Metric Learning for Names

Anonymous ACL submission

Abstract

Many systems rely on the ability to effectively search through databases of personal names. Despite this, techniques for indexing names have yet to leverage recent advances in neural networks, hindering the throughput of these systems due to inefficient indexing. In this work, we present a method for fine-tuning a neural network via metric learning to embed personal names in a vector space which can be used for retrieval. We not only demonstrate up to a 12% improvement over existing systems, but also show that our model significantly outperforms them even in the absence of an external name transliteration engine, which is often required for existing indexing techniques.

1 Introduction

Automated matching of names is a problem with broad use cases, including compliance with financial regulations and medical record linkage. While much of the research in this area focuses on models which can assess whether a given pair of names matches one another, there is comparably less focus on the issue of retrieving a subset of a database in order to perform these comparisons efficiently. The key to this problem is the ability to hash personal names in such a way that two variants of the same name (for example, “John Smith” and “Jon Smith”) result in similar hashes. Variations can range from typographic errors to missing components (“Jon”) to a change of writing script (“جون” [*jun*]), so specialized indexing methods which understand the structure of names are essential to good performance. In this work, we explore how neural networks can be used in order to represent names as vectors such that these vectorized representations can be used for database retrieval.

Our contributions are as follows: (1) we present a method for fine-tuning a pretrained neural network to produce high-quality vectorized representations of names; (2) we demonstrate that this method works effectively even in multilingual contexts, meaning that, unlike most existing techniques, an external transliteration engine is not required; and (3) we reorganize and release the JRC-Names dataset (Steinberger et al., 2011) so it can be used to evaluate multilingual name indexing.

2 Related Work

As the primary use case for name vectorization is efficient database retrieval, the most relevant area of research to this effort is that of *candidate generation*. This is a sub-task of named entity linking which generates possible matching knowledge base entities from an input. For example, given the string “New York," a candidate generator (based on a general knowledge base such as Wikipedia) would produce candidates such as New York City, New York State, the New York Yankees, etc.

An overview of approaches to candidate generation is provided in Shen et al. (2015) and summarized here. Research in this area typically focuses on a single-script use case, often relying on dictionary-based approaches (i.e. looking up substrings in a dictionary of known entity aliases). This has the shortcoming of being incapable of dealing with misspellings. Proposed solutions to this issue include the use of the metaphone algorithm (Deorowicz and Ciura, 2005; Varma et al., 2008) and edit distance-based tools such as Lucene’s fuzzy query mechanism (Chen et al., 2010).

Also noteworthy is research in the area of named entity transliteration. Work such as Khakhmovich et al. (2020) and Merhav and Ash (2018) focus on this task, providing datasets mined from Wikipedia. A key distinction between these works and ours is our focus on producing an indexable representation. In contrast, Khakhmovich et al. (2020) performs an entity search procedure by producing probable transliterations and querying a traditional (edit distance-based) fuzzy index with them.

Our work relies upon prior research in the area of

$$\mathcal{L}_{contrast}(n_a, n) = \frac{\mathbb{I}(n_a, n) \cdot \delta(n_a, n)^2 + (1 - \mathbb{I}(n_a, n)) \left[m - \delta(n_a, n)^2\right]^+}{2} \quad (1)$$

$$\mathcal{L}_{triplet}(n_a, n_+, n_-) = \left[\delta(n_a, n_+)^2 - \delta(n_a, n_-)^2 + m\right]^+ \quad (2)$$

Figure 1: Loss functions used in our experiments.

fine-tuning Transformers (Vaswani et al., 2017) using contrastive loss. Existing research in this space (Ni et al., 2022; Gao et al., 2021) aims to embed semantically related sentences close to one another. Unlike these approaches, our work (a) aims to optimize embeddings for transliterative similarity and (b) relies upon hard negative mining techniques previously utilized in the context of entity normalization (Fakhraei et al., 2020).

Finally, we note existing datasets related to this task. Merhav and Ash (2018) publishes a list of name transliterations mined from Wikipedia, but it is not ideal for evaluating retrieval, as there appears to be only one transliteration pair per entity in the dataset. In contrast, JRC-Names (Steinberger et al., 2011) is a highly multilingual list of entity name variations which have been collected from the European News Monitor[1]. This dataset is intended to support name retrieval, and we have published a version of it designed for evaluation (Section 4).

