A Tweet Text Binary Artificial Neural Network Classifier

LSTM layer is placed after the Embeddings layer and on top of that, we have the previous MLP structure.

Finally, we employed another type of ANN capable of handling sequences - the Convolutional Neural Network (CNN). Here learning a sequence is achieved via a different mechanism which exploits the mathematical operation of convolution of the input sequence with a small kernel. We thus placed after the Embeddings layer two parallel layers with 32 kernels of length 5 each. The outputs of those parallel Convolutional layers are then merged and being fed into the previous MLP architecture.

To convert the continuous (between zero and one) ANN output to binary (i.e. flood-related input text or not) we use a threshold. Texts having output above the threshold are labelled as flood-related (i.e. one) and texts having output below the threshold as labelled zero. The threshold is chosen for each model separately by maximizing the F1-score. Finally, the text's class was assigned by a majority rule on the three models' output.

3 RESULTS AND DISCUSSION

3.1 Model setup and performance

After experimenting with various values, we ended up with a vocabulary of size 3000, sequence length of 40, embedding vector dimension of 300 and under-sampling ratio of 1.75. The vocabulary size and sequence length are small compared to typical Natural Language Processing (NLP) applications due to the short form of the tweet's text. The architecture of the ANNs used is described above.

ANNs were trained and evaluated individually on the same train/validation sets which were created by splitting the devset to an 80-20% ratio. The F1-scores on the validation set were 0.59 for the MLP, 0.60 for the RNN and CNN. Those scores were obtained by choosing thresholds 0.40, 0.65, 0.40 respectively. Finally, we combined the three ANN outputs by assigning to each input the majority class for the three ANN outputs. We chose this strategy, hoping that each ANN would perhaps capture different idiosyncrasies of the input. The overall F1 score improved slightly to 0.61. Our score on the test set was 0.5405, significantly lower, suggesting that we overfitted the training set.

3.2 Limitations of the study

The main challenge of the task was related to the labelling of the training dataset. We noticed that many samples looked flood-related from a visual inspection but were not labeled as such (some example ids are:940319294084202496, 944240672294531073, 950753737466830940, 1059017654088790018, 1055172135587536896). Further, we noticed that many positive samples are from meteorological alerts. This could maybe restrict the training set and explain the difficulties of the model in generalizing well and thus, influence the overall model performance.

3.3 Outlook - Ways to improve the performance

Experimenting with simpler text representations such as Bag of Words (BOW) and Term Frequency Inverse Document Frequency (TF-IDF) vectors and a Logistic Regression classifier revealed that taking into account tweet entities such as hashtags, in addition to the plain text, improved predictive performance.

However, due to time limitations, this approach was not implemented in our ANN framework. Further, it would require more sophisticated tokenization schemes able to extract hashtags, than those used for the ANNs input.

Geographical information of tweets, either in the form of metadata (e.g. coordinates, place attribute) or location mentions in the tweet's text could be exploited to 'geo locate' the tweet and possibly be used as additional inputs to the model. Especially since the dev. set focuses on a particular study area [1].

Finally, let us mention that this study focused solely on the tweet's text without considering the associated image. A two-branch model, where one branch would be the model presented here excluding the output layer and the other branch an image classifier both feeding the same output layer could be used to handle both text and image input.

3.4 Code availability

The model was implemented as a Google Colab Ipython notebook and code is available upon request (theo_nikoletopoulos@yahoo.co.uk).

REFERENCES

[1] Stelios Andreadis, Ilias Gialampoukidis, Anastasios Karakostas, Stefanos Vrochidis, Ioannis Kompatsiaris, Roberto Fiorin, Daniele Norbiato, and Michele Ferri. 2020. The Flood-related Multimedia Task at MediaEval 2020. In MediaEval 2020.

[2] Ian Goodfellow, Yoshua Bengio, and Aaron Courville: Deep learning. www.deeplearningbook.org

[3] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. & Dean, J. (2013). Distributed representations of words and phrases and their compositionality. Advances in neural information processing systems (p./pp. 3111--3119)