Cross-lingual NIL Entity Clustering for Low-resource Languages

Kevin Blissett and **Heng Ji**
Computer Science Department
Rensselaer Polytechnic Institute
{blissk,jih}@rpi.edu

Abstract

Clustering unlinkable entity mentions across documents in multiple languages (*cross-lingual NIL Clustering*) is an important task as part of Entity Discovery and Linking (EDL). This task has been largely neglected by the EDL community because it is challenging to outperform simple edit distance or other heuristics based baselines. We propose a novel approach based on encoding the orthographic similarity of the mentions using a Recurrent Neural Network (RNN) architecture. Our model adapts a training procedure from the one-shot facial recognition literature in order to achieve this. We also perform several exploratory probing tasks on our name encodings in order to determine what specific types of information are likely to be encoded by our model. Experiments show our approach provides up to a 6.6% absolute CEAFm F-Score improvement over state-of-the-art methods and successfully captures phonological relations across languages.

1 Introduction

The objective of Entity Discovery and Linking (EDL; Ji et al. 2014) is to identify within a text or set of texts all of the names which refer to entities in the world and then link those name mentions to a Knowledge Base (KB). Common approaches to EDL, however, often ignore the question of what to do with name mentions that cannot be linked to the KB.

Clustering unlinkable mentions is often critical to successfully extracting relevant information from a given corpus about emergent situations. For example, before June of 2013, no Wikipedia entry existed for Edward Snowden. But, suddenly, in that month, properly identifying and clustering thousands of mentions in dozens of languages for this entity became a key task for IE systems focused on breaking news. Similar situations occur when significant political events occur in remote areas. For example, in November of 2015, protests broke out in the town of Ginci in Ethiopia (Pinaud and Raleigh, 2017), but Ginci also does not appear in Wikipedia.

Orthographic similarity provides one of the best single indicators of which mentions ought to be clustered. By relying on this clue, given two parallel sentences, human annotators are often able to accurately determine which names refer to the same entity even without speaking the relevant languages (Lin et al., 2018). However, this remarkable ability cannot be captured using simple string similarity measures (such as edit distance) alone. For example, Figure 1 shows the edit distance between several mentions of the same entity *Ethiopia* as they appeared in an Oromo corpus. The edit distances between various mentions vary widely, making it extremely difficult to design a clustering system based only on this metric and a predefined threshold.

	#ethiopa	ethiipiyaafi	ethiophiyaattuu	ethiopia	itiyophiyaa	itoophiyaawwidha
#ethiopa	0	7	9	2	8	13
ethiipiyaafi		0	6	5	7	10
ethiophiyaattuu			0	7	7	9
ethiopia				0	6	11
itiyophiyaa					0	8
itoophiyaawwidha						0

Figure 1: Edit distance between several mentions of the same entity *Ethiopia* from an LDC Oromo news corpus.

We propose to encode name mentions using a character-based Recurrent Neural Network (RNN). We train this model in a manner inspired by work on the analogous task of one-shot facial

Proceedings of the 2nd Workshop on Computational Models of Reference, Anaphora and Coreference (CRAC 2019), pages 20–25,
Minneapolis, USA, June 7, 2019. ©2019 Association for Computational Linguistics

recognition as addressed by (Schroff et al., 2015). In that task, a single model encodes images into a shared space. Images of the same person are identified by measuring which encoding vectors are near one another in that space. Analogously, we consider different name mentions of the same entity as different "views" of that entity. Mentions are encoded such that mentions likely to refer to the same entity are close to one another in the encoding space. Mentions can then be clustered using standard clustering techniques.

2 Approach

2.1 Basic Model

In our approach, input mentions are represented as a sequence of vectors $\boldsymbol{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, ..., \boldsymbol{x}_L\}$ where $\boldsymbol{x}_i$ is a vector representing the i-th character of a mention and L is the number of characters in the mention. $\boldsymbol{x}_i$ is an embedding vector for each character trained jointly with the rest of the model. Input sequences are then fed to a bi-directional RNN based on Gated Recurrent Units (GRU; Cho et al. 2014). The hidden representations $\boldsymbol{h}_i$ produced by the model are passed to a fully-connected layer which creates an unnormalized encoding for the mention $\boldsymbol{n}_i$ which is normalized to unit length to produce the final encoding for the mention $\boldsymbol{y}_i$. A margin α, is set during training, which controls the target minimum distance between any mention x and any other mention which does not refer to the same entity.