3 Methodology

Our primary objective is to obtain a neural network which can read a name and produce a vector, such that the vector representations of two variants of the same name are similar to one another. To this end, we use a pretrained ByT5 model (Xue et al., 2022), based on Transformers (Vaswani et al., 2017), in order to encode names. ByT5 was chosen due to its ability to handle text in many different writing scripts; in our evaluation, we focus not only on Latin-script names, but also datasets of names written in multiple scripts (Section 4).

To do the actual fine-tuning, we utilize different standard methods for metric learning with siamese neural networks, including contrastive loss (Chopra et al., 2005) and triplet loss (Schroff et al., 2015). While these loss metrics were presented in the context of learning a metric for face similarity, the same metrics have recently found applications in natural language processing (Ni et al., 2022; Gao et al., 2021). To illustrate our use case, suppose that (n_a, n_+, n_-) is a triplet of names, where n_a and n_+ are variants of the same name (e.g. "John H. Smith" and "Jon Smith"), and n_- is not a variant of the same name as n_a (e.g. "Sue Kim"). Let $\delta(a, b)$ denote the euclidean vector distance between the names a and b, as encoded by our encoder. Finally, let $\mathbb{I}(a, b)$ be equal to one if and only if a and b are variants of the same name (and otherwise be equal to zero), let m denote a margin hyperparameter, and let $[x]^+ = \max(0, x)$. We can then define the various losses as shown in Figure 1.

Because our name matching dataset consists of groups of name variants which do match one another (as is the case in face matching), in order to utilize these losses, we must have some strategy for selecting negative examples (n_-) given a name n_a. A uniform sampling procedure is not effective, as it is far more likely to select a negative example which is not particularly informative; intuitively, given a n_a of "John Smith", we would prefer that our sampling procedure pick "Jon Doe" over "Bill Nye." To this end, we utilize a hard negative mining strategy similar to that which is used in the NSEEN entity normalization system (Fakhraei et al., 2019), where, after each training epoch, we construct an approximate nearest neighbor index of the neural network-encoded dataset names using Annoy (Bernhardsson, 2018). Then, for each query term n_a, we find its nearest neighbor in the index, excluding any names which are variations of the same entity (i.e. positive-matching). We then add this query-negative or query-positive-negative tuple to our training dataset.

In the end, we fine-tune using a hybrid of these two loss functions. To bootstrap reasonable embeddings, we perform a short pretraining epoch using contrastive loss ($\mathcal{L}_{contrast}$). Then, for subsequent epochs, we perform hard negative mining to construct triples to be used in a triplet loss ($\mathcal{L}_{triplet}$). More details about our training are in Appendix B.

4 Results

Our primary focus in developing this system was to produce embeddings which are optimized for

[1] https://knowledge4policy.ec.europa.eu/online-resource/europe-media-monitor-emm_en

Algorithm	MAP	Recall@1	Recall@5	Recall@10	Recall@50	Recall@100
Double Metaphone	0.16427	0.08546	0.16556	0.16968	0.17626	0.18214
+ *pre-transliteration*	0.91355	0.50809	0.89106	0.94381	0.97900	0.98530
Sentence-T5	0.29894	0.17942	0.30980	0.32471	0.35936	0.37559
+ *pre-transliteration*	0.84549	0.48011	0.83323	0.88937	0.94981	0.96674
ByT5	0.06841	0.04018	0.07354	0.09702	0.15778	0.19513
+ *pre-transliteration*	0.23765	0.17140	0.24862	0.28274	0.38930	0.44801
SimCSE	0.22508	0.11528	0.24852	0.29057	0.36891	0.39210
+ *pre-transliteration*	0.74304	0.42829	0.74217	0.80665	0.88724	0.91324
Lucene FuzzyQuery	0.12768	0.08876	0.14699	0.15542	0.16710	0.17427
+ *pre-transliteration*	0.27414	0.17101	0.29490	0.31799	0.34999	0.35650
Ours (La-only)	0.49188	0.28870	0.48443	0.51585	0.61112	0.67194
+ *pre-transliteration*	0.93797	**0.51632**	0.91437	0.95629	0.97745	0.98406
Ours (La+Ar+Cy)	**0.94458**	0.51450	**0.92528**	**0.96840**	**0.98660**	**0.99053**
+ *pre-transliteration*	0.93899	0.51601	0.91361	0.95732	0.98307	0.98903

