an alternating Decision Tree classifiers are used to combine different similarity features in [9]. The authors consider a variety of features corresponding to place name similarity, geospatial footprint similarity, place type similarity, similarity measures corresponding to semantic

relations and temporal similarity. Contrary to these works, our study bases the matching process exclusively on toponym names. Utilizing richer spatio-textual profiles that enable the construction of features based on location-based, temporal or categorical similarity is beyond the scope of this work.

Most closely comparable to ours is the work presented in [14], where a thorough overview of the literature and an extended comparison of 13 different string similarity functions on toponym interlinking is performed. Additionally, these similarity functions are also assessed as training features within state of the art supervised machine learning algorithms for classification. Similarly, most works reviewed in [14] do not take into account the specificities of toponyms, however, using these methods as baselines for comparison allows as to compare with a large part of the literature. Our preliminary work on the problem [6], presents a first version of the **LGM-Sim**, a meta-similarity function aiming to capture the specificities of toponyms. In [6] we demonstrate that applying **LGM-Sim** on top of several baseline similarity functions improves their interlinking accuracy by a large extent. In our current work, we fine-tune the **LGM-Sim** function and we exploit it into deriving training features to be used within classification algorithms for toponym interlinking, demonstrating further increases in accuracy.

Deep Learning methods for toponym interlinking are also being proposed in the literature. [13] present such a method, where Siamese RNNs are applied, yielding better accuracy results than traditional classifiers on similarity-based training features. Incorporating Deep Learning methods eliminates the need for feature extraction and engineering, however, it requires large amounts of data to train proper models, as well as engineering proper DNN architectures. The goal of the currently presented work is to demonstrate the potential and the gains of incorporating domain knowledge into generic similarity measures and classifiers for toponym interlinking. Comparing traditional classification methods with Deep Learning methods, as well as devising approaches for exploiting both worlds comprises part of our ongoing work. Thus, in the current manuscript, the approach of [13] is considered orthogonal but potentially complementary.

A different research strand focus on improving the efficiency of the interlinking process [11], since, in its naive version, it is an $O(n^2)$ problem, and thus prohibitive for large datasets. Such methods consider as a given that the similarity functions they apply will be sufficiently accurate on identifying same entities, and focus on developing indexes, structures and schemes for optimizing the performance of the interlinking process, in terms of time-efficiency and scalability. Following a different rationale, the Magellan system, presented in [7], aims to provide the tools to end users to exploit a wide range of techniques related to the entity matching process under a unified framework. However, the implemented similarity functions are generic, widely adopted similarities that are not specialized or properly tuned for the setting of comparing geospatial entities. Further, an integral concept in Magellan is user interaction, prescribing pipelines where automated interlinking tasks and user feedback are iteratively combined. These categories of works on interlinking solve orthogonal/complementary problems and are not directly comparable to our proposed methods.

3 Background and Domain Knowledge

3.1 Problem Formulation

Given a set of candidate toponym pairs, toponym interlinking can be formalized as the problem of learning a decision function that decides whether a candidate toponym pair contains two toponyms that correspond to the same real world spatio-textual entity². One can identify two generic methodologies for solving the above problem.

Similarity comparison methods apply, on every candidate pair of toponyms, a string similarity function, in conjunction with a similarity threshold to compare the names of the two toponyms. If the similarity score, produced by the similarity function, surpasses or equals the defined threshold, then the pair of toponyms is marked as **True**, otherwise, it is marked as **False**. Depending on the similarity function, several parameters, such as thresholds and weights, need to be tuned either by learning them on a dedicated training set or by being empirically selected by domain experts via tuning/trial-and-error procedures.

Classification methods train a binary classification algorithm that takes as input a candidate pair of toponyms and classifies it in either of the two available classes: {**True**, **False**}. In this case, a feature extraction process needs to be performed before the classification algorithm is deployed, in order to represent candidate pairs into an appropriate feature space, capturing meaningful toponym properties and relations with respect to the task at hand. After the above representations are defined, the classifier is trained on historical data, i.e. candidate toponym pairs for which we already know their class. The trained classifier is then able to decide whether a new pair of toponyms is **True** or **False**.

