Aggression Identification and Multi-Lingual Word Embeddings

Thiago Galery
Idio Ltd / The Cursitor,
London,
WC2A1EN
thiago.galery@idio.ai

Efstathios Charitos
Idio Ltd / The Cursitor,
London,
WC2A1EN
stathis.charitos@idio.ai

Ye Tian
Amazon Research / Cambridge, UK
Laboratoire de Linguistique Formelle
Universit Paris Diderot / Paris, France
tiany.03@gmail.com

Abstract

The system presented here took part in the 2018 *Trolling, Aggression and Cyberbullying* shared task (Forest and Trees team) and uses a Gated Recurrent Neural Network architecture (Cho et al., 2014) in an attempt to assess whether combining pre-trained English and Hindi *fastText* (Mikolov et al., 2018) word embeddings as a representation of the sequence input would improve classification performance. The motivation for this comes from the fact that the shared task data for English contained many Hindi tokens and therefore some users might be doing *code-switching*: the alternation between two or more languages in communication. To test this hypothesis, we also aligned Hindi and English vectors using pre-computed SVD matrices that pulls representations from different languages into a common space (Smith et al., 2017). Two conditions were tested: (i) one with standard pre-trained fastText word embeddings where each Hindi word is treated as an OOV token, and (ii) another where word embeddings for Hindi and English are loaded in a common vector space, so Hindi tokens can be assigned a meaningful representation. We submitted the second (i.e., multilingual) system and obtained the scores of 0.531 weighted F1 for the EN-FB dataset and 0.438 weighted F1 for the EN-TW dataset.

1 Introduction

This paper details a system demonstration for the 2018 *Trolling, Aggression and Cyberbullying* shared task (Kumar et al., 2018). The challenge was to provide a system capable of predicting the true class of a social media post which would be either one of (i) *Non-Aggression* (NOAG), (ii) *Covert Aggression* (CAG), or (iii) *Overt Aggression* (a multi-class but not a multi-label text classification problem). The shared task had two datasets available, one in English and another in Hindi, that were split into test / validation subsets. Systems were benchmarked against two datasets, one sourced from facebook (FB) and another from twitter (TW) for each language (EN or HI).

Despite the fact that data was divided by language, we noticed that some Hindi vocabulary appeared in the English data. This raised an interesting question. Could such vocabulary be conveying information regarding the aggressive or polite nature of the content? Considering words from other languages as mere noise could have unwanted consequences, as they could be instances of *code-switching*: situations in which terms from two or more languages are used in a message to better convey the speaker's intent. For example, by mixing Spanish and English, a speaker who says 'La onda is to fight y jambar' means something similar to 'The in-thing is to fight and steal' (Woolford, 1983). In these cases, understanding the switched word (in this case, 'fight') is essential to understand the sentence's meaning.

Against this background, we have decided to test the following hypothesis. Does the inclusion of word representations from other languages (say, Hindi) increase the classification score of a given shared task's dataset? Although this is applicable to both English and Hindi datasets, we decided to focus our experiments on the English one. This is partially due to time / effort reasons and logistics around the registration for having access to the data.

To test our hypothesis, we used a classification pipeline that would run two conditions. In the *single language condition* we simply regarded foreign (Hindi) words as OOV tokens. In the *multi language condition*, we aligned English and Hindi *fastText* (Mikolov et al., 2018) vectors via SVD matrices that

Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying, pages 74–79
Santa Fe, USA, August 25, 2018.

pull representations from different languages into a common space (Smith et al., 2017). We will discuss some of the background behind these decisions in the next section.[1]

2 Related Work

2.1 Out-of-Vocabulary information

The idea behind using the meaning of words from different languages to improve classification results is somewhat similar to the motivation for capturing emoticons and misspellings in text classification tasks.

The use of emoticons in text classification has been extensively discussed in sentiment analysis (Hogenboom et al., 2013; Hu et al., 2013; Mozetič et al., 2016; Novak et al., 2015; Thelwall et al., 2010; Zhao et al., 2012). Some have defended that the use of emoticons as a cue for sentiment analysis of tweets results in better accuracy compared to using the linguistic text alone (Hogenboom et al., 2013; Hu et al., 2013). Although emoticons tend to be a better indicator for an overall negative tweet than a positive one, their meaning depends on the context, therefore they do not always act as a direct indicator of emotions (Tian et al., 2017).

