Natural Language Processing - Paper report

Júlia Togashi de Miranda

November 13, 2021

Abstract

This report is in the context of the class Natural Language Processing (DS-télécom-20), from the Data Science Master 2, 2021-2022. In the following, the paper entitled [BAAP20] *"A Comprehensive Analysis of Preprocessing for Word Representation Learning in Affective Tasks"* will be summarized and commented.

1 Main contributions of the paper

The paper [BAAP20] main goal is to analyze the effects of different preprocessing methods for three specific affective tasks, that are sentiment analysis, emotion classification and sarcasm detection. Its contribution explores multiple fields of the preprocessing, that goes from comparing the results for each method (such as steaming, negation, stop words removal, for example), following by the results of applying them and different orders and different combinations. At last, it also goes over the analyses for the preprocessing in training Corpora (denominated *Pre*) and for the preprocessing in the classification data (denominated *Pos*).

As mentioned in the article, multiple previous works explored the effects of preprocessing techniques. However, for the case when it's applied in the word-embedding, it is pioneer in the area affective tasks, which is relevant due to the fact that these analyses are heavily task-specific.

The authors are successful in making a didactic explanation for the different preprocessing methods analyzed and their specific intake in each implementation. They are summarized in the following (as can be observed, most of those have been tested in the course's practical sections):

- **Basic:** Includes removing html tags, numbers and lowercasing (*NLTK regexptokenizer*)
- **Spellcheck:** In this method, it is discussed the dichotomy between having a text that better represents the complexity of the natural language and having a text more appropriated for the classification text. (*Substitution made using frequency of self-made replacement vocabulary*)
- **Negation:** Consists of making the substitution of a negation for an antonym (*The negation is also done by searching in a self-made antonyms dictionary and replacing both the negation and the immediate following word* – the paper doesn't comment on the possibility of having a negative word that referees to multiple following subjects, and how this implementation could lead to mistakes in that case)
- **Parts-of-speech:** Only taking into account namely nouns, verbs, adjectives and adverbs, as they naturally hold the most important information in the text. As seen

in class, has the advantage of drastically decreasing the size of the vocabulary, thus resulting in more efficient computations. (*NLTK pos-tagger*)

- **Stop words:** Removal due to the fact that they are the most common words used in the language, thus not bringing a relevant information to our classification that and also reducing the corpus size (*also NLTK library*)
- **Stemming:** Reducing word to their root. (*NLTK Snowball*)

An important highlight can be also found on the order in which these processes have to be applied, that mostly follow common sense in the way that they interfere with each other. The conclusion in the best order should be the following: spellchecking, negation handling, pos classes, removing stop words, and stemming.

In the original article can be found relevant descriptions of the choices made to obtain the specific results, this includes: different training corpora used; the different embedding methods (which all have been seen in the theoretical courses: CBOW (ContinuousBag-of-Words), Skip-gram and BERT(Bidirectional Encoder Representations from Transformers)); the different evaluation datasets, which are different for every specific affective task; the choices of optimization (*ADAM*), loss function (*binary cross-entropy along with sigmoid activation*), model (*LSTM*) of the neural network used in the classification task and evaluation metrics (*F-score*).

To conclude, the paper reaches multiple interesting results. First, the negation preprocessing is found to be consistently the most effective mechanism on its own, followed by parts-of speech. At the same time, stop words, spellchecking and stemming didn't show considerable improvement. The best overall results though vary through datasets, some are obtained using all methods other than steaming and some all other than stop. Secondly, incorporating preprocessing into the training corpora outperforms preprocessing in classification datasets in all cases. Besides, applying both brings little improvement, which strikes even more the significance of the *Pre* reprocessing results. The results obtained in the paper are also compared to the state-of-the-art baselines in pretrained word embedding, this includes GloVe, SSWE, FastText, DeepMoji and EWE. They show that, in all cases, for some method, better than the ones using these pre-rained embeddings. At last, one last remark is that although the multi-class classification analyses, as expected, usually have the worst results, it shows the most significant improvement by using what is discussed in the work.

2 Analysis of the advantages and issues of the proposed method

Overall, the implementation of the paper doesn't show many possible problems. It is a straight forward approach, using already well stablished algorithms and methods, extensively studied in the literature. The way the results are described and progressively tested makes it easy to follow the methodology.