The literature on toponym interlinking includes methods adopting both presented methodologies. A detailed comparison of most of the proposed methods is performed in [14], as well as in our evaluation, which follows a similar experimental setting. The goal of this paper is to establish the value of learning/elaborate domain specific similarities for toponym interlinking. Given that most of the aforementioned methodologies are either based or can benefit from similarity measures and/or respective training features, the contribution of our methods touches and can potentially improve a wide range of interlinking methods.

3.2 Concepts and Intuitions

For our analysis, it is useful to first discuss the concept of *core name* and *core term*, also similarly defined, but differently handled in [4]. Core name consists in the subset of terms from the toponym name that are the most important in distinguishing the toponym from other ones, and, respectively, identifying same toponyms. A core term is a single term contained in the core name. The concept of core name cannot be strictly/formally defined, since it largely depends on human understanding on toponyms and domain knowledge. However, we believe

² Obtaining the set of candidate toponym pairs is an orthogonal problem, with several efficient solutions in the literature, like *blocking* [11]; in what follows, we consider this set available and focus on the problem of toponym interlinking, as defined above.

that approximating the identification of core terms and handling them differently within a meta-similarity function may yield increased interlinking accuracy.

One of the major specificities of toponyms, compared to namings of other entities, is the fact that different terms within their name might largely vary with respect to their significance in deciding whether two toponyms are the same. This variability might also take several forms, such as: (i) The existence of a frequent term that provides categorical information of the toponym, like “community” or “square”. Such terms do not comprise a part of the core name of the toponym, and need to be handled differently; (ii) Terms that comprise part of the core name of a toponym might be of different significance and, thus, some of them might be omitted from its name in one data source, while maintained in the name in another source. An indicative example would be the toponym: “St. Paul’s German Lutheran Church”. In this case, “Church” could be considered a non-core term, however, it is expected to be a frequent term. Also, “St. Paul’s” potentially has higher significance than “German Lutheran” in distinguishing/interlinking the toponym from/with another one.

Additionally, toponyms are inherently characterized by large variability in the terms that actually comprise their core name. That is, the same term might be spelled in different ways, or be expressed in abbreviated, or generally altered, forms. This fact complicates the identification of proper matchings between name terms within a toponym. The aforementioned variability extends to punctuation and accentuation too. A representative example is the following matching pair “Solovejcev Kljuch - Soloveytshev Klyuch”.

Another issue lies in the order of the toponym terms. Two variations of the same toponym might contain the same terms in different order; term ordering becomes even more cumbersome in case some of these terms are missing in one of the names. In this case, it is uncertain whether sorting the terms of each toponym aphanumerically, before comparing them, will facilitate or hinder their similarity comparison process. A representative example is the pair: “Lake Thompson - Thompson Lake Reservoir”. Not only the two toponyms contain core terms in different order but also only one of them contains the term “Reservoir”.

We note that all the presented examples are drawn from Geonames, a large toponym dataset on which we perform our analysis and which contains hundreds of thousands of such cases and specificities. The **LGM-Sim** meta-similarity function, proposed next, comprises several string processing steps that take into account and handle the aforementioned toponym specificities.

4 Models for Toponym Interlinking

In this section, we present **LGM-Sim**, a meta-similarity function for toponym interlinking, that incorporates domain knowledge on toponyms within its processing. Consequently, we discuss how **LGM-Sim** can be transformed into training features and utilized, along with previously proposed similarity-based features and additional statistical features, within classifiers for toponym interlinking.