Building on this literature, we assumed that emoticons could be indicators of the aggression level of a specific social media post. Borrowing from previous work (Eisner et al., 2016), we have generated embeddings for emoticons based on their unicode descriptions and used these representations (Emoji2Vec) in our classification pipeline.

These are not the only out-of-vocabulary signal that might improve classification results discussed by the literature. Given that online abuse unfortunately happens frequently, companies that manage message boards started employing automatic detection methods in order to prevent abuse from happening in message boards and social media platforms. Because these detection mechanisms initially relied on a list of offensive words that would ban posts to be published, many users have changed their writing style in order to bypass such forms of abuse prevention. For example, a message containing 'kill yourself asshole' would be changed into 'kill yrslef a$$hole'. Given that this type of intentional misspelling is very hard to predict or blacklist and yet fundamental to the abusive nature of the message, our system needs to be robust against this kind of stylistic variation (Nobata et al., 2016).

Dense word embeddings, such as *Word2Vec* (Mikolov et al., 2013b) and *Glove* (Pennington et al., 2014), have been used as the *de facto* representations for the meaning of words in complex NLP pipelines. However, these representations do not incorporate information about meaning at the sub-word level, which means that we either get the vector for a word if it exists in the vocabulary, or some random (or zero based) representation. As a consequence, using pre-trained Word2Vec or Glove representations would not allow to capture the kind of creative misspelling just mentioned (e.g. 'a$$hole' would be considered an OOV token). To bridge this gap, a new type of representation has been motivated, namely *fastText* (Mikolov et al., 2018). The main advantage of this type of model is that it allows the retrieval of vectors for character n-grams in a way similar to word lookups. This means that we can devise a method for retrieving the vector of a systematically misspelled word by exploring its character level embeddings.

In conclusion, the discussion in this section motivates a better treatment of two forms of OOV tokens: (i) emoticons and (ii) systematically misspelled words. We will discuss how both are captured in our system when defining the classification pipeline.

2.2 Text Classification Architecture

In order to solve the classification challenge set out by the shared task, we need to employ some supervised machine learning model. Recurrent Neural Networks have been considered a natural match for sequence modelling because they output a probability distribution over the next element of the sequence given the current one) (hence a candidate for capturing sentence meaning). Despite of this, some (Bengio et al., 1994) have showed that they are difficult to train because their gradients tend to vanish or explode.

In order to avoid these problems, Cho et al (Cho et al., 2014) proposed a unit whose activation employs gating functions that determine whether the hidden states should be updated or reset. This allows the network to adaptively 'forget' irrelevant aspects of the data and 'remember' long term information.

[1]The implementation of this experiment is available at `https://github.com/tgalery/geiger`, see *notebooks*.

They proposed this technique in the context of machine translation, but they also showed that the projected embeddings learned by the network captured semantic and syntactic relationships of phrases and sentences (Van Der Maaten, 2013). Therefore, we decided to employ a GRU network for text classification, as the target class might depend on phrasal-like aspects of meaning (for example, an aggressive word's meaning could be neutralised under the scope of negation).

3 Methodology and Data

3.1 Classification pipeline

Our general classification pipeline is defined as follows. After tokenization and preprocessing, the token sequence would be passed to a token identifier, which would determine whether the token is (i) an emoticon, (ii) a foreign (Hindi) token, or (iii) a standard (English) token.

Emoticons would be assigned a vector by averaging the vectors of the tokens of its unicode description. The treatment of foreign tokens would be split into two conditions: (i) an OOV vector (*Single Language Condition*), (ii) aligned fastText vector (*Multi Language Condition*, to be detailed below). Finally, standard (English) tokens would be processed as follows. If a embedding could not be found for a word, we would generate character trigrams from it and average the character-level vectors. In summary, the processing pipeline would look like this.