Table 1: Results from a single run on the JRC-Names dataset, including Latin, Cyrillic, and Arabic names. "Ours (La-only)" indicates our model trained only on Latin-script names, and "Ours (La+Ar+Cy)" is our model trained on Latin, Arabic, and Cyrillic names. The highest score in each column is in bold. "+ pre-transliteration" indicates that all names were transliterated into Latin before evaluating.

database retrieval, meaning that a query name can find its matching name variations in a database using k-nearest neighbors. Thus, the primary metric of interest is recall@k, for various values of k (specifically, $k = 1, 5, 10, 50, 100$). We also measure the mean averaged precision (MAP), so as to compute a single metric of embedding quality.

In order to evaluate our system, we compare it against a variety of baselines. To most closely reflect "traditional" approaches to phonetic indexing, we take two approaches. First, we process each name into their approximate pronunciations with the Double Metaphone algorithm (Philips, 2000), and then create one-hot vectors using the bigrams of these pronunciation strings, which are then used for indexing and retrieval via cosine similarity. Second, we index all names with Lucene (Foundation, 2022) and perform retrieval with Lucene FuzzyQuery instances, which score matched items in the database using Damerau-Levenshtein edit distance (Damerau, 1964; Levenshtein et al., 1966).

Then, to compare our system against deep learning-based baselines, we focus on three systems: ByT5 (Xue et al., 2022) (without any fine-tuning by our procedure), SimCSE (Gao et al., 2021), and Sentence-T5 (Ni et al., 2022). The latter two systems are natural comparisons, as they are similarly tuned using a contrastive loss objective with the goal of embedding semantically related sentences close to one another. We expected that this would result in embeddings which outperform the non-fine-tuned baselines, but not outperform our technique due to it being specifically tuned for this task. The specific HuggingFace (Wolf et al., 2020) checkpoints used by our baselines are listed in Appendix A. Note that the ByT5 checkpoint used in the baseline is the same as what is used during the training of our algorithm.

We measure performance on a retrieval dataset produced from the JRC-Names dataset (Steinberger et al., 2011), which consists of clusters of variations of entity (person and organization) names collected from the European News Monitor. We take these clusters and convert them into query-result pairs, which can then be used to measure our evaluation metrics. More precisely, we filter the JRC-Names dataset to include only person names, and partition it into a large training split of 1,178,357 names (trained using the procedure described in Section 3) and a retrieval test set consisting of 1,886 queries against a database of 5,602 names (with each query having an average of 2.9 matching database elements). We include this data with this paper; further details on this split and the CJK (Chinese characters) split (see below) are in Appendix C.

The results of our algorithm and the various baselines are shown in Table 1. We measure two versions of our system: one trained on only Latin-script names, and one trained on names written in Latin, Arabic, and Cyrillic. Because many of

Algorithm	MAP	Recall@1	Recall@5	Recall@10	Recall@50	Recall@100
Double Metaphone	0.31134	0.10988	0.30951	0.31853	0.33370	0.33619
+ *pre-transliteration*	0.74852	0.17531	0.65205	0.77245	0.87325	0.91044
Ours (La+Ar+Cy)	0.43961	0.17294	0.43333	0.44496	0.46436	0.48065
+ *pre-transliteration*	0.73173	0.17643	0.64073	0.73700	0.84378	0.88488
Ours (Latin+CJK)	**0.87685**	0.18289	**0.76206**	**0.88109**	**0.93519**	**0.95105**
+ *pre-transliteration*	0.82742	**0.18507**	0.72232	0.82854	0.90130	0.92723

Table 2: Results from a single run on the JRC-Names dataset, including Latin and CJK names. "Ours (La+Ar+Cy)" indicates our model trained on Latin, Arabic, and Cyrillic names, and "Ours (Latin+CJK)" is our model trained on Latin and CJK names. The highest score in each column is in bold. "+ pre-transliteration" indicates that all names were transliterated into Latin before evaluating. Lower-scoring baselines are omitted.

our baselines (particularly double metaphone and Lucene) can only be fairly evaluated in a Latin-script context, we measure each system on both the unadulterated multiple-script query-result sets in addition to a version of the dataset which has been processed using an enterprise name transliterator. This transliteration engine utilizes standard rules (consisting of a lookup table from characters in one script to equivalent ones in the Latin alphabet) in use commercially, thereby reflecting a realistic example of how these algorithms might be used on this data in the real world.