4.1 LGM-Sim Meta-Similarity for Toponym Interlinking

A high level description of the **LGM-Sim** meta-similarity is provided in Algorithm 1. **LGM-Sim** takes as input two toponym strings, while it has a set of parameters regarding comparison thresholds and individual score weighting. Additionally, it considers a set of frequent terms that can be automatically gathered by the corpus of toponyms that are to be interlinked. All the parameters of **LGM-Sim** can be automatically learned by evaluating the effectiveness of different parameterizations on a small training dataset, and selecting the parameters that yield the highest accuracy. We show in our experiments that it is sufficient to train **LGM-Sim** in a much smaller train dataset than the deployment dataset.

LGM-Sim aims at properly splitting the compared toponym strings into discrete lists of terms, with each list containing terms of different semantics. First, **LGM-Sim** initializes a list of punctuation marks and two initially empty lists of terms for the two toponyms, $\mathcal{S}_1, \mathcal{S}_2$ respectively (lines 6–7). Next, an initial pre-processing step on the two strings is performed by **TransformNames**, including lowercasing, transliteration and punctuation/accenuation alignment (line 8).

Algorithm 1: LGM-Sim

Input: name of toponym A, s_1 ; name of toponym B, s_2

- 1 set of frequent terms in the corpus of names $\mathcal{S}_{FT}$;
- 2 total similarity comparison threshold θ_{sim} ;
- 3 threshold used in sorting terms θ_{sort} ;
- 4 threshold used in splitting names θ_{split} ;
- 5 weights for each name part $w_{base}, w_{mis}, w_{freq}$;
- Output:** (*True* || *False*) on whether toponyms A and B are the same
- 6 $punct \leftarrow [',', '“”, '”', '!', '?', '/', '“’, ‘.’, ‘-’]$
- 7 $\mathcal{S}_1, \mathcal{S}_2 \leftarrow []$
- 8 $(s_1, s_2) \leftarrow \text{TransformNames}(s_1, s_2, punct)$
- 9 $(\mathcal{S}_1, \mathcal{S}_2) \leftarrow \text{SortTerms}(s_1, s_2)$
- 10 $(\mathcal{S}_1^{base}, \mathcal{S}_2^{base}, \mathcal{S}_1^{freq}, \mathcal{S}_2^{freq}) \leftarrow \text{ExtractFrequentTerms}(\mathcal{S}_1, \mathcal{S}_2, \mathcal{S}_{FT})$
- 11 $(\mathcal{S}_1^{base}, \mathcal{S}_2^{base}, \mathcal{S}_1^{mis}, \mathcal{S}_2^{mis}) \leftarrow \text{CompareCoreTerms}(\mathcal{S}_1^{base}, \mathcal{S}_2^{base}, \theta_{split})$
- 12 $matchScore \leftarrow \text{WeightedSim}(\mathcal{S}_1^{base}, \mathcal{S}_2^{base}, w_{base}, \mathcal{S}_1^{mis}, \mathcal{S}_2^{mis}, w_{mis}, \mathcal{S}_1^{freq}, \mathcal{S}_2^{freq}, w_{freq})$
- 13 **return** $matchScore \geq \theta_{sim}$

Then, the terms within the two toponym strings are sorted alphanumerically and stored to the initialized lists (line 9). The first step of **SortTerms** is to concatenate all the (unsorted) terms in the initial strings and then compare the two concatenated strings with a loose threshold θ_{sort} . If the two concatenated strings are similar enough, then the function returns the two lists of terms *unsorted*. The rationale is that, if the initial, unsorted strings are similar enough, then sorting their terms might reduce their similarity, e.g. by re-ordering small terms that are not common in both strings or that start with different alphabet letter.

On the other hand, if the two initial strings are not similar enough, the function returns two alphanumerically sorted lists of their terms, since, in this case, it is quite probable that sorting will increase their similarity.

Then, the first step of splitting the two toponym strings into separate lists is performed (line 10). **ExtractFrequentTerms** identifies frequent terms within $\mathcal{S}_1, \mathcal{S}_2$, removes them and adds them in two new lists, $\mathcal{S}_1^{freq}, \mathcal{S}_2^{freq}$. The remaining lists are now called $\mathcal{S}_1^{base}, \mathcal{S}_2^{base}$ and contain terms more probable to be *core* ones.

