JRC-Names-Retrieval: A Standardized Benchmark for Name Search

Philip Blair, Kfir Bar
Babel Street
1818 Library Street, Reston, VA 20190, USA
{pblair, kbar}@babelstreet.com

Abstract

Many systems rely on the ability to effectively search through databases of personal and organization entity names in multiple writing scripts. Despite this, there is a relative lack of research studying this problem in isolation. In this work, we discuss this problem in detail and support future research by publishing what we believe is the first comprehensive dataset designed for this task. Additionally, we present a number of baselines against which future work can be compared; among which, we describe a neural solution based on ByT5 (Xue et al., 2022) which demonstrates up to a 12% performance gain over preexisting baselines, indicating that there remains much room for improvement in this space.

Keywords: Corpus, Information Retrieval, Multilinguality, Person Identification

1. Introduction

Automated matching of personal and organization 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. Both personal and organization names pose unique challenges not found in other information retrieval domains due to a variety of onomastic-specific phenomena and the requirement to consider both phonetic and semantic information.

The key to this problem is the ability to hash 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 (“Jon”) to missing components (“Smith”) to a change of writing script (“جون” [*jun*]), so specialized indexing methods which understand the structure of names are essential to good performance. Research in this area is hindered by a lack of both problem definition and consistent benchmarks. In this work, we describe the tasks of name matching and name retrieval in detail and attempt to alleviate this issue.

Our contributions are as follows:

- We reorganize the JRC-Names dataset (Steinberger et al., 2011) into various multi-script splits so that it can be used to evaluate and compare cross-script name indexing systems (Section 4). By defining a consistent query/result-list definition across the dataset, future systems will be able to compare against a uniform baseline; to our knowledge, this is the first dataset for multilingual name retrieval which has been published.
- We present a number of baseline scores for this task, including fine-tuning a pretrained neural network to emit high-quality vectorized representations of names (Sections 5 and 6).
- We demonstrate a sizeable gap between this neural approach and other baselines, showing that there remains much room for improvement on this task (Section 6).

2. Related Work

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

LREC-COLING 2024, pages 9589–9603
20-25 May, 2024. © 2024 ELRA Language Resource Association: CC BY-NC 4.0

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 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., 2019).

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 by allowing standardization of names, and we have published a version of it designed for evaluating generic name retrieval (Section 6).

3. Problem Statement

Before discussing name *retrieval*, we first discuss name *matching*. For this task, we would like to determine how similar a pair of entity names (either personal names or organization names) are. In this context, two names are “similar” if, with no other lexical or real-world context, one can assume that they plausibly refer to the same entity. The lack of context distinguishes this problem from named entity linking (NEL), as we here are (a) being more permissive (by considering “John” and “John” to be a perfect match, whereas in an NEL setup they can’t be linked together without context) and (b) ignoring real-world nicknames, such as “Marshall Mathers” and “Eminem” (as one does not know that these refer to the same person without knowing who Eminem, specifically, is).

The messy nature of real world data often poses a number of challenges for name matching systems. For example, situations can feasibly arise such as a name being in a database in last name-first name order and a query being in first name-last name order *and* a different writing script. In general, a “fuzzy” approach is required, as these phenomena can be difficult to enumerate, often co-occur, and often result in name pairs which exhibit a degree of similarity rather than a simple binary yes-or-no match. Examples of phenomena that name matching systems should be able to cope with include:

- **Phonetically/Lexically Similar Names**, e.g. “John Smith” and “Jon Smith”, or “Phillippe Jones” and “Felipe Jones”.
- **Initials**, e.g. “George Walker Bush” and “George W. Bush”.
- **Missing Components**, e.g. “John A. Macdonald” and “John Macdonald”.
- **Similar Names/Nicknames**, e.g. “William Clinton” and “Bill Clinton”, or “Michael Scott” and “Prison Mike”.
- **Out-of-Order Components**, e.g. “François Mitterand” and “MITTERAND François”.
- **Titles**, e.g. “Sir Tony Blair” and “Tony Blair”.
- **Different Writing Scripts**, e.g. “Hu Jintao” and “胡锦涛”.

In addition to the above, organization names exhibit some unique challenges:

- **Semantically Similar Components,** e.g. “Raven Train Company” and “Raven Locomotive, Inc.”.
- **Semantically Similar Components in Different Languages,** e.g. “株式会社京都アニメーション” (*Kabushiki-gaisha Kyōto Animēshon*) and “Kyoto Animation Co. Ltd.”.

