Sentiment Analysis for User Reviews
Prof. S. P. Pingat1 Rajshree S. Dangche2 Sakshi Jain3 Vaishali B. Lambhate4
1,2,3,4Department of Computer Engineering
SKNCOE, Vadgaon (Bk), Pune, Maharashtra, India

Abstract—An important part of our information-gathering behavior has always been to find out what other people think. With the growing availability and popularity of opinion-rich resources such as online review sites and personal blogs, new opportunities and challenges arise as people now can, and do, actively use information technologies to seek out and understand the opinions of others. Sentiment analysis or opinion mining is one of the major tasks of NLP (Natural Language Processing). Sentiment analysis has gained much attention in recent years. In this paper, we aim to tackle the problem of sentiment polarity categorization, which is one of the fundamental problems of sentiment analysis. A general process for sentiment polarity categorization is proposed with detailed process descriptions.

Key words: RNN (Recurrent Neural Network), LSTM (Long Short Term Memory), NLP (Natural Language Processing), Opinion mining, Sentiment polarity categorization

I. INTRODUCTION

In the recent years, the social network has become a very popular and convenient communication environment in our life. On micro blogging platforms, such as Twitter and Facebook, a diverse range of people are attracted to post short sentences, images, and video links to share life issues and opinions. This popularity results in enormous amount of information covering a wide range of topics on the brands, products, politics and social events.

Sentiment analysis on micro blogging has obtained special interest, because it determines the attitude of a user with respect to some topic or product and thus provides convincing information. For example, it may help manufacturing companies to know how people like their product (or service), what would people prefer.

In this paper, we focus on using sentiment analysis on Twitter. Twitter is a popular micro blogging platform which allows people to post messages of up to 140 characters. Because of the short length of tweets, people often post twitter messages (called Tweets) frequently while attending events like product launches, movie premiers, and music concerts or just to express their opinion on a trending or current topic. As such, they can be a valuable source of public opinion or feedback.

Although there exists plenty of work on text classification, some unique characteristics of tweets present special challenges for sentiment analysis: 1) Tweets are short in length. There is a limitation of 140 words for each tweet which makes analysing them challenging; 2) The language used in tweets is very informal with misspellings (often intentional, like different spellings of "cool" : coool, cooll, coooool!!), new words, slangs, and URLs; 3) The number of tweets increases at a very fast pace, and with new data comes new words, new trends in using abbreviations, which lead to a frequent problem of out-of-vocabulary words 4) Special symbols and their combinations are often used, like emoticons and hashtags.

In view of the challenges in this paper, we propose to use bi-directional Long Short-Term Memory (LSTM) networks which operate at character-level input and make predictions at the tweet-level. Such networks naturally handle the problem of very large vocabulary sizes and the presence of sub-word information, without having to keep many trained embedding which would result from keeping very large vocabularies. We find that the model interestingly performs better than equivalent LSTM model which operates at the word level.

II. RELATED WORK

Twitter sentiment analysis is increasingly drawing attention of researchers in recent years. Given the length limitations on tweets, sentiment analysis of tweets is often considered similar to sentence level Sentiment analysis [1]. However, phrase and sentence level approaches can hardly define the sentiment of some specific topics. Considering opinions adhering on different topics, Wang et.al. [2] proposed a hashtag-level sentiment classification method to generate the overall sentiment polarity for a given hashtag. Recently, following the work of some researchers used neural network to implement sentiment classification. For example, Kim adopted convolutional neural networks to learn sentiment-bearing sentence vectors; Mikolov et al. proposed Paragraph vector which outperformed bag-of-words model for sentiment analysis, and Tang et. al. used ConvNets to learn sentiment specific word embedding (SSWE), which encodes sentiment information in the continuous representation of words. Furthermore, Kalchbrenner proposed a Dynamic Convolutinal Neural Network (DCNN) which uses dynamic k-max pooling, a global pooling operation over linear sequences. Instead of directly applying ConvNets to embeddings of words, one of the researcher applies the network only on characters. They showed that the deep ConvNets does not require knowledge of words and thus can work for different languages. LSTM is another state-of-the-art semantic composition models for sentiment classification. Similar to DCNN, it also learns fixed-length vectors for sentences of varying length, captures words order in a sentence and does not depend on external dependency or constituency parse results.

Recurrent Neural Networks (RNNs) [3] are a class of artificial neural networks used for modelling sequences. RNNs are highly flexible in their use of context information as they can learn what part of the input sequence to store to memory and what parts to ignore. They also allow modelling of various regimes of sequence modelling as shown in Fig. 1. Please refer to [3] for a comprehensive review of sequence modelling using RNN.
Fig. 1: RNNs allow modeling of multiple types of input and output sequences.

Fig. 2: The repeating module of LSTM. $x_t$ is the input as time $t$ and $h_t$ is the output from the LSTM output gate at time $t$. The top horizontal line corresponds to the cell state and the bottom line corresponds to the hidden state (both of which are recurring states).