Thereafter, the second step of splitting the toponyms into separate lists is performed (line 11). Specifically, **CompareCoreTerms** further splits each of the *base* toponym lists into the new lists, containing matching and non-matching terms, as follows. First, four empty lists are initialized, so as to be filled with the matching and non-matching terms of each toponym. Then, the two input lists of *base* terms of the two toponyms are parsed simultaneously, term by term. At each step, if the two considered terms from the two lists (loosely) match, then they are permanently stored as *base* terms. If the two terms do not match, the parsing proceeds only in the list with the alphanumerically lowest term, while the specific term is stored in the respective *mismatch* list. Finally, the remaining, mismatched terms are added to the respective *mismatch* lists, and the function returns the four new lists $\mathcal{S}_1^{base}, \mathcal{S}_2^{base}, \mathcal{S}_1^{mis}, \mathcal{S}_2^{mis}$, containing matching base terms and mismatching terms from the two toponym strings.

At this stage, the two initial toponym strings have been split in three lists of terms each: (i) Two *base* lists containing potential *core terms* of the toponyms that are identified to match between the two toponyms; (ii) Two *mismatch* lists containing potential *core terms*, that, however, have not been matched between the two toponyms; (iii) Two *frequent terms* lists that contain terms from the toponym that are frequently found in the corpus, and thus might not belong to the *core names* of the toponyms, functioning auxiliary to them.

Next, **CompareCoreTerms** calculates three similarity scores, comparing individually the three different types of lists, and properly weights the three individual similarity scores in order to produce the final similarity score for the toponyms (line 12). For this, three individual weights are utilized, w_b, w_m and w_f , which represent the significance of the similarity scores calculated on the three term lists that are individually compared for each pair of toponyms. Each comparison process first examines whether the input lists are empty, so as to re-adjust the corresponding weights for the individual similarity scores. For example, if for a pair of toponyms no frequent terms have been identified, then the score weight w_f is set to null and the rest weights w_b and w_m are proportionately increased according to the lengths of the respective term lists. Next, the significance weights w_b, w_m and w_f are re-calculated, taking into account the lengths of the corresponding lists they refer to. This process is performed in order to compensate for large discrepancies between the lengths of lists (measured in total number of characters) of different types. Finally, the individual scores are summed and the final similarity score is returned by the function (line 13).

In the last step (line 13), **LGM-Sim** compares the final score calculated with a similarity comparison threshold θ_{sim} . Depending on the result of the comparison, the value **True** or **False** is returned denoting whether the two toponyms are the

same or not. **LGM-Sim** is a meta-similarity function, thus it can be applied on top of any generic similarity measure. Following the evaluation paradigm of [14], we consider a large set of similarity measures, presented in Table 1. Presenting the specifics of these similarity measures is out of the scope of this paper. However, these measures are well studied in the literature and the reader can refer to [14] for a short presentation of each of them.

Table 1. Considered similarity measures

Damerau-Levenshtein	Jaro
Jaro-Winkler	Jaro-Winkler Reversed
Sorted Jaro-Winkler	Cosine N-Grams
Jaccard N-Grams	Dice Bi-Grams
Jaccard Skip-grams	Monge-Elkan
Soft-Jaccard	Davis and De Salles
Tuned Jaro-Winkler	Tuned Jaro-Winkler Reversed

We note that the last line of Table 1 presents two similarity measures that are not presented in [14] or [6], rather than comprise a variation of the respective Jaro-Winkler measures, that we propose in the current work, taking into account the characteristics of toponyms that we have studied in Geonames. In particular, it adds the notion of skip-grams into the Jaro-Winkler metric and, especially, the Winkler part, that gives higher scores to strings that match from the beginning up to a given prefix length. Thus, we allow for a gap of one character in the matching of the prefixes between two strings. The same applies to its reverse variation by considering the endings of the strings.