In this work, we opt to focus on the task of name *retrieval*, in which we assume that we have a query name that we are searching for in a database of names. This is because it is the scenario found in many important applications of name matching, such as know-your-customer (KYC) compliance (where one would like to determine if, say, the person opening an account at your bank is on a sanctions list). Additionally, as discussed further in Section 7, the dataset we produce in this paper does not exhibit all of these phenomena; noisier variations such as out-of-order components or missing components are either missing or underrepresented.

4. Dataset

The JRC-Names dataset (Steinberger et al., 2011) consists of clusters of variations of entity (person and organization) names collected from the European News Monitor. Because these variations

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

Type	Script	# Names	# Clusters
PER	Latn+Cyrl+Arab	1,178,357	737,361
PER	Latn+CJK	97,077	38,268
PER	Hang+Hebr	1,331	640
PER	Deva+Kana	1,792	454
ORG	Latn+Cyrl+Arab	30,113	2,942
ORG	Latn+CJK	27,702	2,857
ORG	Hang+Hebr	494	224
ORG	Deva+Kana	380	122

Table 1: JRC-Names-Retrieval training subsplit sizes for each entity type/script combination split.

come from news sources in a large variety of languages, they span over twenty writing scripts and even more languages. We have filtered these down and produced per-entity-type splits (one for persons, one for organizations) of the dataset across four different writing script combinations:

- Latin, Cyrillic, and Arabic (Latn+Cyrl+Arab)
- Latin and Hanzi (Latn+Hanzi)
- Hangul and Hebrew (Hang+Hebr)
- Devanagari, Katakana, and Hiragana (Deva+Kana)

More precisely, we filter the JRC-Names dataset to only include the relevant entity types and writing scripts for each split, and then partition each into training and evaluation sub-splits. The size of the training sub-splits are shown in Table 1. The training data is in the same format as the original JRC-Names dataset, consisting of a list of clusters of names (which are variations of one another). The splits' different sizes simulate high-, medium-, and low-resource scenarios. The specific combinations of writing scripts are useful in different ways: the Latn+Cyrl+Arab and Latn+Hanzi splits reflect common script combinations in real-world scenarios (whether due to the nature of specific applications or the fact that these are high-resource writing scripts). In contrast, the Hang+Hebr and Deva+Kana combinations are less typical, but they are intended to (a) provide resources which are not Latin-based and (b) provide "stress-test" scenarios for multi-script systems to be evaluated on less common script pairs.

Each split similarly has a corresponding evaluation subsplit, with their sizes shown in Table 2. The evaluation data is in a format conducive to evaluating retrieval; each subset consists of a database (a list of names), a list of queries, and the results expected to match each query. These query/result-list pairs were produced from the original JRC-Names clusters by randomly sampling a name from each cluster and designating it as the query. The final column of Table 2 indicates the average number of expected results for each query in the evaluation data.

Unfortunately, it is not possible to provide a definitive breakdown of the full spectrum of name variations found in this dataset, as doing so would require annotating all name pairs in the training clusters and evaluation query/result-list sets (which would be prohibitive). That said, we can offer some general observations:

- The personal names in the dataset are near-universally full names (first name and surname). There still may be missing/extra components in some names (e.g. “Víktor Kassai” and “Hakem Viktor Kassai”, where “Hakem” is the Turkish word for “referee”). Additionally, name components tend to be in the “standard” order for the given name’s origin, as one would find in news media (that is, given name-surname for western and Japanese names written in Latin script, surname-given name for Chinese names and Japanese names written in Kanji/Kana).
- Many variations are small changes due to either linguistic transliteration conventions (e.g. “Julio Cezaro” and “Julius Cæsar”) or occasional spelling/capitalization mistakes (e.g. “PLÁCIDO DOMINGO” and “Plácido Domingo”). The latter were not removed/deduplicated, as casing information is generally important for this task and we wanted to avoid removing higher-quality names in favor of lower-quality ones.
- There are a handful of names with initials (e.g. “Alfredo D. Stephano”).
- No instances of nicknames were found.

The full dataset is available on GitHub, along with example code showing how to load the data. Further details on this dataset are available in Appendix A.

5. Baselines

In order to support future research using this dataset, we include the performance of a number of baseline systems. First, we present here the preexisting baselines from other literature, and we then discuss a metric learning-based baseline developed for this work.

5.1. Preexisting Baselines

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

Entity Type	Script	Database Size	# Queries	Avg. # Results
PER	Latn+Cyrl+Arab	5602	1885	2.97
PER	Latn+CJK	1457	268	5.43
PER	Hang+Hebr	316	266	1.18
PER	Deva+Kana	550	160	3.43
ORG	Latn+Cyrl+Arab	841	152	5.53
ORG	Latn+CJK	738	135	5.51
ORG	Hang+Hebr	119	97	1.23
ORG	Deva+Kana	90	38	2.36

Table 2: JRC-Names-Retrieval evaluation subsplit sizes for each entity type/script combination split.

