Who Am I? Personality Detection Based on Deep Learning for Texts

Conference Paper · May 2018

DOI: 10.1109/ICC.2018.8422105

Xianguo Sun
Southeast University (China)
15 PUBLICATIONS **122** CITATIONS

SEE PROFILE

Xiaojun Shen
University of Missouri - Kansas City
73 PUBLICATIONS **904** CITATIONS

SEE PROFILE

Who Am I? Personality Detection based on Deep Learning for Texts

Xiangguo Sun, Bo Liu, Jiuxin Cao, Junzhou Luo
School of Computer Science and Engineering
Southeast University, Nanjing, China 211189
Email: { sunxiangguo, bliu, jx.cao, jluo }@seu.edu.cn

Xiaojun Shen
School of Computing and Engineering
University of Missouri-Kansas City, Kansas city, USA 64110
Email:shenx@umkc.edu

***Abstract*—Recently, personality detection based on texts from online social networks has attracted more and more attentions. However, most related models are based on letter, word or phrase, which is not sufficient to get good results. In this paper, we present our preliminary but interesting and useful research results to show that the structure of texts can be also an important feature in the study of personality detection from texts. We propose a model named 2CLSTM, which is a bidirectional LSTMs (Long Short Term Memory networks) concatenated with CNN (Convolutional Neural Network), to detect users personality using structures of texts. Besides, a concept, Latent Sentence Group (LSG), is put forward to express the abstract feature combination based on closely connected sentences and we use our model to capture it. To the best of our knowledge, most related works only conducted their experiments on one data set, which may not well explain the versatility of their models. We implement our evaluations on two different kinds of datasets, containing long texts and short texts. Evaluations on both datasets have achieved better results, which demonstrate that our model can efficiently learn valid text structure features to accomplish the task.**

***Index Terms*—computational personality, Big Five, deep learning, online social networks, NLP.**

I. Introduction

In the field of psychology, a generally accepted and mostly influential model to characterize and measure the degree of one's personality is the Big Five Model[1], which consists of five traits: Openness, Conscientiousness, Extroversion, Agreeableness and Neuroticism.

With the development of online social networks such as Twitter, Sina microblog[1] and so on, it is easy to obtain online social texts created by users. A number of articles[2], [3], [4] have suggested that there are strong connections between one's personality traits and their behavior observable from web text records. Therefore, personality detection based on texts from online social networks has a significant meaning for plenty of applications such as individual RecSys, mental diagnosis, human resources management and so on.

However, capturing truly useful features from texts that have close connection to user's personality is still a challenging task to explore. Although some researchers have made comprehensive research and found a number of useful features such as LIWC (Linguistic Inquiry and Word Count)[5], Mairesse[6], letter's level features[7], [8], responsive patterns[9] and so on, unfortunately, most their applications mainly rely on features of only letters, words, or phrases and could only get limited results, which implies the necessity to explore more useful features.

We have noticed, some researchers [10], [11] have started to use ensemble learning to integrate multiple kinds of features instead of just relying on one feature to improve performance. This is a correct direction we will follow in the future. Our research experience shows that it is extremely essential to explore more relevant features which collectively would more completely and accurately reflect the relations between one's personality traits and his original text data. As a matter of fact the structure of online texts created by users, has not been studied as thoroughly as other features obtained by traditional ways. Obviously, in order to develop an effective model for multiple features, we need first to learn, to analyze, and to model each single feature well.

In this paper, we attempt to push forward the study on structure of online texts and its relation to personality. Specifically, we present our research on three questions: (1) Whether a discovered structure from the texts of online social networks can effectively reflect one's personality? (2) How to make an appropriate definition for the structure of texts? (3) How can we capture this structural feature from online texts?

