Sentiment Analysis of Twitter Users

Yeruva, Vijaya Kumari
Computer Science Department
University of Missouri Kansas City
Kansas City,Missouri
vyq4b@mail.umkc.edu

Junaid, Sidrah
Computer Science Department
University of Missouri Kansas City
Kansas City,Missouri
sjhv6@mail.umkc.edu

Goudarzvand, Saria
Computer Science Department
University of Missouri Kansas City
Kansas City,Missouri
sgnbx@mail.umkc.edu

Abstract— **Sentiment analysis is the computational approach of people's attitude in social networks or other channels. It is one of the most active area in natural language processing. Nowadays it is a hot topic because of two reasons. Firstly, it has many applications because opinions are important to everybody and are key factors in many industries and jobs. Secondly, it has not been studied before 2000 mainly because there were little opinions in web pages. Sentiment analysis systems are going to be applied in any social domain due to the information they can provide the businesses. In this research, I will start with the discussion of the extracting information of social networks regarding positivity negativity happiness sadness and regarding their activity.**

I. INTRODUCTION

Nowadays huge amount of text data is available through social networks related each individual. It can be analyzing a person's behavior by analyzing that text data. Text analysis consists in reading texts formed in natural languages, determining the explicit or implicit meaning of each elements such as words, phrases, sentences and paragraphs, and making inferences about the implicit or explicit properties of these texts. This problem has been traditionally difficult because of the extreme variability in language formation [3]. And it requires prelabeled data to classify the individual into a class. There are two approaches to achieve this. One is traditional machine learning techniques, another is with deep learning techniques.

With the advancement of deep learning and availability of large datasets, methods of handling text understanding using deep learning techniques have gradually become available. One technique which draws great interests is CNN. On the other hand, some researchers have also tried to train a neural network from word level with little structural engineering [3]. In these works, a word level feature extractor such as lookup table or word2vec is used to feed a temporal ConvNet [3].

In this paper, used deep learning techniques to achieve text classification. One of the deep learning technique we used for text classification CNN (Convolution Neural Network) and GloVe.
`

II. RELATED WORK

Some researchers have already worked on text classification and they achieved notable results. One of the work [5] related to text classification with CNN (Convolution Neural Network) model. The dataset used for their work is the Movie Review data from Rotten Tomatoes. The dataset contains 10,662 example review sentences, half positive and half negative. The dataset has a vocabulary of size around 20k. Also, the dataset doesn't come with an official train/test split, so they simply use 10% of the data as a dev set. The model they followed is CNN (Convolution Neural Network)

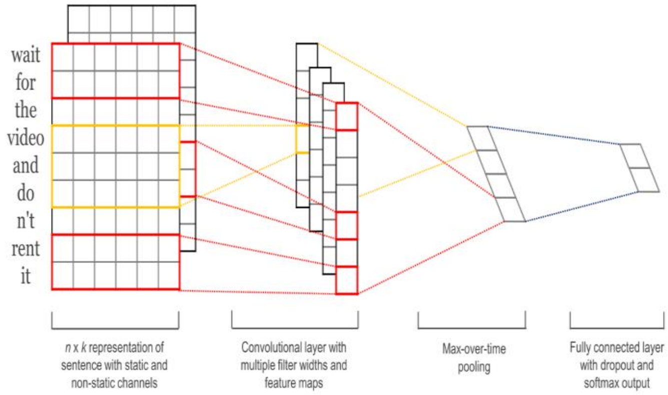

wait
for
the
video
and
do
n't
rent
it
n x k representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output

Figure1: Model architecture with two channels for an example sentence.

The first layers embeds words into low-dimensional vectors. The next layer performs convolutions over the embedded word vectors using multiple filter sizes. For example, sliding over 3, 4 or 5 words at a time. Next, they max-pooled the result of the convolutional layer into a long feature vector, add dropout regularization, and classify the result using a softmax layer. They got 76% of accuracy for text classification with CNN model. In another paper [4], they introduce a recurrent convolutional neural network for text classification without human-designed features like dictionaries, knowledge bases and special tree kernels. In their model, they apply a recurrent structure to capture contextual information as far as possible when learning word representations, which may introduce considerably less noise compared to traditional window-based neural networks. They also employ a max-pooling layer that automatically judges which words play key roles in text classification to capture the key components in texts. They used four commonly used data sets for experiments. The experimental results show that the proposed method out performs the state-of-the-art methods on several datasets, particularly on document-level datasets.