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, 1966).

Then, to compare our system against deep learning-based baselines, we focus on three systems which are often used in literature in order to embed text as a vector: **ByT5** (Xue et al., 2022) (without the fine-tuning procedure described in Section 5.2), **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 ByT5 baseline, but not in ones which 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 Table 3. Note that the ByT5 checkpoint used in the baseline is the same as what is used during the training of our algorithm.

5.2. Metric Learning Baseline

In order to have a baseline which is trained on the training data we provide, we additionally present results from a neural network which is directly trained to 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 fine-tune 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 (as it is token-free); in order to produce a single vector as output, we mean-pool the encoder's representations for each character, run it through a final linear layer, and take the L_2 norm.

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 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 min-

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.

$$\mathcal{L}_{contrast}(n_a, n_+) = \frac{\delta(n_a, n_+)^2}{2},\ \mathcal{L}_{contrast}(n_a, n_-) = \frac{\left[m - \delta(n_a, n_-)^2\right]^+}{2} \tag{1}$$

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

Figure 1: Loss functions used in our metric learning baseline.

ing to construct triples to be used in a triplet loss ($\mathcal{L}_{triplet}$). 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 for the largest split (Latin/Cyrillic/Arabic personal names).

6. Results

There are variations across the baselines, but the principle behind the evaluation is the same among them all: we retrieve the k top-scoring results from the database using the query name and would like the expected subset of names to be in that top-k list. In detail, we do the following for each algorithm:

- For Double Metaphone, we perform the procedure described in Section 5.1.
- For Lucene FuzzyQuery, we build a Lucene (Foundation, 2022) index using the names in the database and query each name using a FuzzyQuery instance. The top-k results are then returned by the internal Lucene edit distance algorithm.
- For the dense vector-based systems (all remaining baselines), we embed every name in the database using the neural network. We then embed the query name and take the top-k nearest neighbors.

As we are measuring retrieval, 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.

A practical issue faced during our evaluation is the fact that many of these algorithms are either only defined over the Latin alphabet (e.g. Double Metaphone) or are not intended to compare names from different writing scripts (e.g. ByT5, SimCSE, etc.). In order to fairly evaluate these baselines, we perform separate evaluations on all datasets using the Rosette enterprise name transliterator[2]. 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.

The results of our algorithm and the various baselines are shown in Tables 4 and 5. We measure two versions of our metric learning-based system: one trained on only Latin-script names, and one trained on names written in their original writing scripts. The error bars shown indicate the results of training the model three times with different random weight initializations and orderings of the dataset. Broadly speaking, we draw a distinction between two sets of results: the high-resource script groups (Latin/Cyrillic/Arabic and Latin/Hanzi) and the low-resource script groups (Hangul/Hebrew and Devanagari/Katakana/Hiragana). While we report all scores for the sake of completeness, as noted above, certain algorithm/script pair combinations are not expected to do well. These are marked in the result tables with a † symbol.

Among the high-resource script groups, we find two primary results. First, our triplet-based approach yields better performance than the other baselines in both MAP and recall@k across the board for personal names. Second, the best-performing algorithm requires no transliteration engine, which is required for competitive performance in each non-metric-learning-based baseline. That being said, 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

[2] The transliterated dataset is available upon request from the authors.

Latin, Cyrillic, and Arabic

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†	0.39133	0.24901	0.39832	0.41042	0.42456	0.428090
+ pre-transliteration	0.76186	0.44020	0.75026	0.80762	0.88085	0.89819
Ours (Latin-only)†	0.50 ± 0.01	0.289 ± 0.00	0.47 ± 0.02	0.50 ± 0.03	0.62 ± 0.01	0.66 ± 0.01
+ pre-transliteration	0.93 ± 0.01	0.513 ± 0.00	0.91 ± 0.01	0.95 ± 0.00	0.97 ± 0.00	0.98 ± 0.00
Ours (La+Ar+Cy)	**0.95 ± 0.01**	**0.521 ± 0.01**	**0.93 ± 0.00**	**0.97 ± 0.00**	**0.99 ± 0.00**	**0.99 ± 0.00**
+ pre-transliteration	0.94 ± 0.01	0.520 ± 0.00	0.92 ± 0.01	0.96 ± 0.01	0.98 ± 0.00	0.98 ± 0.00