Recently deep learning methods have been introduced into personality prediction and have obtained good performance. Intuitively, the convolutional neural network (CNN) attempts to reconstitute the course of writing articles, while the recurrent neural network (RNN) tries to understand texts by imitating the reading process akin to humans. By integrating these two kinds of neural networks, we propose the 2CLSIM Model, a bidirectional LSTMs concatenated with CNN, to capture the structural features from texts. In addition, a concept of latent sentence group (LSG), is introduced to describe the abstract feature combination extracted from closely related sentences. Furthermore, to the best of our knowledge, most related works only conducted their experiments on one dataset, which may not well explain the versatility of their models. We did our evaluations on two datasets that are relatively heterogeneous. In both experiments our model has shown better results, which demonstrate that our model can capture the more advanced structural feature correctly and is effective to detect users personality traits. In summary, contributions of

[1] https://weibo.com/login.php

978-1-5386-3180-5/18/$31.00 ©2018 IEEE

this work are as follows:

- We have introduced the concept of latent sentence group (LSG) to model the structural features of texts at the level of sentences and use CNN to learn the features.
- Based on CNN and RNN, we have proposed a combined neural network model to implement our task. The results of the experiments show that our model outperforms those that only use single RNN or CNN.

The rest of this paper is organized as follows. We present related works in Section II. Then our model will be introduced in Section III. The experiments and evaluations will be presented in Section IV. After that, we come to conclusion and future work in Section V.

II. RELATED WORKS

Broadly, the data type used for this research includes texts, avatars[9], likes[12] and so on. For texts data, most researches treat the task as a peculiar kind of texts classification. Although great progress has been made in the field of text classification in recent years, personality detection from texts is still in its early stage. As previously mentioned, capturing useful features suitable for personality detection remains a challenging problem. Most previous works have mainly paid attentions to the features at character or word level. Authors of [7] used RNN for character level and word level to build hierarchical, vector representations of words and sentences for trait inferences. Qiu et al.[13] have analyzed the relationship between participants' words in Twitter and their personalities. They have found there are connections between personality characteristics and specific words used in tweets. Beyond some specific words, many researchers found that the psychological meaning of words can also reflect ones characteristics. One of the methods used to express this kind of relations is called LIWC (Linguistic Inquiry and Word Count)[14]. Xiaoqian Liu and Tingshao Zhu[11] have used LIWC to represent each tweet and extracted the main part by DFT (Discrete Fourier Transform). Then they have used stacked auto-encoders to implement unsupervised feature learning. In addition, some other researchers also found the effectiveness of using structures extracted from texts in personality detection. Honghao Wei et al.[9] have used CNN with 1,2,3-grams kernels to capture the structure. However, using fixed size for kernels would impose a limitation on long texts. Since their work is based on stacked generalization and texts data is only a part of the whole, the final output cannot demonstrate whether the simple CNN is effective. Navonil Majumder et al.[10] have used 3-dimensional convolution to learn the structure of an article. They encoded the essay from word level to sentence level. Then they tried to continue using CNN to construct an essay vector based on previous work. However, when aggregating sentence vectors into the document vector, they could not continue using convolution operation because of non-convergence. To cope with this problem, they have used the max pooling instead.

Inspired by the work of Siwei Lai el[15], we combine both RNN and CNN in this study to capture structural features but with the following improvements. First, different from their work, we use LSTM instead of simple RNN so that the model can get better performance in both short and long texts. Second, we have observed that most related works only consider features on character, word or sentence. We extend the structures to the LSG, which is derived from the group of related sentences.

III. 2CLSTM MODEL

In this section, we introduce our 2CLSTM model in detail. Subsection III-A overviews the model and present the process of information flowing. Following this process, some important components of the model will be presented in the subsections from III-B to III-E.

A. *Overview of the Model*

The architecture of our model is depicted in Figure 1, which includes word embedding, bidirectional LSTMs, CNN layers with 1,2,3-grams kernels and classification. The whole model mainly consists of two parts: 2LSTMs and CNNLSG. Basically, the first part focuses on assembling context and extracting word semantic features, while the second part attempts to learn features from sentences structure. Five traits are trained separately. As the output from CNNLSG, document vectors will be sent into the softmax layer and get the final category.

The process of a text takes the following steps. Firstly each word is embedded into word vector, which will be explained in section B, and then the context information is encoded by the first part of our model. We name it as 2LSTMs, which will be introduced in section C. In the second part of our model (CNNLSG), we use CNN to learn the structural feature LSG produced from the first part, which will be explained in section D. The final feature vectors produced by the second part (CNNLSG) will be sent into softmax to produce the final personality traits as section E presents.