The only remark on the choices made is on the decision on how to implement negation. As described in the paper, in the example: "I am not happy today". First, the negation 'not' will be identified, then the negated word 'happy'. The antonym will be looked in the dictionary, resulting in the sentence "I am sad today". However, it is not specified how the case of a multiple negation (for example, "the movie didn't have a good script or good actors.") would be handled, or even, if it is not handled, how this impacts the results.

At last, now regarding the decision of the methodology, that is, extensively testing multiple approaches: In one hand, it can be considered advantageous, as it provides undeniable evidence, that is the objective of the work. However, sometimes it reaches already expected results: it varies from task to task, dataset to dataset. So, at some point, this extensive fitting might seem pointless, as it does not bring a concrete result (an answer: doing this is the best course of action). Yet, not having a concrete answer is still a relevant result, showing the reader the importance of extensively testing multiple approaches in natural language processing tasks.

3 Personal take on the interest, the contribution, and the relevance of the paper

Personally, I find that one of the main contributions of the paper is providing a didactic and easy to understand summary of multiple preprocessing methods for text related tasks. It successfully explains all methods seen in the course in a brief way and providing references for useful libraries for implementation. Thus, it can be used as guide for beguines in tasks even not related to affective tasks.

As for the results obtained, the felling I have is that in many points, specially in the analyses made related to which combinations of mechanisms wields the best results, the conclusion is the same found in previously available work: these analyses are heavily task-specific and even more, they can be case-specific. The results for example that usually negation is the most important preprocessing for affective tasks is may and should be taking into account. However, in a real-word application, where a lot of effort is made to obtain the best results, the best course of action is testing the most appropriate approach to the working dataset.

At last, one really important result, that unlike what was said before, has a great importance for different applications, is the ones involving the results preprocessing the text corpus and classification dataset. The results are strong enough to show that, at least for affective tasks, the baseline implementation should be applying it to either only the corpus or both.

Therefore, to conclude, I consider that the overall quality of the paper is good, making the subject interesting. However, its contribution with new research is limited. Nevertheless, this doesn't diminish its relevance in its conclusions, that might be used as base for future works, especially since it provides the implementation of the work.

4 Related works

The only mention of a future work in the paper is related to check if any other space of subsets of preprocessing factors yields more interesting results. There is, however, no evidence of this future work by the same authors. They are though still producing works related to the same field, for example [BDAP20] "Affective and Contextual Embedding for Sarcasm Detection", but that doesn't go into the testing of pre-processing mechanisms.

It can be observed that the paper was cited as reference 13 times from 1 July 2020 to the present date. Most of the works that make reference to it are related to studies regarding affective tasks, even though doesn't say much on preprocessing mechanisms. This supports the argument that the given results are mostly not that relevant, but are useful as a lose guideline for future works in the field.

5 Implementation Comments

The full implementation of the work can be found in the following Github of the author: https://github.com/NastaranBa/preprocessing-for-word-representation.

Although I went through the implementation to fully understand it, I didn't try to reproduce the results by myself. This is due to the fact that the specifications used to run all models is specified in the paper, and is really beyond the resources I have available. Even with this larger computational power, is described: "For a large model such as BERT, it takes up to 4-5 days for each run of the training." Thus, I considered that spending this large time on training wouldn't help improve my understanding of the work as much as only understanding the program.

As for the comments in the code, as already mentioned in this report, many of the preprocessing mechanisms used are similar to the ones tested on the first lab of the class, using the NLTK library tools. Another interesting remark is how much manual work was done: for example, defining manually a dictionary of contractions; a list of special and unknown characters. For the classification methods, instead of using Pytorch (second lab of the course), the library Keras was used, which seems to have a simpler and more straight-forward implementation.

References

[BAAP20] Nastaran Babanejad, Ameeta Agrawal, Aijun An, and Manos Papagelis. A comprehensive analysis of preprocessing for word representation learning in affective tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5799–5810, Online, July 2020. Association for Computational Linguistics.

[BDAP20] Nastaran Babanejad, Heidar Davoudi, Aijun An, and Manos Papagelis. Affective and contextual embedding for sarcasm detection. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 225–243, Barcelona, Spain (Online), December 2020. International Committee on Computational Linguistics.