Latin and Hanzi

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
Sentence-T5†	0.31130	0.11126	0.30871	0.31692	0.33371	0.34055
+ pre-transliteration	0.68389	0.17071	0.60647	0.69882	0.79372	0.83053
ByT5	0.07486	0.03837	0.07718	0.10056	0.18022	0.24148
+ pre-transliteration	0.15977	0.06965	0.14502	0.16835	0.27705	0.37376
SimCSE†	0.42119	0.17357	0.41617	0.43420	0.45417	0.46238
+ pre-transliteration	0.63187	0.16978	0.55939	0.64179	0.73881	0.78750
Lucene FuzzyQuery†	0.26278	0.09944	0.26380	0.27437	0.28613	0.29670
+ pre-transliteration	0.58183	0.15951	0.52562	0.60317	0.65018	0.66013
Ours (Latin-only)†	0.45 ± 0.00	0.17 ± 0.00	0.44 ± 0.00	0.45 ± 0.00	0.46 ± 0.00	0.48 ± 0.00
+ pre-transliteration	0.73 ± 0.03	0.17 ± 0.00	0.64 ± 0.03	0.73 ± 0.03	0.83 ± 0.03	0.86 ± 0.03
Ours (Latin+CJK)	**0.88 ± 0.01**	**0.183 ± 0.00**	**0.76 ± 0.00**	**0.88 ± 0.01**	**0.94 ± 0.01**	**0.95 ± 0.00**
+ pre-transliteration	0.82 ± 0.00	0.180 ± 0.00	0.72 ± 0.00	0.83 ± 0.00	0.90 ± 0.00	0.93 ± 0.00

Hangul and Hebrew

Algorithm	**MAP**	**Recall@1**	**Recall@5**	**Recall@10**	**Recall@50**	**Recall@100**
Double Metaphone†	0.00377	0.00376	0.01880	0.03195	0.11905	0.26880
+ pre-transliteration	**0.69898**	**0.57625**	**0.79286**	**0.85489**	**0.95263**	**0.96278**
Sentence-T5†	0.00942	0.00376	0.02519	0.03929	0.22876	0.43866
+ pre-transliteration	0.11118	0.05996	0.13227	0.20069	0.38929	0.49831
ByT5	0.02046	0.00376	0.01817	0.02644	0.12475	0.28703
+ pre-transliteration	0.01529	0.00000	0.01190	0.02444	0.10652	0.22400
SimCSE†	0.03592	0.02801	0.03001	0.03390	0.04091	0.04217
+ pre-transliteration	0.07182	0.04605	0.07061	0.08828	0.27400	0.39555
Lucene FuzzyQuery†	0.04103	0.03239	0.04035	0.04655	0.07550	0.104323
+ pre-transliteration	0.05835	0.03302	0.07932	0.09511	0.14285	0.14285
Ours (Latin-only)†	0.04 ± 0.00	0.03 ± 0.00	0.04 ± 0.00	0.04 ± 0.00	0.07 ± 0.02	0.17 ± 0.06
+ pre-transliteration	0.37 ± 0.10	0.26 ± 0.10	0.45 ± 0.10	0.53 ± 0.07	0.71 ± 0.07	0.78 ± 0.07
Ours (Hang+Hebr)	0.06 ± 0.00	0.03 ± 0.00	0.05 ± 0.01	0.09 ± 0.01	0.32 ± 0.04	0.52 ± 0.05
+ pre-transliteration†	0.05 ± 0.00	0.03 ± 0.00	0.05 ± 0.00	0.07 ± 0.01	0.13 ± 0.00	0.25 ± 0.03

Devanagari, Katakana, and Hiragana

Algorithm	**MAP**	**Recall@1**	**Recall@5**	**Recall@10**	**Recall@50**	**Recall@100**
Double Metaphone†	0.00625	0.00104	0.00521	0.01125	0.07292	0.15208
+ pre-transliteration	**0.78880**	**0.26271**	**0.77385**	**0.83448**	**0.93417**	**0.94948**
Sentence-T5†	0.01867	0.00698	0.01958	0.02781	0.09500	0.18792
+ pre-transliteration	0.59969	0.21792	0.58042	0.65792	0.82323	0.90031
ByT5	0.12211	0.05271	0.12635	0.16302	0.31656	0.42073
+ pre-transliteration	0.23801	0.12719	0.23385	0.25833	0.37771	0.49510
SimCSE†	0.47849	0.22990	0.47542	0.47646	0.47750	0.47750
+ pre-transliteration	0.53141	0.22156	0.49750	0.55833	0.73000	0.80083
Lucene FuzzyQuery†	0.18886	0.15729	0.18791	0.19000	0.20041	0.217500
+ pre-transliteration	0.48090	0.20958	0.47031	0.51750	0.55114	0.561563
Ours (Latin-only)†	0.47 ± 0.00	0.23 ± 0.00	0.47 ± 0.00	0.48 ± 0.00	0.48 ± 0.00	0.49 ± 0.00
+ pre-transliteration	0.69 ± 0.02	0.24 ± 0.01	0.67 ± 0.03	0.75 ± 0.02	0.87 ± 0.01	0.91 ± 0.00
Ours (Deva+Kana)	0.43 ± 0.01	0.21 ± 0.00	0.44 ± 0.00	0.45 ± 0.00	0.47 ± 0.00	0.47 ± 0.00
+ pre-transliteration†	0.35 ± 0.00	0.18 ± 0.00	0.34 ± 0.00	0.37 ± 0.01	0.51 ± 0.00	0.59 ± 0.00