B. *Word Embedding*

For word embeddings, we have used pre-trained word vectors from GloVe[16], which is an unsupervised learning algorithm for obtaining word vector representations. Each word was embedded into 100 dimensions. If a word doesn't exist in the GloVe word list, a random number in $[-0.25, 0.25]$ will be assigned across the whole coordinates. However, we strongly recommend to train your own word vectors from your datasets. We didn't do this because of our limited hardware and datasets available. You can also try word2vect model instead of GloVe, there is no too much difference for our task.[2]

C. *2LSTMs*

Original LSTM (Long Short Term Memory networks) is shown in Figure 2. Instead of a unit that only executes simple activation function, LSTM has internal self loop as well as

[2] Training process of word2vect with negative sampling is usually much longer than GloVe.

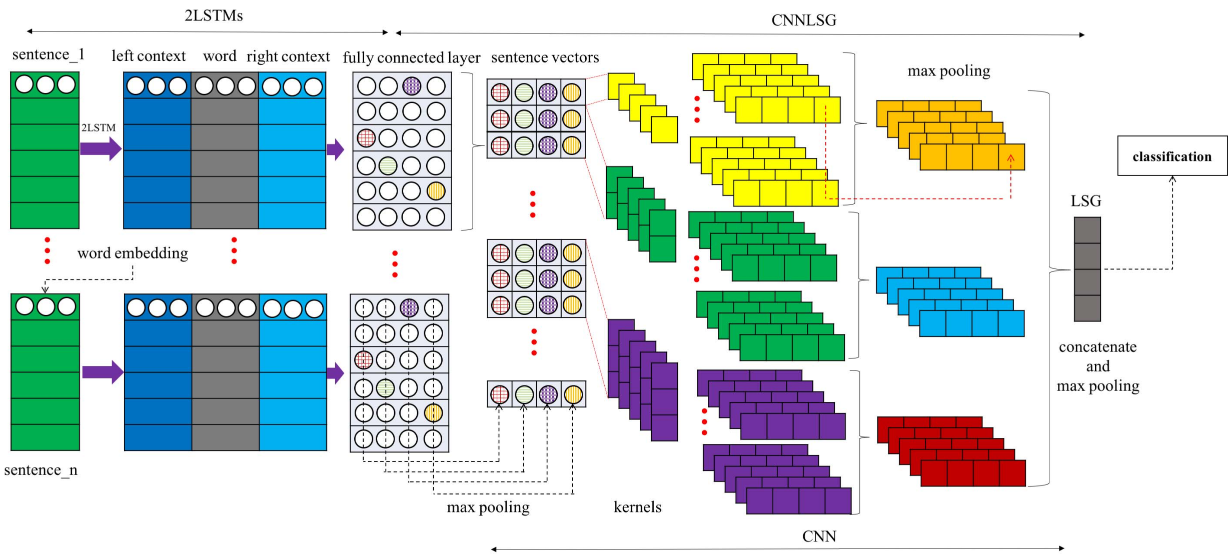


Fig. 1: The architecture of 2CLSTM

outer recurrence of the RNN. Each cell has more parameters and gates to control information flowing.

Specifically, LSTM can tolerate the unit forgetting left-context knowledge to some extent, which is realized by forget gate. The percentage of information leak is set between 0 and 1 via sigmoid function as follows:

$$f_i^{(t)} = \sigma\left(b_i^f + \sum_j U_{i,j}^f x_j^{(t)} + \sum_j W_{i,j}^f h_j^{(t-1)}\right) \tag{1}$$

where $\boldsymbol{x}^{(t)}$ is the input while $\boldsymbol{h}^{(t)}$ is the hidden layer vector. Meanwhile, LSTM also controls input knowledge absorption by input gate as follows:

$$g_i^{(t)} = \sigma\left(b_i^g + \sum_j U_{i,j}^g x_j^{(t)} + \sum_j W_{i,j}^g h_j^{(t-1)}\right) \tag{2}$$

which is similar to forget gate except using its own parameters. By filtering out the input and context information to some extent, the internal state is updated as follows:

$$\begin{aligned} s_i^{(t)} =& f_i^{(t)} s_i^{(t-1)} + \\ & g_i^{(t)} \sigma\left(b_i + \sum_j U_{i,j} x_j^{(t)} + \sum_j W_{i,j} h_j^{(t-1)}\right) \end{aligned} \tag{3}$$

