Classifying and Searching Resource-Poor Languages More Efficiently. Using the FastText Word Embeddings for the Aramaic Language Family.

Mathias Coeckelbergs
Université libre de Bruxelles
Avenue F.D. Roosevelt 50, 1000 Brussels, Belgium
mathias.coeckelbergs@gmail.com

Abstract

Within the context of the FastText initiative, pre-trained word embeddings have been made available for 294 language, based on the respective Wikipedia corpus for the particular languages. One of these languages is Aramaic, which is currently conceived of as endangered, since it is only spoken by a few minority groups in the Middle East. Nevertheless, this language has a rich history of culture and literature, being among others also the language of Jesus and the main language of the Syriac culture, which lives on today as a liturgical language in several church denominations in among others India, Syria and Iraq. This paper wants to provide first insights into the usefulness of these word embeddings to connect the separate parts of Aramaic culture, and to study them as one language with many facets and influence, a subject which hitherto has only seen separated scholarship along the lines of research questions limited to a specific time frame. Using some of the specific assets of the FastText algorithm, we show how traditional difficulties in bringing together the Aramaic literature from a computational perspective, such as limited training resources and significant lexical richness due to external influences throughout the centuries, can now accounted for.

Keywords: word embeddings, Aramaic, FastText, Out-Of-Vocabulary words

1. Introduction

In recent years, word embeddings have developed into a varied research field within natural language processing. The techniques, among which Word2Vec, developed by Mikolov et al. (2013) is the most widely used, are best known for their ability to derive word analogies from a strictly unsupervised machine learning approach. Stated in other words, this means that - as the standard example goes- when one subtracts the 'man' vector from the 'king' vector and adds the 'woman' vector, the resulting vector is found to be most similar to that of 'queen'.

Although word embeddings have proven their worth, an important problem is that a large corpus is needed to learn useful embeddings. Hence, many corpora do not suffice, certainly those containing historical corpora or other resource-poor languages. As machine learning techniques, which are important for information retrieval tasks among others, require numbers, traditionally resource-poor languages lack application value for these techniques, providing one of the main reasons why they are understudied regarding these new wave of methods prevalent within the digital humanities practice. Finding solutions to this problem has yet seen several proposals, including for example methods to (artificially) generate new sentences based on the limited amount which is readily available. For historical texts such approach is methodologically problematic, because, since most texts have a intricate redactional history, it is difficult to provide training data which is undoubtedly representative of a certain language phase.

In this article we show how the study of the Aramaic language in its broad sense -including the various contemporary dialects as well as the stages of its long history- can be leveraged by the FastText word embeddings created by the Facebook AI group. The reason for studying the Aramaic language is that it is currently, despite its rich tradition spanning about 3100 years, considered an endangered language (Naby, 2013). It is spoken throughout various communities in Iran, Iraqi Kurdistan, Syria and South Turkey, but the population who speaks it fluently is growing older, with fewer young people learning the language due to the prevalence of Arabic in the region (Sabar, 2002). Nevertheless, small communities are being formed in Israel and the Netherlands.

The main research question we treat in this article is how to be able to search and classify Aramaic documents more efficiently using the FastText word embeddings. As we will explain further, research into the Aramaic language is splintered according to the specific time period groups of researchers are interested in. Most of the documents available in the various branches of the language are mutually intelligible. FastText uses the Wikipedia articles available in Moderen Aramaic, being the sole resource of machine learning tools for the language to our knowledge. In this article, we show how relatively small amount of training data, and the severe amount of Out-Of-Vocabulary words, two main reasons why machine

learning for Aramaic is difficult, is resolved by the FastText approach.

2. The FastText project

The FastText project provides researchers with 300-dimensional word vectors for (currently) 294 languages, which includes Aramaic. The vectors are trained on the corresponding set of available Wikipedia pages for the respective languages.

The idea behind the use of pre-trained word vectors is that users no longer have to train the embeddings on the corpus which they want to model. This approach can have its drawbacks, for example for research into diachronic variation, for which it is useful to train word vectors on several language phases, after which the vectors can be used to measure semantic shifts across time. Of course, for diachronic time frames for which too few data exists, this approach is not applicable. On the other hand, the FastText approach allows users to start their exploration with salient word vectors for the language under scrutiny, so that corpora with insufficient data to use traditional machine learning methods on, can nevertheless be investigated.

The method of the FastText algorithm is -apart from one condition- exactly the same as that of the Word2Vec algorithm, developed by Mikolov et al. (2013), which brought word embeddings to the forefront of machine learning research, since it was first coined by Bengio et al. (2003). The sole aspect in which both algorithms differ is that Word2Vec takes words as basic entities for which the algorithm assigns vectorial representations, whereas the FastText algorithm does so for n-grams. The best known example for which Word2Vec has received its fame, is that it showed that the algorithm can recognise word analogies, without having any explicit semantic knowledge. The standard example goes that if one starts out from the vector king, subtracts the one for man and adds that for woman, the resulting vector is shown to be closed to that for queen.