Table 4: Results from three runs on the JRC-Names-Retrieval person datasets. The scripts after "Ours" indicates which scripts were used to train the models. The highest score in each column is in bold. "+ pre-transliteration" indicates that all names were transliterated into Latin before evaluating. Results marked with † are expected to be low-quality due to names being in the incorrect writing script.

Latin, Cyrillic, and Arabic

Algorithm	MAP	Recall@1	Recall@5	Recall@10	Recall@50	Recall@100
Double Metaphone†	0.35200	0.11590	0.34803	0.37544	0.44287	0.47522
+ pre-transliteration	0.61014	0.15976	0.53487	0.63575	0.78958	0.83476
Sentence-T5†	0.40391	0.13180	0.38311	0.43399	0.51118	0.54079
+ pre-transliteration	**0.61739**	**0.16086**	**0.53640**	**0.64846**	**0.79956**	**0.85362**
ByT5	0.08393	0.03191	0.06985	0.10614	0.24485	0.33048
+ pre-transliteration	0.10012	0.03355	0.08542	0.11469	0.22763	0.34737
SimCSE†	0.37026	0.13180	0.35461	0.40450	0.47401	0.50581
+ pre-transliteration	0.53519	0.15099	0.47588	0.55888	0.71107	0.76173
Lucene FuzzyQuery†	0.26533	0.10186	0.26447	0.28202	0.31272	0.32149
+ pre-transliteration	0.38923	0.11754	0.35471	0.42708	0.48684	0.50164
Ours (Latin-only)†	0.41 ± 0.02	0.13 ± 0.00	0.37 ± 0.02	0.42 ± 0.02	0.54 ± 0.03	0.61 ± 0.04
+ pre-transliteration	0.57 ± 0.03	0.15 ± 0.00	0.51 ± 0.03	0.59 ± 0.03	0.73 ± 0.04	0.79 ± 0.04
Ours (La+Ar+Cy)	0.49 ± 0.04	0.14 ± 0.00	0.44 ± 0.04	0.52 ± 0.05	0.67 ± 0.03	0.75 ± 0.04
+ pre-transliteration	0.49 ± 0.01	0.14 ± 0.00	0.43 ± 0.01	0.51 ± 0.02	0.65 ± 0.01	0.73 ± 0.02

Latin and Hanzi

Algorithm	MAP	Recall@1	Recall@5	Recall@10	Recall@50	Recall@100
Double Metaphone†	0.43352	0.13000	0.41173	0.46605	0.55247	0.59346
+ pre-transliteration	0.47043	0.13580	0.43247	0.49914	0.64037	0.69901
Sentence-T5†	0.51244	0.14235	0.47728	0.52914	0.59580	0.60963
+ pre-transliteration	**0.54248**	**0.14963**	**0.49852**	**0.55840**	**0.65346**	**0.69815**
ByT5	0.08231	0.02790	0.06728	0.09321	0.22617	0.34309
+ pre-transliteration	0.08535	0.02605	0.07432	0.09802	0.22506	0.35148
SimCSE†	0.47097	0.14136	0.43519	0.49074	0.57617	0.60457
+ pre-transliteration	0.48958	0.14593	0.44420	0.50593	0.63889	0.67284
Lucene FuzzyQuery†	0.34896	0.10309	0.34235	0.37568	0.40037	0.40901
+ pre-transliteration	0.38029	0.10889	0.37691	0.40901	0.44728	0.45593
Ours (Latin-only)†	0.48 ± 0.02	0.14 ± 0.00	0.45 ± 0.03	0.49 ± 0.03	0.58 ± 0.01	0.64 ± 0.01
+ pre-transliteration	0.50 ± 0.03	0.14 ± 0.00	0.46 ± 0.03	0.51 ± 0.04	0.62 ± 0.02	0.68 ± 0.01
Ours (Latin+Hanzi)	0.18 ± 0.00	0.07 ± 0.00	0.16 ± 0.00	0.20 ± 0.00	0.33 ± 0.00	0.42 ± 0.00
+ pre-transliteration	0.18 ± 0.00	0.07 ± 0.00	0.17 ± 0.00	0.21 ± 0.00	0.33 ± 0.00	0.42 ± 0.01