The output $\boldsymbol{h}^{(t)}$ can also be shut off by the output gate as follows:

$$h_i^{(t)} = tanh(s_i^{(t)}) q_i^{(t)} \tag{4}$$

where $\boldsymbol{q}^{(t)}$ is calculated similarly to the forget and input gates:

$$q_i^{(t)} = \sigma\left(b_i^o + \sum_j U_{i,j}^o x_j^{(t)} + \sum_j W_{i,j}^o h_j^{(t-1)}\right) \tag{5}$$

In order to retain more messages from context, we use bidirectional LSTMs concatenated with current word. One of the advantages is that more structural features from context (both left-context and right-context) are concentrated within

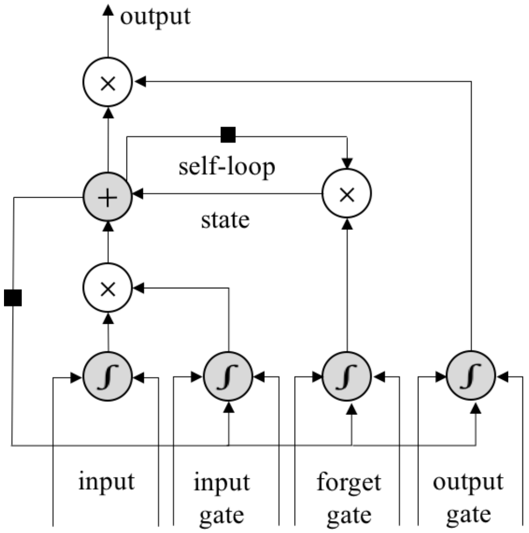


Fig. 2: LSTM cell

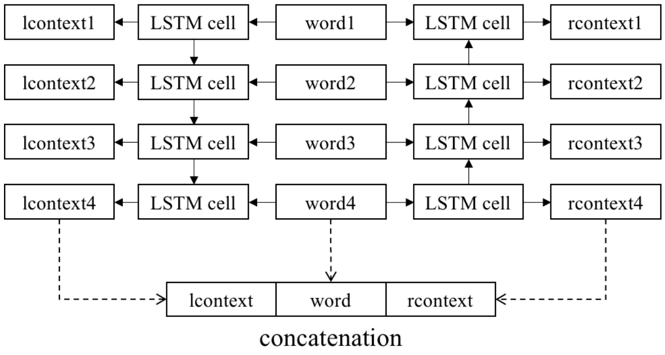


Fig. 3: The architecture of 2LSTMs

the current word, which lays the foundation of the following further feature extraction. We named this architecture as 2LSTMs, which is depicted as Figure 3.

D. Latent Sentence Group

The concept of sentence group derives from the linguistics. Basically, the sentence group means several successive sentences which are closely connected in logic and semantic structure such as coordinate relation, preference relation, causal relation and so on. However, to detect these concrete relations is impractical for most texts tasks. As a matter of fact, we often focus on the relationships between sentence vectors in some dimensions. These sentence relations are based on some specific vector dimensions, which means that is a ”latent” relationship. Therefore, we use the phrase, latent sentences group (LSG), to denote the abstract combination of sentence features. The LSG means several sentences in close relationship from the view of dimension. These sentences in an LSG don’t need to be strictly successive in spatial position. The definition of LSG is as follows:

Definition 1 (latent sentence group): Latent Sentence Group (LSG) is defined as a synthesis that consists of a number of sentence vectors which are closely connected in some coordinates.