Mentions are clustered into disjoint subsets S_i using the DBSCAN algorithm. We select DBSCAN primarily because it does not require the user to pre-specify the number of expected clusters. We set the hyperparameter ϵ for the clustering by performing a grid search over possible values and selecting the value that maximizes the CEAFm score (Ji et al., 2014) on the training data.

Typical hyperparameter values for the encoding model are summarized in Table 2. Dropout was also used in order to provide regularization when training data was limited.

2.2 Training Procedure

During training, the model is presented with triplets of name mentions (x_a, x_p, x_n). These triplets consist of an *anchor* x_a, a *positive* x_p and a *negative* x_n. The anchor is drawn from the set of name mentions M_A which have at least two name mentions in their cluster. That is, given the vocabulary of all name mentions V

$$M_A = \{x_a | (\exists x)[x_a \in V \wedge \texttt{refers\_to}(x_a, e) \wedge \texttt{refers\_to}(x, e) \wedge x_a \neq x]\}$$

The positive is a name mention drawn from the set of name mentions M_{Pi} which refer to same entity as the i-th anchor mention x_{ai}.

$$M_{Pi} = \{x_p | x_p \in V \wedge \texttt{refers\_to}(x_p, e) \wedge \texttt{refers\_to}(x_{ai}, e) \wedge x_p \neq x_{ai}\}$$

An example anchor-positive pair may consist of the names *(Bill Gates, Gates)*. The negative is a name that does not refer to the same entity as the anchor. Rather than selecting the negative randomly, we select the negative example which has an encoding closest to the anchor as measured by Euclidean distance. In this way, we follow the example of (Schroff et al., 2015) and select for the negative a name calculated to provide useful information about areas of poor model performance. For example, the anchor-positive-negative triplet *(Bill Gates, Gates, Gaines)* is more difficult and therefore likely to be informative to the model than the triplet *(Bill Gates, Gates, Smith)*. Over the course of training, this negative sampling technique ensures that the model is consistently exposed to informative examples. Randomly sampling the negative provides no guarantee that the model will ever see the triplets it most needs to improve. Our experimental results show that this non-random negative sampling led to a meaningful improvement in model performance.

More specifically, negatives are sampled according to the following procedure: before training and after the model is presented with each training batch, all names in the dataset are encoded by the model in its current state and the encodings are cached. For any given pair of an anchor encoding and a positive encoding (x_a, x_p), a name is chosen from the dataset vocabulary of name mentions V to serve as the negative x_n such that the encoding of the negative is as close to the anchor encoding as possible. Treating our encoding model as a single function f, we select the negative according to the following equation:

$$\arg\min_{x_n \epsilon V} \|f(x_a) - f(x_n)\|_2^2$$

	Naive Baseline	Edit Distance	Random Sampling	Our Approach
Oromo	0.531	0.840	0.827	**0.868**
Tigrinya	0.573	0.806	0.828	**0.872**
Oromo + Tigrinya	0.454	0.828	0.817	**0.841**

Table 1: CEAFm F-score for baseline models compared to system performance.

Hidden Dim.	64
Num. RNN Layers	2
Embedding Dim.	16
Margin α	0.2
Output Dim.	16

Table 2: Typical hyperparameters for the encoding model.

As an optimization, in practice we subject the negative to the following additional constraint:

$$\|f(x_a) - f(x_n)\|_2^2 > \|f(x_a) - f(x_p)\|_2^2$$

This produces what is referred to as a "semi-hard" example in (Schroff et al., 2015), and was found to be an important optimization to ensure that the model converges.

Triplets constructed according to this procedure are then encoded by the model and the model is trained to optimize the following loss function:

$$\sum_i^N \left[\|f(x_i^a) - f(x_i^p)\|_2^2 - \|f(x_i^a) - f(x_i^n)\|_2^2 + \alpha \right]_+$$

A single epoch of training consists of showing the model all triplets composed of every possible anchor-positive pair and their corresponding negative x_i^n.