Hangul and Hebrew

Algorithm	MAP	Recall@1	Recall@5	Recall@10	Recall@50	Recall@100
Double Metaphone†	0.01947	0.01546	0.06357	0.11512	0.45120	0.85567
+ pre-transliteration	**0.37594**	**0.27526**	**0.40344**	**0.52921**	**0.69072**	0.92096
Sentence-T5†	0.04846	0.03093	0.14433	0.20103	0.55876	0.90034
+ pre-transliteration	0.12220	0.04330	0.15601	0.18385	0.47251	0.89863
ByT5	0.06540	0.00893	0.06254	0.11237	0.41168	0.79897
+ pre-transliteration	0.04693	0.01409	0.02990	0.05258	0.29691	0.76254
SimCSE†	0.05317	0.02440	0.04570	0.04777	0.05979	0.77491
+ pre-transliteration	0.10626	0.05361	0.10103	0.11649	0.32474	0.85052
Lucene FuzzyQuery†	0.05291	0.02955	0.06117	0.06117	0.15739	0.15739
+ pre-transliteration	0.04553	0.02955	0.05430	0.05430	0.06460	0.06460
Ours (Latin-only)†	0.07 ± 0.00	0.03 ± 0.00	0.06 ± 0.01	0.07 ± 0.02	0.35 ± 0.06	0.81 ± 0.02
+ pre-transliteration	0.18 ± 0.06	0.10 ± 0.05	0.22 ± 0.06	0.29 ± 0.05	0.54 ± 0.09	**0.93 ± 0.00**
Ours (Hang+Hebr)	0.07 ± 0.00	0.03 ± 0.00	0.06 ± 0.01	0.08 ± 0.02	0.35 ± 0.10	0.85 ± 0.02
+ pre-transliteration†	0.05 ± 0.00	0.02 ± 0.00	0.04 ± 0.01	0.04 ± 0.00	0.23 ± 0.02	0.85 ± 0.04

Devanagari, Katakana, and Hiragana

Algorithm	MAP	Recall@1	Recall@5	Recall@10	Recall@50	Recall@100
Double Metaphone†	0.02644	0.01316	0.04605	0.09649	0.55263	**1.00000**
+ pre-transliteration	**0.61297**	**0.24781**	**0.68202**	**0.75658**	**0.94518**	**1.00000**
Sentence-T5†	0.03654	0.01535	0.05921	0.10746	0.42544	**1.00000**
+ pre-transliteration	0.52715	0.23246	0.52193	0.65351	0.94737	**1.00000**
ByT5	0.19207	0.05702	0.20175	0.26974	0.56360	**1.00000**
+ pre-transliteration	0.20043	0.05702	0.19079	0.25658	0.75219	**1.00000**
SimCSE†	0.32438	0.16447	0.30263	0.30263	0.50877	**1.00000**
+ pre-transliteration	0.46655	0.21272	0.45175	0.54167	0.84649	**1.00000**
Lucene FuzzyQuery†	0.15282	0.12061	0.15132	0.15132	0.19079	0.19079
+ pre-transliteration	0.34142	0.17763	0.35088	0.36404	0.37719	0.37719
Ours (Latin-only)†	0.34 ± 0.01	0.18 ± 0.00	0.32 ± 0.01	0.34 ± 0.01	0.56 ± 0.04	**1.00 ± 0.00**
+ pre-transliteration	0.56 ± 0.02	**0.23 ± 0.02**	0.59 ± 0.03	0.71 ± 0.01	**0.94 ± 0.06**	**1.00 ± 0.00**
Ours (Deva+Kana)	0.33 ± 0.00	0.16 ± 0.00	0.32 ± 0.00	0.33 ± 0.00	0.49 ± 0.02	**1.00 ± 0.00**
+ pre-transliteration†	0.31 ± 0.00	0.11 ± 0.01	0.30 ± 0.01	0.38 ± 0.02	0.76 ± 0.03	**1.00 ± 0.00**

Table 5: Results from three runs on the JRC-Names-Retrieval organization datasets. The scripts after "Ours" indicates which scripts were used to train the models. The highest score in each column is in bold. "+ pre-transliteration" indicates that all names were transliterated into Latin before evaluating. Results marked with † are expected to be low-quality due to names being in the incorrect writing script. Note that Lucene only returns scores for matching subsets of its index, so the Deva+Kana Recall@100 scores are below 1.0.

our experiments being particularly well-equipped to transliterate names from Arabic and Cyrillic scripts into Latin. This is supported by the fact that the Latin and Hanzi dataset shows a larger gap between the two algorithms, as the transliteration engine we used is known to perform comparably less well on Hanzi names.