A drawback of this approach, however, is that when a word is not encountered in the training phase, it is an Out-Of-Vocabulary (OOV) word, and will be assigned the null vector, because it cannot learn new (and reliable) word vectors in real-time (Chen et al., 2015). It still is useful within the context of the FastText algorithm to speak of an OOV word, because the n-grams are still evaluated within the context of a word (see also Wieting et al. (2016)). As the creators of FastText describe themselves, this allows the algorithm to infer similar meanings between morphemes (Bojanowski et al., 2016).

3. Brief History of Aramaic Languages

The Aramaic language has spanned a long history of contact with various other languages, leading to the development of a web of strongly related languages, which show mutual intelligibility with the exception of specific words, which represent the various external influences. The development of Aramaic languages, which leads up to today, spans about 3100 years, although discussion as to the date of the oldest fragments persists (Beyer 1986).

In summary, we can state that both the modern variety of Aramaic, as well as its many historical phases all have a large part in common, though many differences persist, most notably on the vocabulary level (Creason, 2008). In the next chapter we will show how applying the FastText word embeddings to Aramaic can provide a new point of view for discussing lexical differences and distinct influence on the Aramaic language family.

The contemporary form of Aramaic, denoted by the term Neo-Aramaic or Modern Aramaic, comprises two main forms, being Eastern and Western Aramaic, with the former being much more prominent than the latter. The western variety is today solely spoken in the vicinity of Maalouly, a Syrian city close to the border with Lebanon. Within its history more varieties of the western dialects are attested, but the eastern variety has produced more documents, since most of the literature is written in an eastern variety, with the most well-known text written in the Aramaic language commonly referred to as Syriac, written from the 4th till the 8th century. The literature in this phase of the Aramaic language is so extensive that it comprises about 90% of all Aramaic writings. This language is strongly connected to the varieties of Aramaic spoken most widely today, being Assyrian, Chaldean and Surayt/Turoyo Aramaic. Needless to say, the modern Aramaic on which the Fasttext algorithm was run, hence has strong connection to all of these languages. It has to be noted that the study of the Aramaic languages is done in a haphazard way, being that scholars tend to specialise in a specific area of Aramaic, leaving the comparison between all varieties of the language vastly understudied.

4. Applying FastText to Aramaic

4.1. Language phases characterised by OOV words

As we can conclude from the previous sections, the main reason why advanced classification methods such as word embeddings have not yet seen applications for low-

resource languages such as Aramaic, is because its different language phases rarely have enough training data to achieve salient computational models. Of course, for modern Aramaic more sources exist and can be produced because it is a modern language which is still spoken, lending the application to such resources as Wikipedia articles, on which the pre-trained model for Aramaic by FastText is based.

The algorithm has a particular way of dealing with Out-Of-Vocabulary (OOV) words, which will prove very useful for the purpose of dealing with Aramaic texts which are strongly related to the language used in Wikipedia, but which nevertheless can be considered as a separate dialect (other eastern Aramaic variants, and the historical Syriac). In regular machine learning tasks we would encounter the problem of a significant amount of OOV words, which in the case of word embeddings would either result in no vector (or null vector) corresponding to the OOV word, for which no example in context was presented during training, or in the creation of a random vector. Although this latter option makes sure that each word has a corresponding vector, properties related to the vectorial representation are no longer preserved, meaning that no semantic information, such as needed for the word analogies or words closest in meaning described above, can be derived from them (Joulin et al., 2016).

Since the FastText algorithm creates vectors for n-grams in stead of words, this solves the traditional problem of encountering words for which no vectorial representation was made during the training phase, since the algorithm does not deal with text on the word-level. Moreover, since the newly encountered words are given a vector presents a good estimation of its semantics, this provides an invaluable tool to discuss language variation among the Aramaic language family, and provides a novel viewpoint for issues regarding hapax legomena, words occurring only once in the corpus.

4.2. Assets and Drawbacks of the FastText Algorithm for Aramaic

Apart from the clear asset of dealing with OOV words, which is a general positive - and therefore applicable to all languages - difference of FastText, in comparison with other word embeddings algorithms, other points apply more specifically to the study of Aramaic using these word embeddings.

A first asset of this approach is that lexical and morphological corrections can be performed. Similar applications of word embeddings have previously been explored by Luong et al. (2013). Since Aramaic words are -like all other Semitic languages such as Arabic, Akkadian and Hebrew- based on three basic consonants, which constitute the root of the word and represents the core semantic meaning, this means that once the vowels are added to this basic root, the resulting vectorial representations for the different words derived from the same root will also lie close to each other. This solves an important problem occurring in the modeling of Semitic languages, namely to automatically infer the lexeme (or base root) of every word, including rare ones. As we have pointed out, this is a difficult task, certainly when during training phase we could not include a (clear) context. For example the root ܣܩܠ (SQL) means to take, whereas the derived noun ܥܣܩܠ (SWQL') means arrogance (taking to much). An assessment of the vocabulary of the corpus of a random set of Aramaic documents shows that these relationships are found. Concerning hapax legomena, words which occur only once in a given corpus, and comprise between 40-60% of the corpus according to Zipf's law, we find partly salient results. About 70% of hapax legomena in a Semitic resource contains a root which also occurs in other words, and which makes it likely to share enough n-grams with better-known words. A related drawback is that for unique roots, or for weak roots (which loose at least one characteristic consonant in most of the conjugations), will not achieve salient results. Possible ways for the future to counter this is to use a lemmatizer to discern the lexeme for weak roots.

