PROJECT MODULE REPORT: ARGUMENTATION MINING

UNIVERSITY OF POTSDAM, WINTER SEMESTER 2017/18

Oguz Serbetci
Maria Stazherova

Friday 23rd March, 2018

Abstract

Structure prediction on ”microtext” corpus. We used Keras Functional API throughout the project. The code can be found on GitHub under https://github.com/oguzserbetci/argmin2017

1 Data & Task

We worked with the arg-microtext corpus [1], which contains 112 short argumentative texts (originally in German and profesionally translated to English). Later we received preliminary annotations of the new microtexts and could incorporate them into the project.

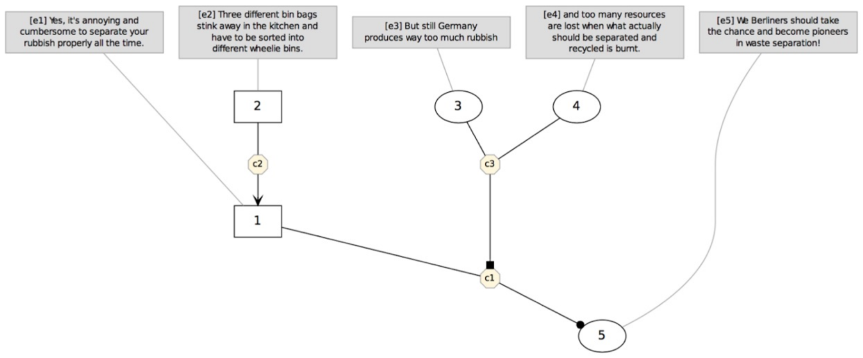
[e1] Yes, it's annoying and cumbersome to separate your rubbish properly all the time.
[e2] Three different bin bags stink away in the kitchen and have to be sorted into different wheelie bins.
[e3] But still Germany produces way too much rubbish
[e4] and too many resources are lost when what actually should be separated and recycled is burnt.
[e5] We Berliners should take the chance and become pioneers in waste separation!
2
3
4
c2
c3
1
c1
5

Figure 1: One example from arg-microtext corpus and its argumentation graph. Round nodes are proponent’s nodes, square ones are opponent’s nodes. The arcs connecting the nodes represent different supporting and attacking moves.

As can be seen in the figure 1, each text in the corpus is a single tree and argument components (ACs) are either claims, or premises. Each text in our corpus had only one claim. We decided to concentrate on two tasks: classifying the type of an AC (claim or premise), and determining the links between ACs.

2 Approach

2.1 Inspiration

Having found the paper “Here’s My Point: Joint Pointer Architecture for Argument Mining” by Potash et al. [2], we agreed that it would be good to try and reproduce the architecture and results from this paper, especially because the authors claimed they achieved state-of-the-art results. The authors used a seq2seq (sequence-to-sequence) architecture with attention, called Pointer Networks, which they then extended to a “joint model” (architecture can be seen in figure 2) which was able to simultaneously address the link extraction task and the classification of argument components.

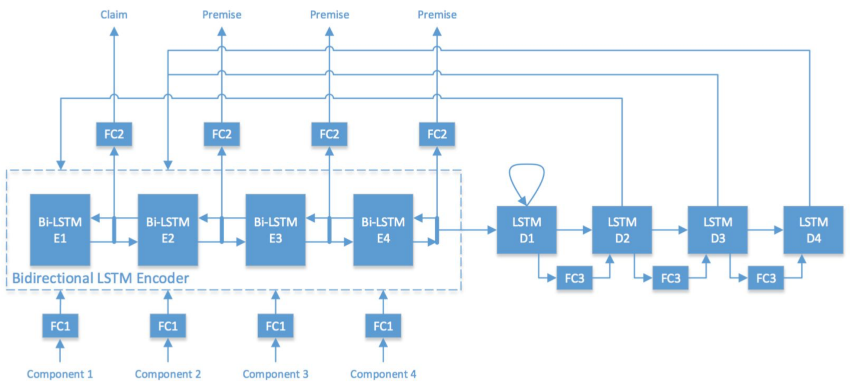

Figure 2: Architecture of joint model from the paper by Potash et al. [2], applied to an example of argument structure with four ACs.
D1 pointing to itself denotes that it has no outgoing link and is the head of a tree.