For the high-resource organization names, we find that the pretrained vector methods excel. As organization name matching often requires semantic information alongside phonetic information, of which the latter is not known to be captured by most text embedding procedures, so we expect that these methods will do better on organization names than personal names. Sentence-T5 in particular appears to be well-suited to organization name matching when a transliteration engine is used. While transliterating organization names often results in tokens which are not actual words in any Latin-script language (e.g. "同济大学", which is machine-transliterated to "Tongji Daxue"), Sentence-T5 is still able to leverage these transliterations in two ways: (1) the non-Latin characters are tokenized as UNK tokens by Sentence-T5, rendering their embeddings largely meaningless, and (2) the transliteration of the non-semantic components (e.g. "Tongji," in the previous example) is often enough to give the embedding for the organization name a substantial boost in the ranked result list.

Regarding the low-resource script groups, we find that, while our triplet-based neural baseline generally achieves the highest results for names in their original writing scripts, they are far and away outmatched by other systems when coupled with transliteration engines. This is likely due to the limited amount of training data available for these low-resource script groups; we hypothesize that some form of transfer learning would be able to close this performance gap.

6.1. 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.

7. Discussion

We present a standardized dataset for evaluating name retrieval alongside a wide variety of baseline scores. These results indicate that, while traditional approaches such as Double Metaphone do quite well when a transliteration engine is present, neural networks offer a promising alternative when none is available. Moreover, the gap in performance shown between our neural network and other baselines indicates that there remains a great deal of room to explore this problem space further in the are of personal names.

We find that Sentence-T5 coupled with a transliteration engine is a strong baseline for the matching of organization names. A multilingual extension of Sentence-T5 would perhaps be able to achieve similar performance on organization names without the need for an external transliteration engine (which can be hard to produce without experts on the writing script). Similarly, this model could perhaps be coupled with one similar to what we present in this paper in order to directly model phonetic information.

Other future directions for this work include extending it to include more scripts in the shared embedding space 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. Additionally, transfer learning approaches should be explored in order to shrink the performance gap of our metric learning baseline. Finally, JRC-Names-Retrieval exhibits a number of phenomena described in Section 3, but not all of them. While some (e.g. out-of-order components) can be introduced manually via noising techniques, it would be preferable to explore alternative data sources which naturally exhibit the sort of variation found in real-world databases.

7.1. Limitations and Ethical Considerations

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).

Acknowledgements

The authors would like to thank Karin Lin, Chris Mack and Jimmy Moon for their project management support. This research was done in coordination with Saltlux Inc. and partially funded by the Exobrain Project.

8. Bibliographical References

Nils Barlaug and Jon Atle Gulla. 2021. Neural networks for entity matching: A survey. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(3):1–37.

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

Weihua Chen, Xiaotang Chen, Jianguo Zhang, and Kaiqi Huang. 2017. Beyond triplet loss: A deep quadruplet network for person re-identification. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1320–1329.

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

Sumit Chopra, Raia Hadsell, and Yann 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.

William W Cohen, Pradeep Ravikumar, Stephen E Fienberg, et al. 2003. A comparison of string distance metrics for name-matching tasks. In *IIWeb*, volume 3, pages 73–78. Citeseer.

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

Dedupe.io. 2020. soft-tfidf. `https://github.com/dedupeio/soft-tfidf`.

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.

Anderson A Ferreira, Marcos André Gonçalves, and Alberto HF Laender. 2012. A brief survey of automatic methods for author name disambiguation. *Acm Sigmod Record*, 41(2):15–26.

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

Octavian-Eugen Ganea, Marina Ganea, Aurelien Lucchi, Carsten Eickhoff, and Thomas Hofmann. 2016. Probabilistic bag-of-hyperlinks model for entity linking. In *Proceedings of the 25th International Conference on World Wide Web*, WWW '16, page 927–938, Republic and Canton of Geneva, CHE. International World Wide Web Conferences Steering Committee.

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.

Yifei Hu, Xiaonan Jing, Youlim Ko, and Julia Taylor Rayz. 2020. Misspelling correction with pretrained contextual language model. In *2020 IEEE 19th International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*, pages 144–149. IEEE.

Sai Muralidhar Jayanthi, Danish Pruthi, and Graham Neubig. 2020. NeuSpell: A neural spelling correction toolkit. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 158–164, Online. Association for Computational Linguistics.