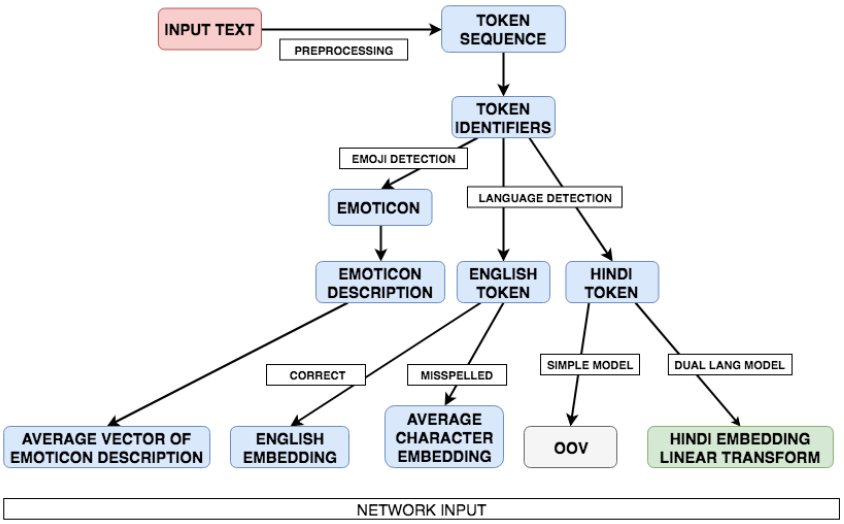


Figure 1: Text Pipeline

The token sequence would then be padded and clipped to maximum length of 200 tokens. Then, each token would be then mapped to its corresponding word embedding given the details above (Embedding Layer). This layer is followed by a 1D spatial dropout layer (value is 0.2), which is then piped into a Bidirectional Gated Recurrent Unit layer. This layer is then followed by a pooling layer that concatenates an 1D global average pooling and global max pooling. This would then be finally fed into a Dense layer whose dimensionality corresponds to the number of classes defined by the task: Non-Aggression, Covert-Aggression, and Overt-Aggression and whose activation is softmax. We selected the class with the maximum probability as the output. The model used categorical cross-entropy and Adam as an optimizer and was trained for 10 epochs. Schematically, we have:

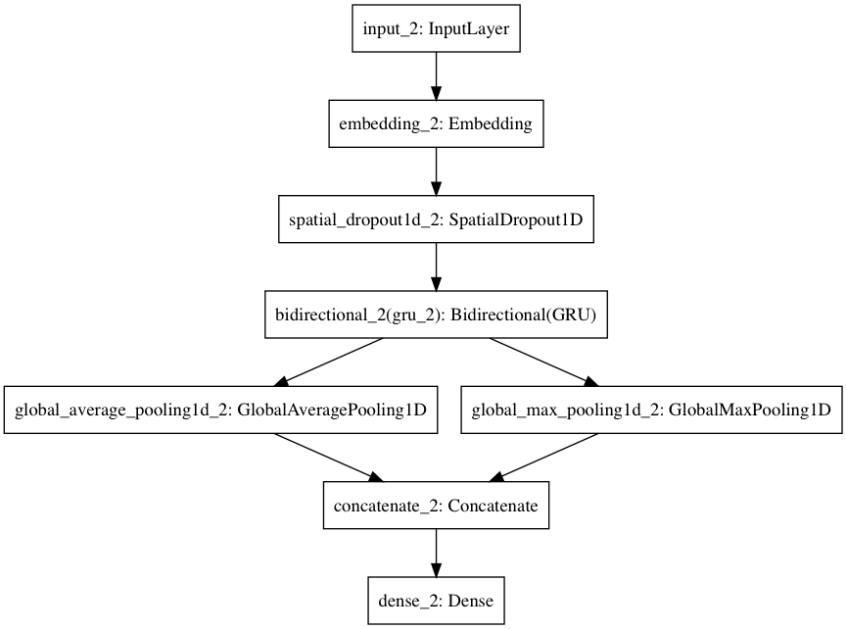


Figure 2: Network Architecture

3.2 Aligning Multi Lingual Vectors

At this point, it is important to discuss the alignment process to embed pre-trained English and Hindi embeddings within a single space. Combining pre-trained fastText word embeddings from English and Hindi generates a problem. Since they were trained by different corpora, the word distribution in each model does not necessarily align with the word distribution of similar words in the other model. In our previous example of code-switching, 'La onda is to fight y jambar.', we would like for the word embedding corresponding of 'fight' to be very similar to the word embedding of its Spanish counterpart (namely, 'luchar'), but that is not something that is guaranteed by our use of the pre-trained vectors.