We use CNN to learn LSG features. The details of this part of our model are shown in Figure 1. By means of the max pooling of each sentence, we get sentence vectors. Then in each dimension, we use 1,2,3-gram kernels to learn LSG features in each coordinate. After that, a dense layer and max pooling layer follow immediately, where the final vector will be produced and sent into the classifier.

E. Classification

Various classifiers can be used in classification part such as softmax, SVM, KNN or even Adaboost. In most deep learning models, softmax is one of the most widely used classifiers. Softmax receives feature vectors as input, and calculates the probability in each category as follows:

$$p(y=i|x;\theta)=\frac{e^{\theta_i^T x}}{\sum_j e^{\theta_j^T x}}, \tag{6}$$

where y is the classes, and x is the input vector. The class which gets higher score is the output of softmax. In our model, we use softmax as the final classification.

IV. Experiments

In this section, we present the experiments to evaluate the effectiveness of our model for personality detection.

A. Dataset

We use the following data sets in our experiments:

1) Stream-of-consciousness essays: We used stream-of-consciousness essay dataset from James Pennebaker and Laura King[17]. It contains 2,467 valid anonymous essays tagged with the authors personality traits: EXT (Extroversion), NEU (Neuroticism), AGR (Agreeableness), CON (Conscientiousness), and OPN (Openness).

2) YouTube: The YouTube personality dataset[3] comes from about 400 YouTube vloggers’ webcam videos. It contains speech transcriptions, genders and behavioral features translated manually from the videos. A main difference from the first kind of dataset is that most texts are shorter in this dataset. Another difference is that the labels here (personality impressions) are collected from the annotators rating impressions by watching each vlog, not from the authors own.

Remark: We choose these two data sets because we wish to test if our model can handle different cases. Table I shows a number of statistics we summarized from these two kinds of datasets. The mainly reasons why we choose these two datasets are as follows:

- Firstly, as depicted in Figure 4, most documents from YouTube are shorter than stream-of-consciousness essays, which can verify whether the 2CLSTM model is effective both for short and long texts.
- Secondly, the labels of stream-of-consciousness essays data come from the author’s own questionnaire, which can be explained as autognosis while the other dataset (YouTube dataset) can be treated as outer-perception because the personality labels of this dataset come from the volunteers watching the vlogger’s videos. Therefore, these two datasets can demonstrate that our model is valid in both cases, no matter whether the labels are generated by authors or by other people.

TABLE I: Macro statistics for datasets

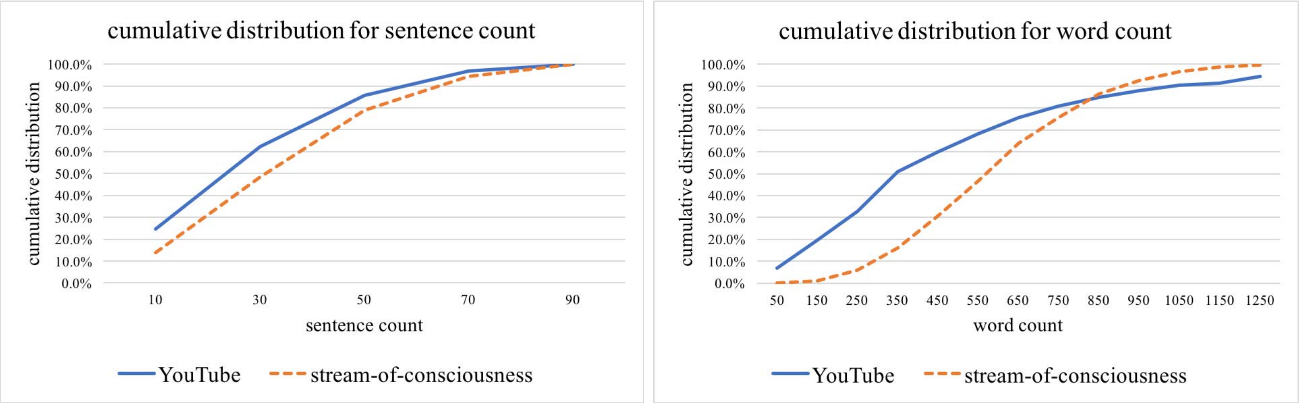
Property	stream-of-consciousness	youtube
average word count	648	526
max word count	2488	1972
min word count	33	36
average sentence count	46	41
max sentence count	327	147
min sentence count	1	2

B. Contrast Models

In this subsection, we choose five well-known models as our contrast models, which are listed in table III.