4.2 LGM-Sim Based Classifiers for Toponym Interlinking

The generic process of training classifiers for toponym interlinking is described in Sect. 3.1. Here, we discuss the training features we introduce, for better capturing and exploiting the domain knowledge of toponyms. One of the major merits of training a classifier is the combinatorial exploitation of several features within a model. While in the generic similarity comparison based setting, only one similarity function at a time can be examined/deployed, in the classification based setting, several similarity functions can be encoded as training features of the model. Past approaches, as well as the study presented in [14], have used combinations of the similarity measures presented in Table 1 as training features.

In this work, we adopt the aforementioned set of training features, however, we enrich it with a corresponding set of features generated by applying the **LGM-Sim** on all the similarity measures presented in Table 1. Further, we consider an “intermediate” set of training features, corresponding again to the similarity measures of Table 1, however, having performed only the sorting function of

LGM-Sim, before comparing the toponym strings. Additionally, we consider the three individual similarity scores calculated on the three individual lists that **LGM-Sim** splits the two toponym names. That is, we consider $score_b$, $score_m$ and $score_f$, derived by individually comparing $(\mathcal{S}_1^{base}, \mathcal{S}_2^{base})$, $(\mathcal{S}_1^{mis}, \mathcal{S}_2^{mis})$ and $(\mathcal{S}_1^{freq}, \mathcal{S}_2^{freq})$, as separate features, allowing the model to learn the significance of the similarity of each of these individual parts of the toponym names, along with the significance of their total similarity.

Table 2. Considered training features

Feature Type	Number of Features
Basic similarity measures	14
Sorted similarity measures	13
LGM-Sim based similarity measures	13
Individual matching scores from LGM-Sim based on Damerau	3
Statistical features	44

Further, we define a set of non-similarity based features, that concern statistical aspects of the compared toponyms. Specifically, for each pair of toponyms, the number of terms contained in each forms two integer features; the existence of a frequent term in each forms two boolean features, while the existence of one or more of the 20 more frequent terms in the whole dataset in each toponym forms 40 additional boolean features. Eventually, we consider five groups of training features (Table 2): (i) the *basic* similarity features as presented in [14]; (ii) the *sorted* similarity features; (iii) the *LGM-Sim* similarity features; (iv) the individual scores on the split toponyms produced by applying *LGM-Sim* based Damerau-Levenshtein similarity; and (v) the statistical features on toponyms.

5 Evaluation

This section presents the evaluation of the proposed methods for toponym interlinking, with respect to the two different settings presented in Sect. 3.1: *similarity-based* and *classification-based*. In the former setting, we compare the interlinking effectiveness of our proposed **LGM-Sim** meta-similarity against the traditional similarity measures and functions that are used in several works of the literature (Table 1). In the latter setting, we compare the interlinking effectiveness of our proposed method, that uses additional, novel, **LGM-Sim**-derived and statistical features within classifiers, against the approach presented in [14], that only uses traditional similarity measures as features³.

³ We note that the baselines we compare with cover a large part of the presented literature, presented in the following papers: [2, 3, 5, 12, 14].

The evaluation dataset is drawn from Geonames, a database that contains more than 11 Million toponyms from around 250 countries. For each toponym in the dataset, there exists its main name and a list of alternate names. By following the exact procedure of [14], we construct a balanced dataset of 5M **True** and **False** toponym pairs. The False toponym pairs are created by selecting a name and an alternate name from different toponym records of the initial dataset. The True toponym pairs are created by selecting the name and the alternate name from the same toponym record, but ensuring, to some extent, that some of the created pairs vary in their name. To measure the effectiveness of the evaluated methods, four standard IR measures are adopted: *Accuracy*, *Precision*, *Recall*, *F1-Score*. To better evaluate the generalization capacity of the compared methods, we slightly modify the setting used in [14], introducing a separate training set, where compared similarity methods are trained⁴. We keep the 5M toponym pairs test set the same as in [14], for evaluating the trained models. The training set contains 100K toponym pairs, equally balanced between **True** and **False**. The code and the respective datasets are available on GitHub⁵.