Secondly, we find that many documents of potential historical importance have not yet been published and/or translated and edited. This is an important problem for Aramaic in particular, and historical languages in general. We currently lack the knowledge to know to which points of interest the unpublished text are relevant, because studying them manually requires too much time. However, the method of the FastText algorithm allows us to efficiently search a large amount of Aramaic texts, once they are digitised, based on the ability to query of semantic relatedness in a given collection of documents. This makes it possible to find the most relevant documents for a given query, which can then be further analysed, as previously explored by Levi et al. (2015).

The sole drawback we have discovered is that when we apply the FastText algorithm on place names, that it does not discover the fact that these substantives indicate a location. The same is true for personal names, making the identification of named entities difficult, also due to the fact that traditional methods of discerning named entitiess automatically, such as the fact that they start with a capital

letter, do not apply to Aramaic. Traditional word embeddings, which look at the context in which a word occurs, have a certain conception of the name indicating a person or a location, leaving the algorithm to discern a semantic correlation between for example Germany, Switzerland and France, on the sole basis of the words among which they tend to occur. This is one of the assets of taking words as the basic unit to assign vectorial representations, rather than n-grams. However, for modeling Aramaic we have a vast amount of named entities available through the Syriaca platform (Kiraz 2005), which through term matching allows to recognise the se named entities.

5. Conclusions and Areas for Further Research

As we have seen throughout this paper, the FastText algorithm has provided an important step towards the NLP treatment of the Aramaic language family. However, as with all unsupervised machine learning methods, it is difficult to discover salient methods of evaluating the algorithm, certainly if we want an objective standard to evaluate the quality of the in real-time created vectors for OOV words. Therefore, our main further research will involve evaluating the semantic relationships between words with work on the lexical semantics of Aramaic, which has already been done using traditional methods.

6. Bibliographical References

Bengio, Y., Ducharme R., Vincent P. and Janvin C. (2003). A neural probabilistic language model. *The Journal of Machine Learning Research* 3 (1): 1137-1155.

Bojanowski, P., Grave, E., Joulin, A. and Mikolov, T. (2016). Enriching word vectors with subword information. Retrieved from http://arxiv.org/abs/1607.04606.

Chen, X., Xu, L., Liu, Z., Sun, M. and Luan H. (2015). Joint learning of character and word embeddings. *Proceedings of the 24th International Joint Conference on Artificial Intelligence,* pp. 1236-1242.

Creason S. (2008). Aramaic. In R.D. Woodard (ed), *The Ancient Languages of Syria-Palestine and Arabia*. Cambridge University Press, 45-48.

Joulin, A., Grave, E., Bojanowski, P. and Mikolov, T. (2016). Bag of tricks for efficient text classification. Retrieved from http://arxiv.org/abs/1607.01759.

Kiraz, G.A. (2005). Computing the Syriac lexicon: historical notes and considerations for a future implementation. In A.D. Forbes and D.G.K. Taylor (eds.), *Perspectives in Syriac Linguistics I: Colloquia of the International Syriac Language Project*. Piscataway NJ, Gorgias Press, pp. 93-104.

Levi, O., Goldberg, Y. and Dagan, I. (2015). Improving distributional similarity with lessons learned form word embeddings. *Transactions of the Association for Computational Linguistics* 3(1), pp. 211-225.

Luong, T., Socher, R. and Manning C.D. (2013). Better word representations with recursive neural networks for morphology. *Proceedings of the 17th Conference on Computational Natural Language Learning,* pp. 104-113.

Mikolov, T., Chen, K., Corrado, G. and Dean, J. (2013). Efficient estimation of word representations in vector space. Retrieved from http://arxiv.org/abs/1301.3781.

Naby, E. (2013). From lingua franca to endangered language. The legal aspects of the preservation of Aramaic in Iraq. In J.A. Argenter and R. McKenna Brown (eds.), *Endangered Languages and Linguistics Rights on the Margins of Nations,* pp. 197-206.

Sabar, Y. (2002). A Jewish Neo-Aramaic dictionary: dialects of Amidya, Dihok, Nerwa and Zakho, Northwestern Iraq. Harrasowitz, Wiesbaden.

Wieting, J., Bansal, M., Gimpel, K. and Livescu, K. (2016). CHARAGRAM: embedding words and sentences via character n-grams. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pp. 1504-1515.*

Xu, P. and Fung P. (2013). Crosslingual language modeling for low-resource speech recognition. *IEEE Transactions on Audio, Speech and Language Processing* 21(6): 1134-1144.

7. Language Resource References

FastText (2016). Facebook AI Research Team, https://github.comfacebookresearch/fastText/