A Parallel Hierarchical Attention Network for Style Change Detection

Notebook for PAN at CLEF 2018

Marjan Hosseinia and Arjun Mukherjee

University of Houston
mhosseinia@uh.edu, arjun@cs.uh.edu

Abstract We propose a model for the new problem of style change detection. Given a document, we verify if it contains at least one style change. In other words, the task is to investigate whether it is written by one or multiple authors. The model is composed of two parallel attention networks. Unlike the conventional recurrent neural networks that use the character or word sequences to learn the underlying language model of documents, our model focuses on the hierarchical structure of the language and observes the parse tree features of a sentence using a pre-trained statistical parser. Besides, our model is independent of style change positions although they are given during the training phase. The reason is to have a more applicable approach to the real world problems where such information is not available. PAN 2018 results show that it achieves 82% accuracy and stays at the second rank.

1 Introduction

Given one document, the problem of style change detection is to find if the writing style of the document has changed. In other words, we investigate whether it is written by multiple authors or one author. This is relatively a new problem in the area of mining writing style of text documents and is similar to the authorship verification and attribution problems where the task is to decide *who* has written them by comparing *how* they are written. Similarly, we need to study the writing style of a document by focusing on its linguistic aspects. Actually, the writing style expresses the selection of words and (grammatical) structure of a sentence.

According to the literature, there are two frequent NLP approaches for document representation. First, the bag-of-words model that is independent of the word order of a sentence and expresses the *word selection* [5]. Second, the sequence models such as word embedding that are sensitive to the *order* of words of a sentence [6]. Although the English language is recognized to have a latent hierarchical, tree-based structure [1], none of the two approaches deploy the hierarchical structure of a sentence for its representation.

In the problem of style change detection, we consider the latent structure of English language to represent a sentence. We use a context-free grammar parser to predict the tree-based structure, Parse Tree, of a sentence. Then, we extract the ordered features of

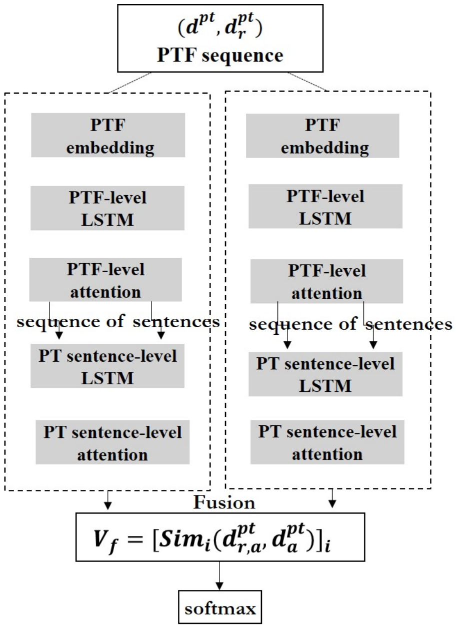

Figure 1. Parallel hierarchical attention network architecture

a parse tree to be used by a parallel hierarchical attention network to find the determinative parts of a sentence and a document. Finally, a fusion layer is used to compare a document with its reverse version to predict the class label. The results show that our model achieves promising results for PAN 2018 dataset.

The model is described in details in Section 2. In Section 3 we provide the results and discussion.

2 Parallel Hierarchical Attention Network

Our model is inspired by the two successful neural network architectures in authorship verification and document classification. Similar to [2] it has a parallel structure: two columns of recurrent neural networks, one fusion layer, and one softmax layer. However, each column is not a simple RNN anymore but is a hierarchical attention network with two levels of attention mechanism proposed in [8]. To be for specific, each column includes a Parse Tree Feature (PTF) embedding, a PTF-level LSTM, a PTF-level attention, a PT sentence-level LSTM, and a PT sentence-level attention layer. Here, the key difference is that the LSTM input is not the conventional character/word sequence but is the sequence of Parse Tree Features (PTFs) extracted from the tree-based structure of a sentence. The model architecture is shown in Figure 1. Each part will be described in the following sections.

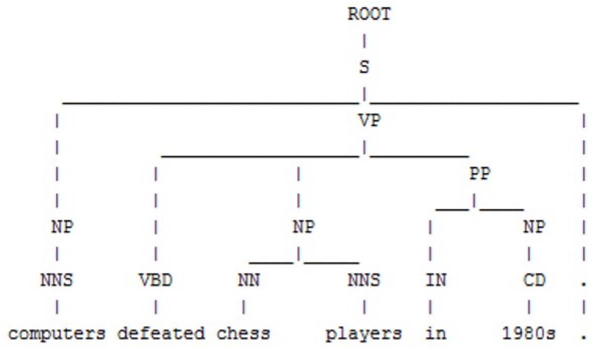

Figure 2. parse tree for “Computers defeated chess players in 1980s.”