Our method can be applied to cross-lingual datasets without modification. New anchor-positive pairs are constructed by pairing mentions that refer to the same entity regardless of the language of origin and training proceeds as normal.

3 Experiment

3.1 Data and Scoring Metric

We use two languages, Oromo and Tigrinya, for our experiments. Both languages are members of the Afro-Asiatic language family, but they belong to separate branches of that family and have distinct phonologies, grammars, and writing systems from one another. We select these languages as exemplars of extremely low-resource languages. These languages have been used in the standard NIST shared tasks LoreHLT[1]. Specifically we use data from the DARPA LORELEI program[2] because these data sets include human annotated ground truth. Each mention in the dataset belongs to one of the four following types: person, organization, geopolitical entity, or location. We also create a combined Oromo-Tigrinya dataset by merging the two datasets. Table 3 shows the detailed data statistics. The test set consists only of NIL mentions. We produce it by collecting all NIL mentions from the complete dataset and dividing them randomly. One portion is used as the test set and the other is used during model training as a development set. All non-NIL mentions are also used during training. Our final model for Oromo is trained on 4101 mentions from 327 clusters while for Tigrinya we use 3990 mentions from 330 clusters.

	Tigrinya	Oromo
Test Set NIL Mentions	640	894
Test Set NIL Clusters	78	70

Table 3: Statistics for experimental datasets. *Test Set NIL Clusters* refers to the number of clusters with size > 1.

3.2 Results

We compare our system's score to three baselines. The first is a naive baseline which gives the score on the dataset if every mention is assigned a unique cluster ID. We also compare the scores to a baseline based primarily on edit distance and enhanced with simple heuristics (such as merging a mention of a person's last name with mentions of their full name that appear in the same document). The edit distance baseline does not incorporate any special weighting for edits, but does use the Python Unidecode package (Solc, 2009) in order to map Tigrinya into ascii characters in the combined dataset task. Finally, we report the

[1] https://www.nist.gov/itl/iad/mig/lorehlt-evaluations

[2] LDC2017E57 and LDC2017E58 in the LDC Catalog

results for a model that is the same as our final model, but trained by sampling the negative randomly. Table 1 shows the results.

Our model improves on the baseline for all datasets. Of special note is the increase in performance we get from using the negative sampling technique designed by (Schroff et al., 2015). For all datasets, sampling the most difficult negative rather than sampling randomly gives a significant increase in performance of about 4% CEAFm F-score. In our experience, this technique also seemed to help reduce overfitting by varying the training data from batch to batch.

The baseline model performed significantly worse on Tigrinya compared to Oromo. The reasons for this are not entirely clear, but it could be due to the fact that the Ge'ez script used for Tigrinya is an abugida (a writing system which represents entire syllables with single characters) and contains a large number of characters. This means that syllables (and by extension, words) that are phonetically similar are spelled with entirely different characters even if they share some vowel or consonant sounds. Notably, applying the Unidecode package did not seem to remedy this issue. Whatever the problem, our model did not seem to encounter the same struggles and actually performed better on Tigrinya relative to the other tasks.

4 Probing Mention Encodings

In order to illustrate some of the linguistic information captured by our model, we give an example examination of the vectors produced by several closely related input sequences. This qualitative analysis illustrates that our model learns which letter alternations are most and least important when transliterating words between given language pairs.

We hypothesize that alternating among characters that represent very similar sounds between two languages should make little difference in the final encoding. By determining which alternations cause the smallest difference in output encodings, we can ascertain which letters the model finds are most interchangeable for this language pair.

We train an encoder model on the Google Arabic to English transliteration dataset (Rosca and Breuel, 2016), for this example.

Table 4 shows the result of alternating the first letter of the name 'peter' after training our model.

Replacement	Distance
p → b	0.026
p → baa	0.060
p → shiin	1.05
p → raa	1.02

Table 4: First letter replacements for the name 'peter' which make the largest and smallest differences in the output encoding.