5.1 Evaluation Results

We denote as **Basic** all baseline models as presented in [14], while as **LGM/LGM-Sim** all our proposed models. Moreover, we mark with bold the best reported value, per evaluation measure, and per compared approach (**Basic** vs. **LGM-Sim**). We note that we exclude from our experiments the *Permuted Jaro-Winkler* similarity measure, since it is reported in [14] that it is orders of magnitude slower than the rest similarity measures, without any substantial gains in interlinking effectiveness. Also, we do not report values for the **LGM-Sim** version of the *Jaro-Winkler Sorted* similarity, since it would conflict with the fact that **LGM-Sim** incorporates its own mechanism for deciding whether to sort or not two toponym strings.

Examining the effectiveness of the compared methods in the test set⁶, Table 3 demonstrates that **LGM-Sim** improves the Accuracy of all baseline models by 8–15%. Specifically, in the similarity-based setting, the best **LGM-Sim** model increases the Accuracy of the best **Basic** model by 14.9%; in the classification setting, the respective increase is 8.1%. Similar observations stand for the rest of the evaluation measures where both Precision and Recall of the models increase, noting that **LGM-Sim** meta-similarity seems to give a large boost to Recall, whereas Precision

⁴ To learn the hyper-parameters for each classification model, we perform 5-fold cross-validation on the train set, averaging the Accuracy score for each examined hyper-parameterization; the one with the higher Accuracy is selected for the classifier. The weights and thresholds used within the similarity measures are also handled as parameters of the models and learned on the training set. Reporting the optimal values for these is omitted due to lack of space, however, they can be reproduced by executing the referenced GitHub code.

⁵ <https://github.com/LinkGeoML/ToponymInterlinking>.

⁶ Similar numbers and differences are also reported in the training set but omitted due to lack of space.

is the one significantly boosted on the **LGM-Sim** derived features within classifiers. Another observation is that the **LGM-Sim** meta-similarity methods close the gap between similarity comparison and classification based methods, making the former an acceptable solution in scenarios where the more heavy-weight classification models are not an option. A third observation is that the introduced *Tuned Jaro-Winkler Reversed* similarity marginally increases the effectiveness of the *Jaro-Winkler Reverse* in the similarity based setting (79.8% vs. 79.6%), comprising a seemingly insignificant boost in Accuracy.

Table 3. Evaluation results on test dataset (5M)

	Accuracy		Precision		Recall		F1-score	
	Basic	LGM	Basic	LGM	Basic	LGM	Basic	LGM
Damerau-Levenshtein	0.645	0.780	0.791	0.830	0.393	0.704	0.526	0.762
Jaro	0.634	0.771	0.776	0.826	0.377	0.686	0.508	0.750
Jaro-Winkler	0.632	0.768	0.722	0.778	0.431	0.748	0.540	0.763
Jaro-Winkler Reverse	0.646	0.796	0.782	0.830	0.405	0.744	0.533	0.784
Jaro-Winkler Sorted	0.615	—	0.719	—	0.377	—	0.495	—
Cosine n-grams	0.614	0.718	0.710	0.741	0.386	0.669	0.500	0.703
Jaccard n-grams	0.609	0.709	0.753	0.785	0.326	0.575	0.455	0.664
Dice bi-grams	0.621	0.731	0.761	0.758	0.352	0.678	0.481	0.716
Jaccard skip-grams	0.625	0.738	0.741	0.753	0.385	0.710	0.507	0.730
Monge-Elkan	0.595	0.764	0.664	0.775	0.385	0.745	0.488	0.759
Soft-Jaccard	0.594	0.762	0.705	0.767	0.322	0.751	0.442	0.759
Davis/De Salles	0.617	0.771	0.716	0.787	0.389	0.742	0.504	0.764
Tuned Jaro-Winkler	0.630	0.770	0.728	0.796	0.413	0.725	0.527	0.759
Tuned Jaro-Winkler Reverse	0.649	0.798	0.808	0.822	0.390	0.761	0.526	0.791
Gradient Boosted Trees	0.773	0.854	0.764	0.876	0.790	0.824	0.777	0.849
SVM	0.719	0.824	0.688	0.864	0.802	0.768	0.741	0.813
Random Forests	0.770	0.849	0.769	0.877	0.772	0.811	0.770	0.843
Extr. Rand. Trees	0.766	0.844	0.769	0.878	0.760	0.799	0.765	0.837
Decision Tree	0.718	0.778	0.708	0.779	0.741	0.775	0.724	0.777