We note two takeaways from the data presented in Table 1. First, our algorithm outperforms the baselines in both MAP and recall@k across the board. Second, the best-performing algorithm requires no transliteration engine, which is required for competitive performance in each baseline. However, we do note that the gap in performance between double metaphone and our algorithm is small when a transliteration engine is available. We hypothesize that this is due to the engine used in our experiments being particularly well-equipped to transliterate names from Arabic and Cyrillic scripts into Latin. Consequently, we created a separate training and testing dataset consisting of Latin and CJK names and re-ran our experiments (shown as "Latin+CJK"), as the transliteration engine we used is known to perform comparably less well on CJK names. The results are presented in Table 2, which indeed indicate a much more significant improvement over the double metaphone baseline.

4.1 Qualitative Analysis

We sought to obtain an intuition for when our algorithm was better or worse than the baseline systems. To this end, we analyzed the results of each algorithm and filtered the outputs to include those which were significantly better than the baselines and those which were significantly worse.

It is difficult to draw many conclusions about where specifically our algorithm outperforms the baselines, but one situation did stand out: our system can more effectively handle name pairs which have been transliterated using different conventions. For example, given the query-result pair ("Valentina Vladimirovna Tereškova", "Walentina Wladimirowna Tereschkowa"), our system is much more effectively able to recognize that the letter "w" in the result is pronounced the same as the letter "v" in the query.

Regarding names on which our system did worse, we find two broad categories: (1) transliteration errors/noise, and (2) crosslingual personal titles. An example of the latter would be the query-result pair ("Francis I of France", "فرانسوا الأول" [*fransuu al'awal*]), in which the *al'awal* suffix is the Arabic version of the *I* suffix in Latin-based languages.

5 Discussion

Our results demonstrate that our system is capable of indexing personal names more effectively than existing baselines. Moreover, it is able to do so in a way which can flexibly deal with names written in different alphabets, meaning that database retrieval can be effectively supported without requiring external methods for transliterating names into a single, shared script.

Future directions for this work include extending it to include more scripts in the shared embedding space, adding the ability to encode non-personal names (e.g. organizations and locations), and finding ways of leveraging the structure intrinsic to personal names (i.e. decomposing names into components such as first name, last name, and title) in order produce higher quality embeddings.

5.1 Risks and Limitations

The primary ethical consideration of this work is that the model it presents is based on a pretrained ByT5 model, which was trained on a large body of text collected from the internet. Consequently, the outputs and performance of our neural network may reflect the biases present in the original ByT5 training data (such as performance on a specific name origin or domain to which the name(s) is related).

There are two primary limitations of the system presented in this work. First, it is only able to effectively deal with names written in scripts which it was exposed to during training, meaning that inputs written in other scripts may yield unreliable results. Second, the quality of embeddings is also related to how closely input names reflect the cultural background of names to which the system was exposed during training. For example, a system which was not trained on personal names originating from Eastern Europe would not be able to properly retrieve names as shown in Sec. 4.1.

Acknowledgements

(Hidden during review)

References

Erik Bernhardsson. 2018. *Annoy: Approximate Nearest Neighbors in C++/Python*. Python package version 1.13.0.

Zheng Chen, Suzanne R. Tamang, Adam Lee, Xiang Li, Wen-Pin Lin, Matthew G. Snover, Javier Artiles, Marissa Passantino, and Heng Ji. 2010. Cunyblender tac-kbp2010 entity linking and slot filling system description. *Theory and Applications of Categories*.

S. Chopra, R. Hadsell, and Y. LeCun. 2005. Learning a similarity metric discriminatively, with application to face verification. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 539–546 vol. 1.

Fred J. Damerau. 1964. A technique for computer detection and correction of spelling errors. *Commun. ACM*, 7(3):171–176.

Sebastian Deorowicz and Marcin Ciura. 2005. Correcting spelling errors by modelling their causes. *International Journal of Applied Mathematics and Computer Science*, 15:275–285.