Shown in the right column is the distance of the output encoding from the original encoding after making each replacement. Our model encodes names beginning with 'p', 'b', or the Arabic 'baa' similarly because alternations among these sounds do not often distinguish the names of entities from one another in this language pair. This is because Arabic has no equivalent of the English 'p' sound, and thus, in English names containing 'p's and 'b's, those letters are most commonly transliterated to the single Arabic character 'baa'.

5 Related Work

The task of clustering NIL entity mentions was introduced in the TAC2014 Knowledge Base Population track (Ji et al., 2014). Approaches to this task have included using direct string or substring matches (Cassidy et al., 2011; Jiang et al., 2017) and edit distance based metrics (Ploch et al., 2011; Greenfield et al., 2016). Elaborations on these methods include leveraging systems for entity coreference (Huynh et al., 2013), query expansion (Radford et al., 2011; Yu et al., 2013) using context features either on the document level (Fahrni et al., 2013; Graus et al., 2012; Hong et al., 2014) or the sentence level (Ploch et al., 2011), and applying hand-crafted heuristic rules (Al-Badrashiny et al., 2017; Li et al., 2016). Our method differs from the above by clustering based on a measurement of surface similarities between words, but not relying directly on edit distance.

We found in our experiments that the standard DBSCAN algorithm for clustering (Ester et al., 1996) performed well, but many additional clustering techniques for NIL entities were explored in (Tamang et al., 2012). In particular, hierarchical agglomerative clustering methods have seen some success (Zhang et al., 2012; Graus et al., 2012; Ploch et al., 2012) Because our work focused on the effective encoding of word forms rather than new techniques for clustering the encoded vectors,

we did not pursue these techniques.

Neural machine transliteration models have used RNNs to encode input character sequences into fixed length vectors (Finch et al., 2016; Jadidinejad, 2016; Ameur et al., 2017). This is similar to our own approach, but where these models produce vectors only as an intermediate step (to be later fed to a decoder network), we use the vectors produced by the encoder directly and do not use a decoder at all.

Our training procedure relies on a negative sampling technique from the one-shot facial recognition literature (Schroff et al., 2015). More specifically, we sample our negative samples according to the method used to train FaceNet.

6 Conclusion and Future Work

We construct a model to encode the surface features of words and cluster those encodings to determine which unlinkable mentions refer to the same entities. Our model shows improvement over baseline models based on edit distance and ad hoc heuristic rules. Future work may include incorporating more information from the context surrounding the name mentions and exploration of new encoding architectures and clustering algorithms.

Acknowledgments

This research is based upon work supported in part by U.S. DARPA LORELEI Program # HR0011-15-C-0115, and the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via contract # FA8650-17-C-9116, and ARL NS-CTA No. W911NF-09-2-0053. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of DARPA, ODNI, IARPA, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright annotation therein.

References

Mohamed Al-Badrashiny, Jason Bolton, Arun Tejasvi Chaganty, Kevin Clark, Craig Harman, Lifu Huang, Matthew Lamm, Jinhao Lei, Di Lu, Xiaoman Pan, et al. 2017. Tinkerbell: Cross-lingual cold-start knowledge base construction. In *TAC*, pages 1–12.

Mohamed Seghir Hadj Ameur, Farid Meziane, and Ahmed Guessoum. 2017. Arabic machine transliteration using an attention-based encoder-decoder model. *Procedia Computer Science*, 117:287–297.

Taylor Cassidy, Zheng Chen, Javier Artiles, Heng Ji, Hongbo Deng, Lev-Arie Ratinov, Jing Zheng, Jiawei Han, and Dan Roth. 2011. CUNY-UIUC-SRI TAC-KBP2011 entity linking system description. In *TAC*, pages 1–13.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *EMNLP*, pages 1724–1734.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. 1996. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, pages 226–231. AAAI Press.

Angela Fahrni, Benjamin Heinzerling, Thierry Göckel, and Michael Strube. 2013. HITS' monolingual and cross-lingual entity linking system at TAC 2013. In *TAC*, pages 1–10.