Finally, we examine the most informative training features utilized by the best classifier on our problem, in terms of higher Accuracy, i.e. the *Gradient Boosted Trees*. Table 4 presents the top-10 features from higher to lower order of importance. A first observation is that the majority (7/10) of the top-10 training features are introduced in this work. This is consistent with the findings from Table 3. Another observation is that variations of our proposed *Tuned Jaro-Winkler* measure comprise 3/10 most informative features, demonstrating that the proposed modification, despite its marginal effect in the similarity-based setting, it is rather useful in the classification based setting. Further, it becomes evident that appropriate sorting of toponym strings, following the processing of

SortTerms, is of high importance, since 5/10 features incorporate this processing (either plain *Sorted* or *LGM-Sim* features). Finally, although the introduced statistical features seem promising, it is evident that they require more elaboration, since they occupy only the last two positions in the list.

Table 4. Evaluation of top-10 most important training features for the best classifier

Ranking	Gradient Boosted Trees
1	Damerau-Levenshtein Sorted
2	Tuned Jaro-Winkler
3	Sorted Jaro-Winkler Reverse
4	LGM-Sim-Damerau-Levenshtein
5	Tuned Jaro-Winkler Reverse
6	LGM-Sim-Tuned Jaro-Winkler Reverse
7	LGM-Sim-Jaro-Winkler Reverse
8	Jaccard n-grams
9	Top-20 Freq. Term exists in ($\mathcal{S}_1$)
10	Number of terms in string ($\mathcal{S}_1$)

Regarding computation times, we present some indicative runtimes to showcase that the **LGM-Sim** methods do not introduce prohibitive overheads. Indicatively, for 5M toponym pairs, *JW Reversed* similarity runs in 10s, while its **LGM-Sim** version in 114s; both times are marginal considering the magnitude of the dataset. Respectively, *Gradient Boosted Trees* with the baseline features runs in 113 min., while its **LGM-Sim** version in 135 min., introducing less than 20% overhead, which is negligible considering the Accuracy gains presented above.

6 Conclusion

In this paper, we presented domain specific models that can be applied for toponym interlinking. We demonstrated that the proposed meta-similarity, **LGM-Sim** and the training features derived from it consistently, and to a large extent, improve the interlinking effectiveness of widely used baseline models. As future work, further examining and refining non-similarity based training features (e.g. structural, statistical), as well as combining traditional features and methods with Deep Learning/embedding-based methods comprise promising directions.

Acknowledgments. This research has been co-financed by the European Regional Development Fund of the European Union and Greek national funds through the Operational Program Competitiveness, Entrepreneurship and Innovation, under the call RESEARCH – CREATE – INNOVATE (project codeT1EDK-04568).