One of the shortcomings of RNN is that it is very difficult to store information over long sequences because of problems due to vanishing and exploding gradients as explained in [4]. Long Short-Term Memory (LSTM) are designed to remedy this and store information over larger input sequences. They achieve this using special “memory cell” units. Fig. 2 shows the architecture of this cell which consists of an input gate, a forget gate, an output gate and a recurring cell state. Refer to [4] for a gentle introduction to LSTM and to [4] for a more comprehensive review and applications.

III. METHODOLOGY

A. Long Short Term Memory (LSTM):

Recurrent Neural Networks (RNNs) are a class of artificial neural networks used for modeling sequences. One of the shortcomings of RNN is that it is very difficult to store information over long sequences because of problems due to vanishing and exploding gradients as explained in [3]. Long Short-Term Memory (LSTM) are designed to remedy this and store information over larger input sequences. Long Short Term Memory networks – usually just called “LSTMs” – are a special kind of RNN, capable of learning long-term dependencies. This is achieved by using special “memory cell” units.

![LSTM Architecture](image)

Fig. 3: The long short-term memory cell

Fig. 3 shows the architecture of this cell which consists of an input gate $i$, a forget gate $f$, an output gate $o$, a recurring cell state $c$ and hidden state output $h$. These values are estimated through the equations Eqs. (1.1), (1.2), (1.3), (1.4), (1.5), where $\sigma$ is the logistic sigmoid function.

- The LSTM’s final output: Output of the LSTM scaled by a tank transformation of the current state:
  $$ h_t = o_t \cdot \tanh(c_t) $$ (1.1)
- Output gate: Scale the output from the cell
  $$ o_t = \tanh(W_{ox}x_t + W_{oh}h_{t-1} + W_{oc}c_t + b_o) $$ (1.2)
- Cell state update step: computes the next time steps state using the gated previous state and the gated input. In other hand, transforms the input and previous state to be taken into account into the current state:
  $$ c_t = f_t \cdot c_{t-1} + i_t \cdot \tanh(W_{cx}x_t + W_{ch}h_{t-1} + b_c) $$ (1.3)
- Forget (reset) gate: Decides whether to erase (set to zero) or keep individual components of the memory:
  $$ f_t = \sigma(W_{fx}x_t + W_{fh}h_{t-1} + W_{fc}c_t + b_f) $$ (1.4)
- Input gate: Controls how much of the current input $x_t$ and the previous output $h_{t-1}$ will enter into the new cell:
  $$ i_t = \sigma(W_{ix}x_t + W_{ih}h_{t-1} + W_{ic}c_{t-1} + b_i) $$ (1.5)

However, this variant LSTM model can only capture context sequence on one input direction. In order to access long-range context in both input directions, bi-directional LSTM is used through forward and backward sequence to two separate recurrent hidden layers.

B. Bi-Directional Long Short Term Memory

A BLSTM consists of two LSTMs that are run in parallel: the first LSTM processes the input sequence from left to right and the second LSTM does the input from right to left. At each time step, the hidden state of the BLSTM is the concatenation of the forward and backward hidden states. This initialization allows the hidden state to capture both past and future information that exploits information follows both directions. The Fig. 4 shows the structure of a Bi-directional LSTM, in which $y_t$ is one of elements in output vector sequence $y$ and computed by equation Eq. (2.1).

$$ y_t = W_y h_t + b_y $$ (2.1)
by the second layer of the LSTM. This LSTM layer then encodes the information of the entire tweet in a r (=256 in experiments) dimensional space which can be used to classify the tweet. The same model can be used with either character input data or word input data. Note that when used with word input, it is common to initialize the models with pre-trained word embeddings [2]. We explore multiple such initializations in our experiments. For the character input model, the embeddings are always initialized randomly, since pre-trained character embeddings are not yet available. We explored other variants of the model like have single-directional LSTM, having a single layer as opposed to multiple layers, using concatenation instead of averaging after first layer. We also tried an ensemble LSTM which takes both characters and words as separate inputs which are combined at the last stage by concatenation before feeding to the softmax. Unfortunately, none of these models gave good preliminary results and in the experiments, we focused on just the model of Fig. 6 with either character or words as inputs.

IV. CONCLUSION AND FUTURE WORK

By using the sentiment analysis, we can predict the attitude, judgment, opinion of a person. In this paper, we have focused on the problem of polarity classification for sentiment analysis of tweets. Here, we have applied the DBLSTM model to solve this problem. Because the reviews are noisy and usually contain new words and new acronyms then in this paper we have proposed a character-based modelling for this DBLSTM model. In future work, we have to focus on the sentiments or judgments containing sarcasms, idioms which are complex to understand by the sentiment analyser. Now a days there are many people using short forms for words like only ‘r’ is used for are. Model for sentiment analysis should be able to extract the sentiment from such tweets.

REFERENCES