2.1 PTF Embedding

We use Stanford PCFG parser[1] to retrieve the hierarchical structure of a sentence [4]. Figure 2 shows the underlying parse tree of sentence *“Computers defeated chess players in 1980s.”*. To use this tree structure by the following LSTMs in our model we need to preserve the word sequence of a sentence. We define Parse Tree Feature (PTF) of each word in a sentence as a path starting from the root to the corresponding leaf (word) of its parse tree. The path is a set of all rules of the form $parent \rightarrow child_1...child_n$ from root to a leaf (word). Here, punctuation marks are considered as word unigrams. For example, the PTF of “computers” consists of three rules and is $[S \rightarrow NP\,VP\,., NP \rightarrow NNS, NNS \rightarrow computers]$ and PTF of “chess” with four rules is $[S \rightarrow NP\,VP\,., VP \rightarrow NP, NP \rightarrow NN, NN \rightarrow chess]$ (Figure 2). We ignore rule $ROOT \rightarrow S$ because it is a shared rule among all PTFs. Accordingly, the Parse Tree (PT) representation of a sentence is the set of PTF of all its word unigram. For the above example, s has seven PTFs where $PTF_{computer}$ is the first and $PTF_{.}$ is the last feature i.e., $s = [[S \rightarrow NP\,VP\,., NP \rightarrow NNS, NNS \rightarrow computers], ..., [S \rightarrow ., . \rightarrow .]]$

Let $d^{pt} = [s_i | i \in [0, n]]$ be Parse Tree (PT) representation of document d with n sentences where $s = [PTF_j | j \in [0, l_s]]$ is PT representation of sentence s with size l_s. As we mentioned earlier, we preserve the order of sentences and words according to their occurrence in a document. We define d_r^{pt} to be the reverse PT representation of d^{pt}. To make d_r^{pt}, first, we reverse the order of sentences of d^{pt} then the order of PTFs of each sentence. In other words, the last PTF of the last sentence of d^{pt} is the first PTF of d_r^{pt}. So, $d_r^{pt} = [s_{r,i} | i \in [n, 0]]$ where $s_r = [PTF_j | j \in [l_s, 0]]$ is the reverse of s. In our example, $s_r = [[S \rightarrow ., . \rightarrow .], ..., [S \rightarrow NP\,VP\,., NP \rightarrow NNS, NNS \rightarrow computers]]$.

[1] https://stanfordnlp.github.io/CoreNLP/

2.2 LSTMs and Attention Mechanism

Here, the PT document representation and its reverse (d^{pt}, d_r^{pt}) are the inputs of our model, each for one of the two columns. Later, we will explain why the PT reverse version of a document is one of the two inputs. Each column is a hierarchical attention network proposed in [8]. It has two layers of LSTM each followed by an attention layer. However, the input of the network is no longer word unigrams but is the PTFs. Besides, we use unidirectional LSTM instead of a bidirectional as we feed the reverse version of a document, almost similar to the backward pass of a bidirectional LSTM, to the second column of the network.
and includes a Parse Tree Feature (PTF) embedding, a PTF-level LSTM, a PTF-level attention, a PT sentence-level LSTM, and a PT sentence-level attention layer.

Parse Tree Feature Embedding

For a sentence $s = \{PTF_1, ..., PTF_{l_s}\}$ of l_s PTFs, we embed PTFs using PTF matrix embedding W that is initialized randomly and learned during the training phase. Here, $x_i = WPTF_i$ is the PTF embedding of the *ith* feature.

PTF-level LSTM and PTF-level Attention

We choose LSTM in our model that is known to achieve promising results for long-term dependencies while its forget and update gates control the flow of information effectively. Here, h_i is the LSTM hidden state after *ith* feature of sentence s. Although we believe that PTFs express the writing style(s) of a given document, some are more determinative than others in predicting the class label. To highlight the importance of each PTF, we apply the attention mechanism proposed in [8] to the hidden state of PTF-level LSTM at step i. This mechanism computes a weight vector α_i using a Multi-Layer Perceptron and a softmax function. Here, W_{pt} and b are weight matrix and bias vector respectively, u_{pt} is a random context vector and will be adjusted during the training phase and s_a^{pt} is the sum of weighted hidden states of sentence s:

$$u_i = \tanh(W_{pt} h_i + b) \tag{1}$$

$$\alpha_i = \frac{\exp(u_i^\top u_{pt})}{\sum_i \exp(u_i^\top u_{pt})} \tag{2}$$

$$s_a^{pt} = \sum_i \alpha_i h_i \tag{3}$$

PT Sentence-level LSTM and PT Sentence-level Attention

The sequence of weighted sentences (s_a^{pt}) are the inputs of the PT sentence-level LSTM. Again, we would like to find which part of a document is more important for classification. In other words, we need to know in which sentence the style of the document is changed significantly. So, the PT sentence-level attention layer is applied to the hidden state of the PT sentence-level LSTM [8]. Similarly, W_s and $b^{'}$ are weight matrix and bias vector respectively, u_s is a random context vector and will be adjusted during the training phase, β_j is the weight vector and d_a^{pt} is the final weighted document vector:

$$u_j^s = \tanh(W_s h_j^s + b^{'}) \tag{4}$$

$$\beta_j = \frac{\exp(u_j^{s^\top} u_s)}{\sum_j \exp(u_j^{s^\top} u_s)} \tag{5}$$

Table 1. Similarity functions. a, b: document vectors, n: number of features in a and b

Metric	Description	Metric	Description
Chi2 kernel	$\exp(-\gamma \sum_i [\frac{(a_i-b_i)^2}{(a_i+b_i)}])$	Cosine similarity	$\frac{ab^T}{\lvert\lvert a\rvert\rvert \lvert\lvert b\rvert\rvert}$
Euclidean	$(\sum_i (a_i-b_i)^2)^{0.5}$	Linear kernel	$a^T b$
RBF kernel	$\exp(-\gamma \lvert\lvert a-b\rvert\rvert^2)$	Mean of L1 norm	$\frac{\sum_i^n \lvert a_i-b_i\rvert}{n}$
Sigmoid kernel	$\tanh(\gamma a^T b + c_0)$		

$$d_a^{pt} = \sum_j \beta_j h_j^s \tag{6}$$

The reverse version of a document passes the same process through the second column of layers simultaneously. At this step, we have two weighted document vectors, the original (d_a^{pt}) and its reverse version $(d_{a,r}^{pt})$, from two parallel columns of layers. Next, we explain how to use the two document vectors for classification.

2.3 Fusion and Output

The last and important step is to investigate the style change in a document. In this problem, the number of authors is unknown to us and it is an open set problem with respect to the authors. Indeed, learning writing styles and observing a change may not be applicable solely. So, we need a mechanism independent of the number of authors/writing styles. Here, learning the *difference* between the two versions of a document is the key to find the existence of a style change. To do so, we use the fusion layer of our previous work [2] where several similarity functions compute the similarity/difference between a pair of documents in a fully connected neural network layer. The similarity functions are listed in Table1. The weighted document vector and its reverse version $d_a^{pt}, d_{r,a}^{pt}$ are compared in the fusion layer to learn the existence of a style change in documents by different authors and with various writing styles:

$$V_f = [sim_i(d_a^{pt}, d_{r,a}^{pt})]_i \tag{7}$$

where V_f is the similarity vector and sim_i belongs to one of the functions in Table1. For documents with no style change, it compares the language model of one author between the forward and backward pass. However, for documents by multiple authors, there are two possible cases. In the first case, the order of different writing styles differs in both versions of a document. For example, document d by three authors $d = [author_1, author_2, author_3]$ and its reverse $d_r = [author_3, author_2, author_1]$. In the second case, the order of writing styles is the same in both regular and the reverse version of a document. For example, $d = [author_1, author_2, author_1]$ and $d_r = [author_1, author_2, author_1]$. For both cases, the fusion layer compares the language model of multiple authors and the model learns the transition from one writing style to another. However, in the first case, the PT representation of the two documents are much more different than the second case as the order of authors differs in both versions of the document. Finally, the similarity vector V_f is given to a softmax function for binary classification.

Table 2. PAN2018 dataset statistics and results

	Train set	Validation set	Test set
Size	2980	1492	1352
Accuracy	100	83.78	**82.47**

3 Results and Analysis

We participate in PAN 2018 style change detection task [7,3]. The task contains two training and validation sets that are publicly available before the competition and one test set which is not visible to the participants and is used to evaluate the participating models including our parallel attention network. In the training phase, we use the negative log-likelihood as our loss function and RMSprop with learning rate $= 1e - 03$ as the optimizer. We initialize the 100-dimensional PTF embedding from a uniform distribution over $[0, 1)$. The size of the hidden layer of the two LSTMs is 8 with the batch size $= 1$. We also apply a dropout of 0.3 on the output of the fusion.