References

1. Bilenko, M., Mooney, R.J.: Adaptive duplicate detection using learnable string similarity measures. In: Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2003, pp. 39–48. ACM, New York (2003). <https://doi.org/10.1145/956750.956759>, <http://doi.acm.org/10.1145/956750.956759>
2. Christen, P.: A comparison of personal name matching: techniques and practical issues. In: Sixth IEEE International Conference on Data Mining - Workshops (ICDMW 2006), pp. 290–294, December 2006. <https://doi.org/10.1109/ICDMW.2006.2>
3. Cohen, W.W., Ravikumar, P., Fienberg, S.E.: A comparison of string distance metrics for name-matching tasks. In: Proceedings of the 2003 International Conference on Information Integration on the Web, IIWEB 2003, pp. 73–78. AAAI Press (2003). <http://dl.acm.org/citation.cfm?id=3104278.3104293>
4. Dalvi, N., Olteanu, M., Raghavan, M., Bohannon, P.: Deduplicating a places database. In: Proceedings of the 23rd International Conference on World Wide Web, WWW 2014, pp. 409–418. ACM, New York (2014). <https://doi.org/10.1145/2566486.2568034>, <http://doi.acm.org/10.1145/2566486.2568034>
5. Davis, C.A., de Salles, E.: Approximate string matching for geographic names and personal names. In: GeoInfo, pp. 49–60, January 2007
6. Kaffes, V., Giannopoulos, G., Karagiannakis, N., Tsakonas, N.: Learning domain specific models for toponym interlinking. In: Proceedings of the 27th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems, SIGSPATIAL 2019, Chicago, IL, USA, 5–8 November 2019, pp. 504–507 (2019). <https://doi.org/10.1145/3347146.3359339>
7. Konda, P., et al.: Magellan: toward building entity matching management systems. PVLDB **9**, 1197–1208 (2016)
8. KilingS, D.: An accurate toponym-matching measure based on approximate string matching. J. Inf. Sci. **42**(2), 138–149 (2016). <https://doi.org/10.1177/0165551515590097>
9. Martins, B.: A supervised machine learning approach for duplicate detection over gazetteer records. In: Claramunt, C., Levashkin, S., Bertolotto, M. (eds.) GeoS 2011. LNCS, vol. 6631, pp. 34–51. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-20630-6_3. <http://dl.acm.org/citation.cfm?id=2008664.2008669>
10. Moreau, E., Yvon, F., Cappé, O.: Robust similarity measures for named entities matching. In: Proceedings of the 22nd International Conference on Computational Linguistics - Volume 1, COLING 2008, pp. 593–600. Association for Computational Linguistics, Stroudsburg (2008). <http://dl.acm.org/citation.cfm?id=1599081.1599156>
11. 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). <https://doi.org/10.14778/2856318.2856326>. <http://www.vldb.org/pvldb/vol9/p312-papadakis.pdf>
12. Recchia, G., Louwerse, M.: A comparison of string similarity measures for toponym matching. In: COMP 2013 - ACM SIGSPATIAL International Workshop on Computational Models of Place, pp. 54–61, November 2013. <https://doi.org/10.1145/2534848.2534850>
13. Santos, R., Murrieta-Flores, P., Calado, P., Martins, B.: Toponym matching through deep neural networks. Int. J. Geogr. Inf. Sci. **32**, 1–25 (2017). <https://doi.org/10.1080/13658816.2017.1390119>

14. Santos, R., Murrieta-Flores, P., Martins, B.: Learning to combine multiple string similarity metrics for effective toponym matching. *Int. J. Digit. Earth* **11**, 1–26 (2017). <https://doi.org/10.1080/17538947.2017.1371253>
15. Sehgal, V., Getoor, L., Viechnicki, P.D.: Entity resolution in geospatial data integration. In: *Proceedings of the 14th Annual ACM International Symposium on Advances in Geographic Information Systems, GIS 2006*, pp. 83–90. ACM, New York (2006). <https://doi.org/10.1145/1183471.1183486>, <http://doi.acm.org/10.1145/1183471.1183486>
16. Zheng, Y., Fen, X., Xie, X., Peng, S., Fu, J.: Detecting nearly duplicated records in location datasets. In: *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems, GIS 2010*, pp. 137–143. ACM, New York (2010). <https://doi.org/10.1145/1869790.1869812>, <http://doi.acm.org/10.1145/1869790.1869812>