To bridge this gap, we have borrowed from a technique that uses a Singular Value Decomposition (SVD) to learn a linear transformation capable of aligning difference vector spaces (Smith et al., 2017). This 'offline' way to project word embeddings contrasts the 'online' way in which embeddings from multiple languages are learnt together via bilingual corpora of aligned sentences (Mikolov et al., 2013a).

Building on previous work (Xing et al., 2015), Smith et al. (2017) argue that Mikolov's linear transformation is an orthogonal matrix (mapping a vector from *source* to *target* language should allow you to map that target vector back and obtain the original representation). To infer the orthogonal matrix, they form two ordered matrices X_D and Y_D from a bilingual dictionary (obtained via google translate), such that the ith row of both matrices corresponds to the source and target language word vectors of the ith pair in the dictionary. This allows SVD to be applied to both matrices (multiplying the transpose of one by the other) maximising the cosine similarity of translation pairs in the dictionary.

This somehow makes the meaning of 'fight' and 'luchar' in our example above to be more similar than they otherwise would be. We did not learn the orthogonal transform ourselves, but instead relied on a pre-learned representation built on top of *fastText* embeddings made available by Babylon.[2]. Thus, for the *multi language* condition, the fastText embeddings for English and Hindi were aligned after being loaded, making our embedding matrix consistent for both languages.

4 Results

With regards to our hypothesis, we did not find any striking difference between multi (weighted F1 0.58) and single language conditions(weighted F1 0.554) using validation data as a benchmark.

[2]Obtained through `https://github.com/Babylonpartners/fastText_multilingual`

There are multiple reasons for this. First, it could be that the foreign (Hindi) tokens were not indicative of the aggressive nature of the content. Another possibility is that the SVD method for aligning the geometry of embeddings in different languages did not provide good enough representations. In hindsight, we realized that the number of Hindi tokens that occur as part of the English dataset is too low, suggesting that *code switching* in not a deep characteristic of the data.

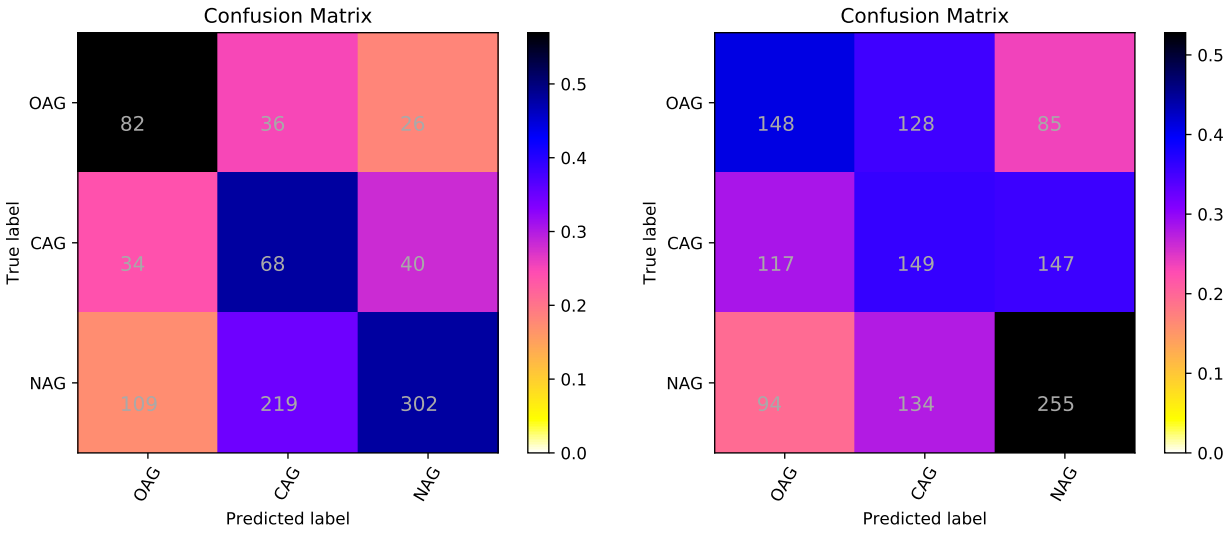
We were curious to see how our system would perform against others in the shared task, so we submitted results from the multi lingual version, taking the 24th place in the shared task's leaderboard. The results (and confusion matrices) were the following:

System	**F1 (weighted) EN-FB**	**F1 (weighted) EN-TW**
Random Baseline	0.3535	0.3477
Multi Lingual Pooled GRU	**0.5315**	**0.4389**

Table 1: Results for the English (Facebook) task and English (Social Media) task.