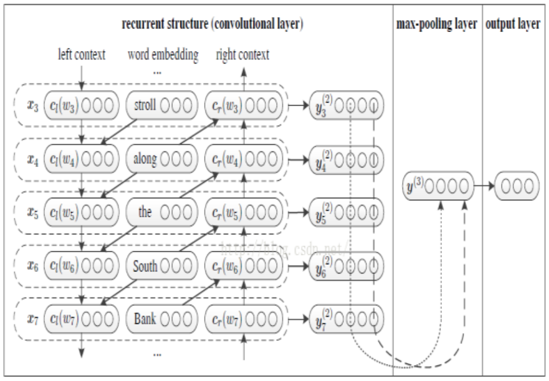
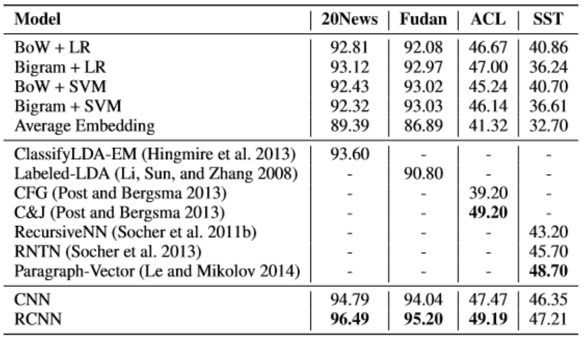

recurrent structure (convolutional layer)
max-pooling layer
output layer
left context
word embedding
right context
...
x_3 $c_l(w_3)$ stroll $c_r(w_3)$ $y_3^{(2)}$
x_4 $c_l(w_4)$ along $c_r(w_4)$ $y_4^{(2)}$
x_5 $c_l(w_5)$ the $c_r(w_5)$ $y_5^{(2)}$
x_6 $c_l(w_6)$ South $c_r(w_6)$ $y_6^{(2)}$
x_7 $c_l(w_7)$ Bank $c_r(w_7)$ $y_7^{(2)}$
$y^{(3)}$
...

Figure2: The structure of the recurrent convolutional neural network

The data sets they used are 20News, Fudan, ACL and SST. 20Newsgroups dataset contains messages from twenty newsgroups. They use the by date version and select four major categories (comp, politics, rec, and religion). Fudanset – The Fudan University document classification set is a Chinese document classification set that consists of 20 classes, including art, education, and energy. ACL Anthology Network dataset contains scientific documents published by the ACL and by related organizations. It is annotated by Post and Bergsma (2013) with the five most common native languages of the authors: English, Japanese, German, Chinese, and French. Stanford Sentiment Treebank dataset contains movie reviews parsed and labeled by Socher et al. (2013). The labels are Very Negative, Negative, Neutral, Positive, and Very Positive. Below table represents the accuracy for their text classification model.



Model	20News	Fudan	ACL	SST
BoW + LR	92.81	92.08	46.67	40.86
Bigram + LR	93.12	92.97	47.00	36.24
BoW + SVM	92.43	93.02	45.24	40.70
Bigram + SVM	92.32	93.03	46.14	36.61
Average Embedding	89.39	86.89	41.32	32.70
ClassifyLDA-EM (Hingmire et al. 2013)	93.60	-	-	-
Labeled-LDA (Li, Sun, and Zhang 2008)	-	90.80	-	-
CFG (Post and Bergsma 2013)	-	-	39.20	-
C&J (Post and Bergsma 2013)	-	-	**49.20**	-
RecursiveNN (Socher et al. 2011b)	-	-	-	43.20
RNTN (Socher et al. 2013)	-	-	-	45.70
Paragraph-Vector (Le and Mikolov 2014)	-	-	-	**48.70**
CNN	94.79	94.04	47.47	46.35
RCNN	**96.49**	**95.20**	**49.19**	47.21

Table1: Test set results for the datasets

III. PROPOSED WORK

Word embeddings are a family of natural language processing techniques aiming at mapping semantic meaning into a geometric space. This is done by associating a numeric vector to every word in a dictionary, such that the distance (e.g. L2 distance or more commonly cosine distance) between any two vectors would capture part of the semantic relationship between the two associated words. The geometric space formed by these vectors is called an embedding space.

For instance, "coconut" and "polar bear" are words that are semantically quite different, so a reasonable embedding space would represent them as vectors that would be very far apart. But "kitchen" and "dinner" are related words, so they should be embedded close to each other.

Ideally, in a good embeddings space, the "path" (a vector) to go from "kitchen" and "dinner" would capture precisely the semantic relationship between these two concepts.

Word embeddings are computed by applying dimensionality reduction techniques to datasets of co-occurence statistics between words in a corpus of text. This can be done via neural networks (the "word2vec" technique), or via matrix factorization.

A. GloVe word embeddings

We will be using GloVe embeddings GloVe stands for "Global Vectors for Word Representation". It's a somewhat popular embedding technique based on factorizing a matrix of word co-occurrence statistics.

Specifically, we will use the 200-dimensional GloVe embeddings of 30k words computed on twitter. The task we will try to solve will be to classify tweets coming from twitter, into their original 2 categories.