Andrew Finch, Lemao Liu, Xiaolin Wang, and Eiichiro Sumita. 2016. Target-bidirectional neural models for machine transliteration. In *Proceedings of the Sixth Named Entity Workshop*, pages 78–82.

David Graus, Tom Kenter, Marc Bron, Edgar Meij, Maarten De Rijke, et al. 2012. Context-based entity linking-University of Amsterdam at TAC 2012. In *TAC*, pages 1–6.

Kara Greenfield, Rajmonda S Caceres, Michael Coury, Kelly Geyer, Youngjune Gwon, Jason Matterer, Alyssa Mensch, Cem Safak Sahin, and Olga Simek. 2016. A reverse approach to named entity extraction and linking in microposts. In *#Microposts*, pages 67–69.

Yu Hong, Xiaobin Wang, Yadong Chen, Jian Wang, Tongtao Zhang, Jin Zheng, Dian Yu, Qi Li, Boliang Zhang, Han Wang, et al. 2014. RPI-BLENDER TAC-KBP2014 knowledge base population system. In *TAC*, pages 1–12.

Huy M Huynh, Trong T Nguyen, and Tru H Cao. 2013. Using coreference and surrounding contexts for entity linking. In *RIVF*, pages 1–5. IEEE.

Amir H Jadidinejad. 2016. Neural machine transliteration: Preliminary results. *arXiv preprint arXiv:1609.04253*.

Heng Ji, Joel Nothman, and Ben Hachey. 2014. Overview of TAC-KBP2014 entity discovery and linking tasks. In *TAC*, pages 1–15.

Shanshan Jiang, Yihan Li, Tianyi Qin, Qian Meng, and Bin Dong. 2017. SRCB entity discovery and linking (EDL) and event nugget systems for TAC 2017. In *TAC*, pages 1–11.

Manling Li, Xinlei Chen, Yantao Jia, Yuanzhuo Wang, Xiaolong Jin, Zixuan Li, Juan Yao, Fan Yang, Yunqi Qiu, Jialin Su, et al. 2016. OpenKN at TAC KBP 2016. In *TAC*, pages 1–8.

Ying Lin, Cash Costello, Boliang Zhang, Di Lu, Heng Ji, James Mayfield, and Paul McNamee. 2018. Platforms for non-speakers annotating names in any language. In *Proceedings of ACL 2018, System Demonstrations*, pages 1–6.

Margaux Pinaud and Clionadh Raleigh. 2017. Data analysis: The roots of popular mobilization in Ethiopia.

Danuta Ploch, Leonhard Hennig, Ernesto William De Luca, and Sahin Albayrak. 2011. DAI approaches to the TAC-KBP 2011 entity linking task. In *TAC*, pages 1–9.

Danuta Ploch, Leonhard Hennig, Angelina Duka, Ernesto William De Luca, and Sahin Albayrak. 2012. GerNED: A German corpus for named entity disambiguation. In *LREC*, pages 3886–3893.

Will Radford, Ben Hachey, Matthew Honnibal, Joel Nothman, and James R Curran. 2011. Naïve but effective NIL clustering baselines–CMCRC at TAC 2011. In *TAC*, pages 1–3.

Mihaela Rosca and Thomas Breuel. 2016. Sequence-to-sequence neural network models for transliteration. *arXiv preprint arXiv:1610.09565*.

Florian Schroff, Dmitry Kalenichenko, and James Philbin. 2015. FaceNet: A unified embedding for face recognition and clustering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 815–823.

Tomaz Solc. 2009. Unidecode. [Online; accessed 10 December 2018].

Suzanne Tamang, Zheng Chen, and Heng Ji. 2012. CUNY BLENDER TAC-KBP2012 entity linking system and slot filling validation system. In *TAC*, pages 1–21.

Dian Yu, Haibo Li, Taylor Cassidy, Qi Li, Hongzhao Huang, Zheng Chen, Heng Ji, Yongzhong Zhang, and Dan Roth. 2013. RPI-BLENDER TAC-KBP2013 knowledge base population system. In *TAC*, pages 1–14.

Tao Zhang, Kang Liu, and Jun Zhao. 2012. The NLPRIR entity linking system at TAC 2012. In *TAC*, pages 1–5.