The accuracy and the size of each set are listed in Table2. It shows that our model achieves 82% accuracy on test dataset and 83.78% on the validation set and stays at the second rank in PAN 2018. As there is not much difference (less than 1.4%) between the accuracy of the two sets it indicates that the model is generalized well. Besides, our model is independent of style change positions although they are given during the training phase. The reason is to have a more applicable approach to the real world problems where such information is not available. To see the effect of utilizing PTFs in style change detection and fusion layer we do some experiments on PAN 2018 dataset.

PTF vs Word Unigram

To show that PTFs are effective elements to represent one's style of writing, we train our model using only word unigram features instead of PTFs. We keep all settings intact and take the advantage of Glove pre-trained word vectors [2] as the input of embedding layer. Here, the reverse of a document is created as before and contains the set of reverse sentences from the last to the first sentence of the document. Results show that the accuracy of the validation set[3] is 71%, almost 13% less than PTFs.

Fusion vs Concatenation

The effect of the fusion layer that includes several well-known similarity metrics can be addressed with ablation. We replace the fusion layer with a fully connected layer that takes the concatenation of the two weighted document vectors produced from the two columns of attention networks. The accuracy on the validation dataset reduced by 10%.

The downside of this method is its expensive pre-processing phase that results in producing a huge PTF embedding dimensionality. The size of PTFs of the training set is around $1,300,000$ compared to $70,402$ word unigram features which is almost 19 times larger. However, this huge dimensionality helps the attention mechanism to find and focus on discriminative features for class label prediction. [4] We believe the model

[2] https://nlp.stanford.edu/projects/glove/

[3] the test set was not released.

[4] Producing PTFs using Stanford standalone parser made it slow and took around 10 hours in PAN evaluation process (test phase). It is much faster if one uses CoreNLP server available at

can be improved if we feed the style change positions to the network in an appropriate way. Or instead of using a pre-trained tree structure of a sentence we can learn the latent hierarchical structure of a sentence.

4 Conclusion

We propose a model to solve the new problem of style change detection in PAN 2018. We use parse tree features to deploy the hierarchical structure of a sentence and extract them such that the order of the corresponding words will be preserved to be used by a parallel hierarchical attention network. The results show that our model achieves promising results although we do not use the style change positions for training the model and only rely on the raw text of the dataset.

Acknowledgments

This work is supported in part by NSF 1527364. We also thank anonymous reviewers for their helpful feedback.

References

1. Chomsky, N.: Syntactic structure. Mouton (1957)
2. Hosseinia, M., Mukherjee, A.: Experiments with neural networks for small and large scale authorship verification. arXiv preprint arXiv:1803.06456 (2018)
3. Kestemont, M., Tschugnall, M., Stamatatos, E., Daelemans, W., Specht, G., Stein, B., Potthast, M.: Overview of the Author Identification Task at PAN-2018: Cross-domain Authorship Attribution and Style Change Detection. In: Cappellato, L., Ferro, N., Nie, J.Y., Soulier, L. (eds.) Working Notes Papers of the CLEF 2018 Evaluation Labs. CEUR Workshop Proceedings, CLEF and CEUR-WS.org (Sep 2018)
4. Klein, D., Manning, C.D.: Accurate unlexicalized parsing. In: Proceedings of the 41st annual meeting of the association for computational linguistics (2003)
5. Landauer, T.K., Dumais, S.T.: A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. Psychological review 104(2), 211 (1997)
6. Mikolov, T.: Statistical language models based on neural networks. Presentation at Google, Mountain View, 2nd April (2012)
7. Stamatatos, E., Rangel, F., Tschuggnall, M., Kestemont, M., Rosso, P., Stein, B., Potthast, M.: Overview of PAN-2018: Author Identification, Author Profiling, and Author Obfuscation. In: Bellot, P., Trabelsi, C., Mothe, J., Murtagh, F., Nie, J., Soulier, L., Sanjuan, E., Cappellato, L., Ferro, N. (eds.) Experimental IR Meets Multilinguality, Multimodality, and Interaction. 9th International Conference of the CLEF Initiative (CLEF 18). Springer, Berlin Heidelberg New York (Sep 2018)
8. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.: Hierarchical attention networks for document classification. In: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 1480–1489 (2016)

https://stanfordnlp.github.io/CoreNLP/corenlp-server.html. However, we were not allowed to use any external resources during PAN evaluation phase.