Shobeir Fakhraei, Joel Mathew, and José Luis Ambite. 2019. Nseen: Neural semantic embedding for entity normalization. In *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part II*, page 665–680, Berlin, Heidelberg. Springer-Verlag.

Shobeir Fakhraei, Joel Mathew, and José Luis Ambite. 2020. Nseen: Neural semantic embedding for entity normalization. In *Machine Learning and Knowledge Discovery in Databases*, pages 665–680, Cham. Springer International Publishing.

Apache Software Foundation. 2022. *Apache Lucene*. Java version 9.4.1.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. SimCSE: Simple contrastive learning of sentence embeddings. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Timnit Gebru, Jamie Morgenstern, Briana Vecchione, Jennifer Wortman Vaughan, Hanna Wallach, Hal Daumé Iii, and Kate Crawford. 2021. Datasheets for datasets. *Communications of the ACM*, 64(12):86–92.

Aleksandr Khakhmovich, Svetlana Pavlova, Kira Kirillova, Nikolay Arefyev, and Ekaterina Savilova. 2020. Cross-lingual named entity list search via transliteration. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 4247–4255, Marseille, France. European Language Resources Association.

Vladimir I Levenshtein et al. 1966. Binary codes capable of correcting deletions, insertions, and reversals. In *Soviet physics doklady*, volume 10, pages 707–710. Soviet Union.

Yuval Merhav and Stephen Ash. 2018. Design challenges in named entity transliteration. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 630–640, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022. Sentence-t5: Scalable sentence encoders from pretrained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, Dublin, Ireland. Association for Computational Linguistics.

Lawrence Philips. 2000. The double metaphone search algorithm. *C/C++ Users J.*, 18(6):38–43.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. Facenet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Transactions on Knowledge and Data Engineering*, 27(2):443–460.

Ralf Steinberger, Bruno Pouliquen, Mijail Kabadjov, Jenya Belyaeva, and Erik van der Goot. 2011. JRC-NAMES: A freely available, highly multilingual named entity resource. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 104–110, Hissar, Bulgaria. Association for Computational Linguistics.

Vasudeva Varma, Prasad Pingali, Rahul Katragadda, Sai Krishna, Surya Ganesh Veeravalli, Kiran Sarvabhotla, Harish Garapati, Hareen Gopisetty, Vijay Bharath Reddy, B KranthiReddy, Praveen Bysani, and Rohit G. Bharadwaj. 2008. Iiit hyderabad at tac 2009. *Theory and Applications of Categories*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. ByT5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.

A HuggingFace Checkpoints

The checkpoints used in the baselines (and, in the case of ByT5, our algorithm) are listed in Table 3.

B Training Details

Our fine tuning experiments were done using the Adafactor optimizer (Shazeer and Stern, 2018) with the recommended parameters of $\epsilon_1 = 10^{-30}$, $\epsilon_2 = 10^{-3}$, $d = 1$, $\rho_t = \min(10^{-2}, \frac{1}{\sqrt{t}})$, and $1 - t^{-0.8}$. We used an NVIDIA A100 GPU on a Google Cloud Platform `a2-highgpu-1g` instance for our experiments, and our fine-tuning procedure took about 24 hours to run.

C Name Dataset Datasheet

This datasheet template is taken from Gebru et al. (2021).

Motivation

For what purpose was the dataset created? *Was there a specific task in mind? Was there a specific gap that needed to be filled? Please provide a description.*

The goal was to create a multilingual personal name retrieval dataset.

Who created this dataset (e.g., which team, research group) and on behalf of which entity (e.g., company, institution, organization)?
The European Union Joint Research Centre produced the JRC-Names dataset, and the published splits were produced by (Hidden during review).

Who funded the creation of the dataset? *If there is an associated grant, please provide the name of the grantor and the grant name and number.*

The European Union funded the creation of the JRC-Names dataset.

Any other comments?

Composition

What do the instances that comprise the dataset represent (e.g., documents, photos, people, countries)? *Are there multiple types of instances (e.g., movies, users, and ratings; people and interactions between them; nodes and edges)? Please provide a description.*

Each split of the dataset consists of a list of names (for training), grouped by which entity the names are a variant of (see the description of the JRC-Names dataset in Steinberger et al. (2011)), and database/query/expected files for evaluation. The structure of those files is explained below, and a script is provided to demonstrate loading them.