2.2 Workflow

We started our work by doing the data preprocessing and creating simple bag-of-words vector representations of argument components. After that we implemented a pointer network with one-directional encoder for the single task (determining the links between ACs). We sticked to the regularization techniques and their strength, as it was applied in the paper [2].

Our next step was to implement the fully-connected inputs (add a fully connected layer to encoder input, as well as to the decoder input), to make the LSTM bidirectional and to use all applicable to our corpus features from the paper: structural feature (binary, is the AC the first one in text or not), bag-of-words over the whole data and embedding representations (average, min and max pooling of word vectors of tokens).

After that we moved further on implementing the joint model architecture with an additional task of classifying the type of AC (claim or premise). The diagram we made for better understanding of the architecture can be seen in figure 3.

We implemented 5-fold cross-validation for hyperparameter tuning and evaluation.

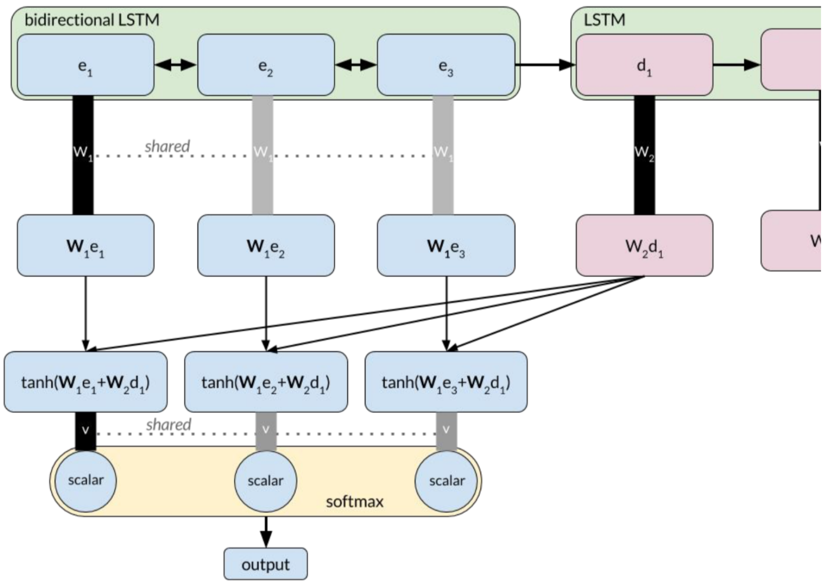

Figure 3: Essential part of our architecture of the joint model

2.3 Challenges

While working on the project, we encountered some difficulties and made some errors that we then corrected. The first one was to think of the proper way to pad input sequences so that they have the same length. When we implemented the joint model, we decided to filter out examples that were longer than 6 in length, because of the distribution of document length in the corpus, which can be seen in the figure 4. Documents were padded with zeros, e.g. link conversion would look like following: from 1,1,2 we would get [1,0,0,0][1,0,0,0][0,1,0,0][0,0,0,0].

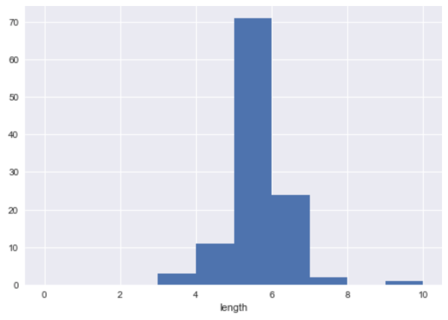

Figure 4: Distribution of document length in the corpus

3 Results

Intermediate 5-fold cross validation on the joint model yielded **0.603(+/-0.086)** validation accuracy using all features.

results!

4 Conclusion

future ideas and improvements

References

[1] Andreas Peldszus and Manfred Stede. An annotated corpus of argumentative microtexts. In *Proceedings of the First Conference on Argumentation, Lisbon, Portugal*, June 2015.

[2] Peter Potash, Alexey Romanov, and Anna Rumshisky. Here's my point: Joint pointer architecture for argument mining. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1364–1373, 2017.