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.

Hanna Köpcke, Andreas Thor, and Erhard Rahm. 2010. Evaluation of entity resolution approaches on real-world match problems. *Proc. VLDB Endow.*, 3(1–2):484–493.

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

Yanen Li, Huizhong Duan, and ChengXiang Zhai. 2012. Cloudspeller: Query spelling correction by using a unified hidden markov model with web-scale resources. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12 Companion, page 561–562, New York, NY, USA. Association for Computing Machinery.

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 pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874, Dublin, Ireland. Association for Computational Linguistics.

George Papadakis, Dimitrios Skoutas, Emmanouil Thanos, and Themis Palpanas. 2019. A survey of blocking and filtering techniques for entity resolution. *arXiv preprint arXiv:1905.06167*.

Minh C. Phan, Aixin Sun, and Yi Tay. 2019. Robust representation learning of biomedical names. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3275–3285, Florence, Italy. Association for Computational Linguistics.

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

Keisuke Sakaguchi, Kevin Duh, Matt Post, and Benjamin Van Durme. 2017. Robsut wrod reocginiton via semi-character recurrent neural network. In *Thirty-first AAAI conference on artificial intelligence*.

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.

Mujeen Sung, Hwisang Jeon, Jinhyuk Lee, and Jaewoo Kang. 2020. Biomedical entity representations with synonym marginalization. In *ACL*.

Chuanqi Tan, Furu Wei, Pengjie Ren, Weifeng Lv, and M. Zhou. 2017. Entity linking for queries by searching wikipedia sentences. In *EMNLP*.

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.

Perceval Wajsbürt, Arnaud Sarfati, and Xavier Tannier. 2021. Medical concept normalization in french using multilingual terminologies and contextual embeddings. *Journal of Biomedical Informatics*, 114:103684.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Charagram: Embedding words and sentences via character n-grams. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1504–1515, Austin, Texas. Association for Computational Linguistics.

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.

Xiyuan Yang, Xiaotao Gu, Sheng Lin, Siliang Tang, Yueting Zhuang, Fei Wu, Zhigang Chen, Guoping Hu, and Xiang Ren. 2019. Learning dynamic context augmentation for global entity linking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 271–281, Hong Kong, China. Association for Computational Linguistics.

Wei Zhang, Yan Chuan Sim, Jian Su, and Chew Lim Tan. 2011. Entity linking with effective acronym expansion, instance selection and topic modeling. In *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence - Volume Volume Three*, IJCAI'11, page 1909–1914. AAAI Press.

Shuyan Zhou, Shruti Rijhwani, John Wieting, Jaime Carbonell, and Graham Neubig. 2020. Improving Candidate Generation for Low-resource Cross-lingual Entity Linking. *Transactions of the Association for Computational Linguistics*, 8:109–124.

A. JRC-Names-Retrieval 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 Babel Street.

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 evaluation data is shown in Table 6.

Person Entities

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

Organization Entities

Database Rows	841
Queries	152
Avg. Expected per Query	5.53

Table 6: Latin/Arabic/Cyrillic data split breakdown.

The Latin/Hanzi training data consists of 97,077 names, grouped into clusters of an average size of 2.53 names, and the breakdown for the Latin/Hanzi evaluation data is shown in Table 7.

The Hangul/Hebrew training data consists of 1,331 names, grouped into clusters of an average size of 2.07 names, and the breakdown for the Hangul/Hebrew evaluation data is shown in Table 8.

The Devanagari/Kana training data consists of 1,792 names, grouped into clusters of an average

Person Entities

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

Organization Entities

Database Rows	738
Queries	135
Avg. Expected per Query	5.51

Table 7: Latin/Hanzi data split breakdown.

Person Entities

Database Rows	316
Queries	266
Avg. Expected per Query	1.18

Organization Entities

Database Rows	119
Queries	97
Avg. Expected per Query	1.23

Table 8: Hebrew/Hangul data split breakdown.

size of 3.94 names, and the breakdown for the Devanagari/Kana evaluation data is shown in Table 9.

Person Entities

Database Rows	550
Queries	160
Avg. Expected per Query	3.43

Organization Entities

Database Rows	90
Queries	38
Avg. Expected per Query	2.36

Table 9: Devanagari/Kana 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 subsplits for Latin/Arabic/Cyrillic, Latin/Hanzi, Hangul/Hebrew, and Devanagari/Kana script combinations for both person and organization entity types.

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.*

Not at present.

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 https://github.com/peblair/jrc-names-retrieval.

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 the license specified in the dataset repository.

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?

Babel Street will be supporting this dataset.

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

`pblair@babelstreet.com`

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?