Table 2: Datasets confusion matrices

EN-FB **EN-TW**



For the facebook dataset it seems that the system provides much better answers than a random baseline. Looking at the confusion matrix it seems that our classifier is more comfortable with overt aggression and fairs worse with distinguishing covert aggression from non aggressive content.

5 Conclusion

It seems that for the shared task's dataset, the use of aligned multi lingual vocabulary did not improve the classification results. We do feel that our hypothesis has not been completely invalidated though. It would be interesting to test it against a dataset with more systematic forms of code-switching in order to draw better conclusions.

With regards to the architecture, we felt that the model had a problem when distinguishing between overt aggression and non aggression. Given this limitation, a better solution would involve a combination of two binary classifiers: (i) one distinguishing aggressive from non aggressive content and (ii) another distinguishing covert from overt types of aggression.

References

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166.

Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

Ben Eisner, Tim Rocktäschel, Isabelle Augenstein, Matko Bošnjak, and Sebastian Riedel. 2016. emoji2vec: Learning emoji representations from their description. *arXiv preprint arXiv:1609.08359*.

Alexander Hogenboom, Daniella Bal, Flavius Frasincar, Malissa Bal, Franciska de Jong, and Uzay Kaymak. 2013. Exploiting emoticons in sentiment analysis. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 703–710. ACM.

Xia Hu, Jiliang Tang, Huiji Gao, and Huan Liu. 2013. Unsupervised sentiment analysis with emotional signals. In *Proceedings of the 22nd international conference on World Wide Web*, pages 607–618. ACM.

Ritesh Kumar, Atul Kr. Ojha, Shervin Malmasi, and Marcos Zampieri. 2018. Benchmarking Aggression Identification in Social Media. In *Proceedings of the First Workshop on Trolling, Aggression and Cyberbulling (TRAC)*, Santa Fe, USA.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013a. Exploiting similarities among languages for machine translation. *arXiv preprint arXiv:1309.4168*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Tomas Mikolov, Edouard Grave, Piotr Bojanowski, Christian Puhrsch, and Armand Joulin. 2018. Advances in pre-training distributed word representations. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018)*.

Igor Mozetič, Miha Grčar, and Jasmina Smailović. 2016. Multilingual twitter sentiment classification: The role of human annotators. *PloS one*, 11(5):e0155036.

Chikashi Nobata, Joel Tetreault, Achint Thomas, Yashar Mehdad, and Yi Chang. 2016. Abusive Language Detection in Online User Content. In *Proceedings of the 25th International Conference on World Wide Web*, pages 145–153. International World Wide Web Conferences Steering Committee.

Petra Kralj Novak, Jasmina Smailović, Borut Sluban, and Igor Mozetič. 2015. Sentiment of emojis. *PloS one*, 10(12):e0144296.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Samuel L Smith, David HP Turban, Steven Hamblin, and Nils Y Hammerla. 2017. Offline bilingual word vectors, orthogonal transformations and the inverted softmax. *arXiv preprint arXiv:1702.03859*.

Mike Thelwall, Kevan Buckley, Georgios Paltoglou, Di Cai, and Arvid Kappas. 2010. Sentiment strength detection in short informal text. *Journal of the Association for Information Science and Technology*, 61(12):2544–2558.

Ye Tian, Thiago Galery, Giulio Dulcinati, Emilia Molimpakis, and Chao Sun. 2017. Facebook sentiment: Reactions and emojis. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media*, pages 11–16.

Laurens Van Der Maaten. 2013. Barnes-hut-sne. *arXiv preprint arXiv:1301.3342*.

Ellen Woolford. 1983. Bilingual code-switching and syntactic theory. *Linguistic inquiry*, 14(3):520–536.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. 2015. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011.

Jichang Zhao, Li Dong, Junjie Wu, and Ke Xu. 2012. Moodlens: an emoticon-based sentiment analysis system for chinese tweets. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1528–1531. ACM.