How many instances are there in total (of each type, if appropriate)?
The Latin/Arabic/Cyrillic training data consists of 1,178,357 names, grouped into clusters of an average size of 1.59 names.The following is the size breakdown of the Latin/Arabic/Cyrillic data is shown in Table 4.

The Latin/CJK training data consists of 97,077 names, grouped into clusters of an average size of

Algorithm	Checkpoint
Sentence-T5	`sentence-transformers/sentence-t5-base`
SimCSE	`princeton-nlp/sup-simcse-roberta-base`
ByT5	`google/byt5-base`

Table 3: HuggingFace checkpoints used for baselines.

Database Rows	5,602
Queries	1,885
Avg. Expected per Query	2.97

Table 4: Latin/Arabic/Cyrillic data split breakdown.

2.53 names, and the breakdown for the Latin/CJK data is shown in Table 5.

Database Rows	1,457
Queries	268
Avg. Expected per Query	5.43

Table 5: Latin/CJK data split breakdown.

Does the dataset contain all possible instances or is it a sample (not necessarily random) of instances from a larger set? *If the dataset is a sample, then what is the larger set? Is the sample representative of the larger set (e.g., geographic coverage)? If so, please describe how this representativeness was validated/verified. If it is not representative of the larger set, please describe why not (e.g., to cover a more diverse range of instances, because instances were withheld or unavailable).*

No. This dataset is a downsampling of the JRC-Names dataset. The sampling was done in a way to bias towards entities who have names written in multiple writing scripts, and the script coverage statistics were validated after the downsampling procedure.

What data does each instance consist of? "Raw" data (e.g., unprocessed text or images) or features? *In either case, please provide a description.*

The dataset consists of four files per split. The training data is in the same format as the JRC-Names dataset: a list of names with entity identifiers which can be used to associate name variations for the same entity with one another. For details, readers are referred to Steinberger et al. (2011).

The remaining files are for the evaluation dataset: a *database file*, consisting of a list of names (or "rows" in a database to be queried), a *query file*, consisting of a list of names which are each search queries intended to be run against the database, and an *expected file*, consisting of the database rows which each query is intended to match. Note that the dataset includes a Python script which demonstrates how to load the data.

For example, the first query of the Latin/Arabic/Cyrillic dataset is "Sergey Polonski", which is expected to match the following entries in the database: "Sergey Polonsky", "Сергей Полонски" (*Sergey Polonski*), and "Сергей Полонский" (*Sergey Polonskiy*).

Is there a label or target associated with each instance? *If so, please provide a description.*

Each query has a set of one or more expected rows in the database file which that query is intended to match.

Is any information missing from individual instances? *If so, please provide a description, explaining why this information is missing (e.g., because it was unavailable). This does not include intentionally removed information, but might include, e.g., redacted text.*

No.

Are relationships between individual instances made explicit (e.g., users' movie ratings, social network links)? *If so, please describe how these relationships are made explicit.*

N/A.

Are there recommended data splits (e.g., training, development/validation, testing)? *If so, please provide a description of these splits, explaining the rationale behind them.*

Yes. The published dataset includes training and testing splits for Latin/Arabic/Cyrillic and Latin/CJK scripts.

Are there any errors, sources of noise, or redundancies in the dataset? *If so, please provide a description.*

Not to our knowledge. We note that the dataset deliberately has specific sorts of noise (e.g. variations in spellings for a person's name).

Is the dataset self-contained, or does it link to or otherwise rely on external resources (e.g., websites, tweets, other datasets)? *If it links to or relies on external resources, a) are there guarantees that they will exist, and remain constant, over time; b) are there official archival versions of the complete dataset (i.e., including the external resources as they existed at the time the dataset was created); c) are there any restrictions (e.g., licenses, fees) associated with any of the external resources that might apply to a future user? Please provide descriptions of all external resources and any restrictions associated with them, as well as links or other access points, as appropriate.*

The dataset is self-contained.

Does the dataset contain data that might be considered confidential (e.g., data that is protected by legal privilege or by doctor-patient confidentiality, data that includes the content of individuals non-public communications)? *If so, please provide a description.*

No.