B. Nearest neighbors

The Euclidean distance (or cosine similarity) between two word vectors provides an effective method for measuring the linguistic or semantic similarity of the corresponding words. Sometimes, the nearest neighbors according to this metric reveal rare but relevant words that lie outside an average human's vocabulary.

Here's how we will solve the classification problem:

• convert all text samples in the dataset into sequences of word indices. A "word index" would simply be an integer ID for the word. We will only consider the top 8000 most commonly occuring words in the dataset, and we will truncate the sequences to a maximum length of 400 words.

- Avoid combining SI and CGS units, such as current in amperes and magnetic field in oersteds. This often leads to confusion because equations do not balance dimensionally. If you must use mixed units, clearly state the units for each quantity that you use in an equation.
- prepare an "embedding matrix" which will contain at index I the embedding vector for the word of index I n our word index.
- load this embedding matrix into a Keras Embedding layer, set to be frozen (its weights, the embedding vectors, will not be updated during training).
- build on top of it a 1D convolutional neural network, ending in a softmax output over our 2 categories

After preparing dataset we can then build a small 1D

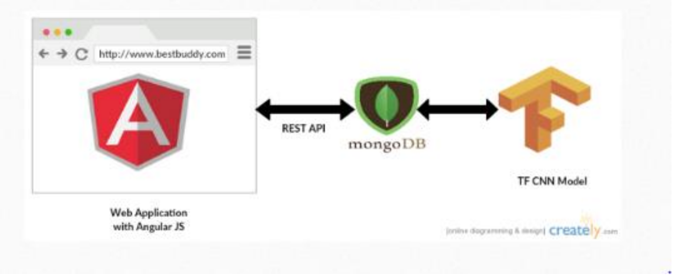
convNet to solve our classification problem

This model reaches 52% on the validation set after only 2 epochs. You could probably get to an even higher accuracy by training longer with some regularization mechanism (such as dropout) or by batch normalization.

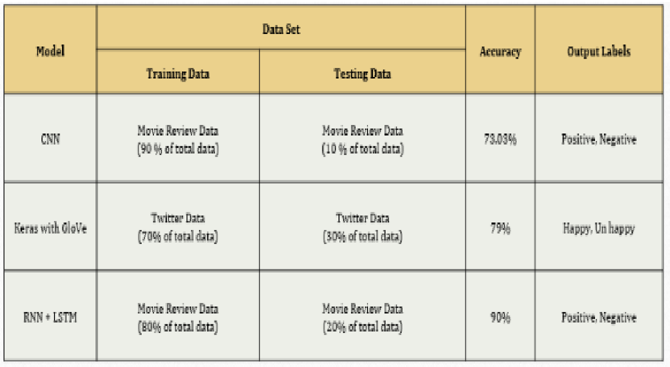
IV.IMPLEMENTATION AND EVALUATION

The Convolutional Nueral Network is implemented to predict the sentiments of user.The model is trained over Movie Review data.The Movie Review data is divided in a way that 90%is used for training and 10% for validation.The trained model generated the labels “HAPPY” and “UNHAPPY”.The accuracy of the model is 73.07%.Tensorflow is used for implementing that Deep Learning Model. In our approach, we used tensor flow for implementing the project which is an open source library for numerical computation using data flow graph. In addition to that Tensorflow came up with distinguishing results in the field of Deep Learning. Flask a light weight web framework, is used to connect Tensorflow model with web.Angular JS is used for creating Web Interface which is used to take user input and MongoDB ,NoSQL database is used to save the user input in oreder to send it to model.

i. System Architecture



ii. Models Comparison



Model	Data Set		Accuracy	Output Labels
	Training Data	Testing Data		
CNN	Movie Review Data (90 % of total data)	Movie Review Data (10 % of total data)	73.03%	Positive, Negative
Keras with GloVe	Twitter Data (70% of total data)	Twitter Data (30% of total data)	79%	Happy, Un happy
RNN + LSTM	Movie Review Data (80% of total data)	Movie Review Data (20% of total data)	90%	Positive, Negative

V. CONCLUSION

The objective of this application is to perform sentiment analysis on Twitter data in order to classify the user as happy or unhappy and on the basis of results he or she can change his/her life style. Moreover, we compare the computational accuracy of different CNN classification model using Tensorflow .In future, the application can be extended in order to perform more classification of data.

REFERENCES

[1] C.Cheng, “Reporting of semi-Supervised-CNN-for text categorization via Region Embedding ” University at Buffalo 2016.

[2] A. Agarwal, B.Xie, I.Vovsha, O.Rombow “Sentiment Analysis of Twitter Data” Columbia University, New York , NY 10027 USA, 43-52 2012

[3] Zhang, Xiang, and Yann LeCun. "Text understanding from scratch." arXiv preprint arXiv:1502.01710 (2015).

[4] Lai, Siwei, et al. "Recurrent Convolutional Neural Networks for Text Classification." *AAAI*. Vol. 333. 2015.

[5] http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow/