Inspiratron.org - Natural language processing, machine learning and cybersecurity

Might of the word embeddings

by Nikola Miloševi? - Wednesday, May 29, 2019

https://inspiratron.org/blog/2019/05/29/might-of-the-word-embeddings/

I have started using neural networks about two years ago, for TAC challenge, when colleague and I participated on ADR track of TAC. However, even thought I have given some contribution, pick of word embeddings and some of similar stuff fell on my colleague. The result was overall quite good and we have with our approach finished 4th.

However, recently, I have started working on a project that required me to do a full circle and make a whole model on my own. Neural networks proved to be easier to wrap head around than different versions of word embeddings. So what is the deal?

- Everyone talks about the use of neural networks and word emdeddings and how powerful they are. And yes they are mighty and powerful, but there is no free lunch. One should know that neural networks are slow learners. Therefore, training classification or NER network can take hours and a magnitude of time longer than some of the classical classifiers such as SVM, Naive Bayes, Decission trees or even CRFs. In case you want to train or fine-tune your embeddings, it may take days or weeks. Having expensive NVIDIA chipset with CUDA capability can help, but still it may take some time.
- There are different kinds of pre-trained embeddings, and in order for your method to work well, you have to pick well. What usually works is the one that is trained on the biggest dataset. Let me give an example. I am working on de-identification. So firstly, I created a CRF model with some bit of feature engineering. That performed with overall F1 score of 91%. Then I tried to make bidirectional LSTM-based model. According to literature, this kind of model should work better. I have tried to use GLoVE pre-trained model on Wikipedia (from 2014) and Gigaword 5 (news article). So the dataset that the embeddings were trained was quite large, containing 6 billion tokens, 400 000 word vocabulary. However, the best result for NER task I was able to get with about 100 epochs of training was about 76% F1 score. After couple of days of trying to improve this result, I have tried other embeddings. I downloaded GLoVE trained on common crawl, containing 840 billion tokens and about 2.2 million words in vocabulary. This immediately boosted the performance to 95% F1 score (translates to accuracy over 0.9999). In conclusion larger dictionaries give better performance. Even better if they are fitted to the text type. However, if the text is large enough, it may likely contain texts or text type you use.
- Hyperparameters still matter and there is a good deal of hyperparameter engineering going on. Once, I had an interview with a company that probably at the time started working with neural networks, and I got opinion from their CTO that they want someone who would be able to just make networks and throw data at them and they don't want to waste time on feature engineering. How people get blinded by news hypes! It is true, there is not much semantic feature engineering when it comes to neural networks. However, there are quite a few parameters to set that matter a lot and quite can change the results. Also, architecture matters a lot depending on a task that is performed. As I said, there is no free lunch, even with the new technology. You don't waste time on one thing, but you will have to waste it on something else or there will be something that will balance the costs.

I believe this would be 3 most valuable learning lessons so far from engaging more seriously to the neural language processing.