Does the dataset contain data that, if viewed directly, might be offensive, insulting, threatening, or might otherwise cause anxiety? *If so, please describe why.*

No.

Does the dataset relate to people? *If not, you may skip the remaining questions in this section.*

Yes; it is a list of peoples' names.

Does the dataset identify any subpopulations (e.g., by age, gender)? *If so, please describe how these subpopulations are identified and provide a description of their respective distributions within the dataset.*

No.

Is it possible to identify individuals (i.e., one or more natural persons), either directly or indirectly (i.e., in combination with other data) from the dataset? *If so, please describe how.*

Yes. The dataset is a list of names of people who have appeared in news articles, so these individuals would be identifiable if their name is unique.

Does the dataset contain data that might be considered sensitive in any way (e.g., data that reveals racial or ethnic origins, sexual orientations, religious beliefs, political opinions or union memberships, or locations; financial or health data; biometric or genetic data; forms of government identification, such as social security numbers; criminal history)? *If so, please provide a description.*

No.

Any other comments?

Collection Process

How was the data associated with each instance acquired? *Was the data directly observable (e.g., raw text, movie ratings), reported by subjects (e.g., survey responses), or indirectly inferred/derived from other data (e.g., part-of-speech tags, model-based guesses for age or language)? If data was reported by subjects or indirectly inferred/derived from other data, was the data validated/verified? If so, please describe how.*

The original JRC-Names data (Steinberger et al., 2011) was collected from the European News Monitor.

What mechanisms or procedures were used to collect the data (e.g., hardware apparatus or sensor, manual human curation, software program, software API)? *How were these mechanisms or procedures validated?*

The JRC-Names data was automatically collected (Steinberger et al., 2011), and the splits we release were programmatically downsampled from the JRC-Names data.

If the dataset is a sample from a larger set, what was the sampling strategy (e.g., deterministic, probabilistic with specific sampling probabilities)?

The splits we have released are downsampled from the full JRC-Names dataset. After filtering by script type, the entity clusters are downsampled non-uniformly, with a bias towards entity clusters containing entity names written in different writing scripts.

Who was involved in the data collection process (e.g., students, crowdworkers, contractors) and how were they compensated (e.g., how much were crowdworkers paid)?

The European Union Joint Research Centre collected the names. Detailed information on the collectors was not provided by the authors.

Over what timeframe was the data collected? Does this timeframe match the creation timeframe of the data associated with the instances (e.g., recent crawl of old news articles)? *If not, please describe the timeframe in which the data associated with the instances was created.*

The data in JRC-Names has been collected since 2004 through the present.

Were any ethical review processes conducted (e.g., by an institutional review board)? *If so, please provide a description of these review processes, including the outcomes, as well as a link or other access point to any supporting documentation.*

No. We only reorganized an existing public dataset, and we are not aware if an institutional review board was involved with the publication of the original JRC-Names dataset.

Does the dataset relate to people? *If not, you may skip the remaining questions in this section.*

Yes; it is a list of names of people found in news articles.

Did you collect the data from the individuals in question directly, or obtain it via third parties or other sources (e.g., websites)?

It was obtained via third parties (public news websites).

Were the individuals in question notified about the data collection? *If so, please describe (or show with screenshots or other information) how notice was provided, and provide a link or other access point to, or otherwise reproduce, the exact language of the notification itself.*

No.

Did the individuals in question consent to the collection and use of their data? *If so, please describe (or show with screenshots or other information) how consent was requested and provided, and provide a link or other access point to, or otherwise reproduce, the exact language to which the individuals consented.*

No.

If consent was obtained, were the consenting individuals provided with a mechanism to revoke their consent in the future or for certain uses? *If so, please provide a description, as well as a link or other access point to the mechanism (if appropriate).*

N/A

Has an analysis of the potential impact of the dataset and its use on data subjects (e.g., a data protection impact analysis) been conducted? *If so, please provide a description of this analysis, including the outcomes, as well as a link or other access point to any supporting documentation.*

N/A

Any other comments?