The first one is Bayes classification based on TF-IDF feature, which is one of the most basic and common methods for texts classification. TF-IDF is the product of term frequency (TF) and inverse document frequency (IDF). The value expresses the importance of some specific words distinguishing documents. However, in practice, words with too small (nearly zero for example) or too high TF-IDF values are not necessary to consider. In our experiment, we set the valid range of TF-IDF as $[0.2, 0.5]$.

[3] https://www.idiap.ch/dataset/youtube-personality



(a) sentence count

(b) word count

Fig. 4: Cumulative distributions of sentences and words for YouTube and stream-of-consciousness datasets

The second contrast model we use is 2CNN, 2-dimension convolution networks coming from Honghao Wei et al[9]. Since text data is only part of their whole dataset and the task is only one of the classifiers in their heterogeneous ensembles, we removed LIWC features and reserved document vectors from their CNN. We choose it as one of our contrast models because this model is the latest method which focuses on the relationship between words.

The third model is 3-dimensional CNN, which is proposed by Navonil Majumder et al[10]. They use convolution kernels in 3-dimensions (word vector, phrase, and sentence). Since the computation is huge, we joined sentences into a single one, and then use 2-dimension convolution. All maps from 1,2,3-gram kernels were padded with zeros to keep the same shape. At last, we concatenated the maps again and use max pooling layer to get the document vector. The reason why we choose it is that this model is the latest method which attempts to learn the feature based on sentences.

As for the last two contrast models, they all belong to RNN method. One is single RNN (1LSTM), and the other is bi-directional LSTMs without current word concatenated. They are all typical models in text classification.

C. Experimental Setting

Our experiments are based on ubuntu 16.04 LTS with 8GB memory and Intel Core i7-4790 CPU. The maximal word count in one document is set 800 (pad zero if less than it), and the word vector is 100 dimensions. According to these, each layer's output shape is shown in Table II. Besides, we opened our source code for research purposes[4].

As ordinary operations, we split our dataset as test data and train data (about 9 : 1). In each training epoch, about 20% percent of the train data is treated as validation data. The maximum epoch times set here is 100, but in the practical experiment, most tasks can early stop at 20-30 epochs. In order to overcome the overfitting, we added some dropout layers with drop rate from 0.2 to 0.3, which depends on the situation.

TABLE II: Architecture shape of main layers

Layer	Output Shape
Bidirectional LSTM	(800, 600)
Concatenate with current word	(800, 700)
Conv2D with 1-gram	(25, 300, 10)
Conv2D with 2-gram	(24, 300, 10)
Conv2D with 3-gram	(23, 300, 10)
MaxPooling2D	(3, 300, 10)
MaxPooling2D	(1, 300, 1)
Softmax	(2)

* In order to overcome the overfitting, we added some dropout layers with drop rate from 0.2 to 0.3, which are not listed in the table.

D. Results

The macro precision of these models for both stream-of-consciousness dataset and YouTube dataset is listed in Table III. We use bold font to mark the top two best results in each category. As a whole, we can find that the 2CLSTM model performs pretty well in all categories. Although in some classes such as CON or OPN for stream-of-consciousness dataset and AGR or OPN for YouTube dataset, 2CLSTM doesn't come to the top, it still keeps relatively good precision. In addition, for the stream-of-consciousness dataset, 2CLSTM get better in categories such as AGR and NEU, while for YouTube dataset, it gets better in EXT and NEU.

The macro precision of YouTube is generally superior to that of stream-of-consciousness, which reflects, to some extent, that models listed here (including 2CLSTM) seem to be better at detecting the personality traits in the eyes of other people than the author himself.

[4] Source code is available for research purposes. You can download from this URL: https://github.com/sunxiangguo/2CLSTM