Preprocessing/cleaning/labeling

Was any preprocessing/cleaning/labeling of the data done (e.g., discretization or bucketing, tokenization, part-of-speech tagging, SIFT feature extraction, removal of instances, processing of missing values)? *If so, please provide a description. If not, you may skip the remainder of the questions in this section.*

The JRC-Names data was collated into script-specific splits using scripts provided in the dataset.

Was the "raw" data saved in addition to the preprocessed/cleaned/labeled data (e.g., to support unanticipated future uses)? *If so, please provide a link or other access point to the "raw" data.*

The "raw" JRC-Names data was not saved by the authors, but it is available at the following URL: https://joint-research-centre.ec.europa.eu/language-technology-resources/jrc-names_en

Is the software used to preprocess/clean/label the instances available? *If so, please provide a link or other access point.*

Yes, it is included with the dataset.

Any other comments?

Uses

Has the dataset been used for any tasks already? *If so, please provide a description.*

Yes, this paper.

Is there a repository that links to any or all papers or systems that use the dataset? *If so, please provide a link or other access point.*

(Hidden during review)

What (other) tasks could the dataset be used for?

Transliteration (of names or other text).

Is there anything about the composition of the dataset or the way it was collected and preprocessed/cleaned/labeled that might impact future uses? *For example, is there anything that a future user might need to know to avoid uses that could result in unfair treatment of individuals or groups (e.g., stereotyping, quality of service issues) or other undesirable harms (e.g., financial harms, legal risks) If so, please provide a description. Is there anything a future user could do to mitigate these undesirable harms?*

Not to our knowledge.

Are there tasks for which the dataset should not be used? *If so, please provide a description.*

No.

Any other comments?

Distribution

Will the dataset be distributed to third parties outside of the entity (e.g., company, institution, organization) on behalf of which the dataset was created? *If so, please provide a description.*

Yes. The data shall be publicly released alongside this paper.

How will the dataset will be distributed (e.g., tarball on website, API, GitHub) *Does the dataset have a digital object identifier (DOI)?*

The dataset is available for download on GitHub at (Hidden during review).

When will the dataset be distributed?

It is already distributed.

Will the dataset be distributed under a copyright or other intellectual property (IP) license, and/or under applicable terms of use (ToU)? *If so, please describe this license and/or ToU, and provide a link or other access point to, or otherwise reproduce, any relevant licensing terms or ToU, as well as any fees associated with these restrictions.*

The dataset is available under (Hidden during review).

Have any third parties imposed IP-based or other restrictions on the data associated with the instances? *If so, please describe these restrictions, and provide a link or other access point to, or otherwise reproduce, any relevant licensing terms, as well as any fees associated with these restrictions.*

The original JRC-Names dataset was released under an EULA specified here: `https://wt-public.emm4u.eu/Resources/LICENCE-EULA_JRC-Names_2011.pdf`.

Do any export controls or other regulatory restrictions apply to the dataset or to individual instances? *If so, please describe these restrictions, and provide a link or other access point to, or otherwise reproduce, any supporting documentation.*

No.

Any other comments?

Maintenance

Who will be supporting/hosting/maintaining the dataset?

(Hidden during review) will be supporting this dataset.

How can the owner/curator/manager of the dataset be contacted (e.g., email address)?

(Hidden during review)

Is there an erratum? *If so, please provide a link or other access point.*

No.

Will the dataset be updated (e.g., to correct labeling errors, add new instances, delete instances)? *If so, please describe how often, by whom, and how updates will be communicated to users (e.g., mailing list, GitHub)?*

No.

If the dataset relates to people, are there applicable limits on the retention of the data associated with the instances (e.g., were individuals in question told that their data would be retained for a fixed period of time and then deleted)? *If so, please describe these limits and explain how they will be enforced.*

N/A

Will older versions of the dataset continue to be supported/hosted/maintained? *If so, please describe how. If not, please describe how its obsolescence will be communicated to users.*

N/A

If others want to extend/augment/build on/contribute to the dataset, is there a mechanism for them to do so? *If so, please provide a description. Will these contributions be validated/verified? If so, please describe how. If not, why not? Is there a process for communicating/distributing these contributions to other users? If so, please provide a description.*

Contributors are welcome to make pull requests against the dataset repository on GitHub containing new splits of the JRC-Names data. Before acceptance, the maintainers shall verify that the submitted names are indeed included in the JRC-Names dataset.

Any other comments?