

# Computational Intelligence Lab: Tweet Sentiment Classification Group: SNYY (2021)

Department of Computer Science,  
ETH Zurich, Switzerland

Ying Jiao  
yijiao@student.ethz.ch

Shuting Li  
shutli@student.ethz.ch

Yuyan Zhao  
yuyzhao@student.ethz.ch

Nils Ebeling  
ebelingn@student.ethz.ch

## ABSTRACT

As interactions on social media become an ever more prevalent aspect of daily life, performing various forms of analysis on these vast quantities of data is becoming a task of significant interest. In this report, we focus particularly on sentiment classification, i.e. assigning to a piece of text given as input a label that indicates either a positive or negative sentiment. We describe a model ensemble that performs such sentiment classification, trained on over 2 million tweets. Our ensemble incorporates individual models based on BERT and LSTMs augmented with sentiment information. We show that our ensemble outperforms both baselines as well as any individual model in terms of classification accuracy on the public test set.

## 1 INTRODUCTION

As internet access becomes ubiquitous, many conversations move into online spaces. Social media services like twitter have not only become an integral part of interpersonal communication, but also play a significant role in things like political and cultural discourse or marketing campaigns. Twitter’s main functionality is to allow users to publicly post short-form messages, so called tweets, of at most 280 characters.

The goal of this work is to develop a tool that can be used to perform sentiment analysis on tweets. Each tweet should be categorized as expressing either a positive or a negative sentiment. To do this, we employ a machine learning approach based on state of the art models.

Different models excel at capturing different aspects of language. Hence we employ ensemble learning techniques to combine different models. We use two types of models, one type based on BERT, in order to capture global language structure as well as context, and one based on an LSTM which is augmented with sentiment information to hone in specifically on the aspects of language that convey sentiment. The ultimate verdict is obtained through a majority vote among the outputs of the individual models.

We compare the performance of our model both to simple approaches as a baseline, as well as to any of the component models when used individually. Our findings indicate that our ensemble approach provides superior results in terms of classification accuracy on the public test set.

## 2 RELATED WORK

The underlying models we use are based on two architectures: BERT [4] and LSTMs [6].

### 2.1 BERT

BERT functions as a kind of autoencoder. The model architecture employed by BERT is that of the attention based transformer described in [2]. The underlying idea behind BERT is to mask a small subset of the input tokens, and then train the model to predict those masked tokens. This way, BERT finds a complex representation of the input that can then be used in future downstream tasks such as next token prediction or sentiment classification. The strength of BERT when applied to our task is that it captures profound aspects of the underlying structure of language.

### 2.2 LSTMs

We make use of the recurrent neural networks described in [6]. These are RNNs using long short term memory cells, but with the addition of bidirectionality. This allows the model to capture both left and right contexts. These architectures can be implemented with information from a sentiment lexicon to hone in on the sentiment information contained in the input.

## 3 METHOD

We utilize baseline ML models, LSTM models and BERT models in this sentiment classification task. We find that adding the attention mechanism to the LSTM yields no significant improvement. This leads to our hypothesis that the attention layer may not be able to recognize the sentiment information in tweets effectively. Motivated by [9], we implement a sentiment augmented LSTM-attention model (LSTM-SAT) which integrates sentiment information from sentiment lexicons. In this section, we introduce the applied models and our motivations for using them.

### 3.1 Baseline ML Models

The baseline is set by training classic machine learning classifiers on vectorized tweets. First, we apply pre-trained GloVe word embeddings to obtain a vector for each word in the tweet. In addition, each tweet is represented using a vector by averaging the word vectors for each word. Lastly, two well-established classifiers, i.e.,

random forest classifier and support vector machine, are used on the resulting embedded tweet.

### 3.2 LSTM Models

LSTM models are widely used in text processing tasks. Benefiting from the gating mechanism, they are able to maintain long term information [8].

1) **Stacked bidirectional LSTM (Stacked BLSTM)**: Inspired by [6], we use bidirectional LSTMs (BLSTM) to capture all input information from both the left and right contexts. We further deepen this architecture by vertically stacking them, with the outputs of a BLSTM becoming the input sequence of the next one. The stacked representation has been reported to outperform shallow ones in many tasks such as machine-translation [7]. In our implementation, we use two-stacked BLSTMs with dimensionality 32 and 16 respectively. Then the results are passed to a fully-connected layer of 32 hidden states, followed by a dropout layer with probability 0.5 and a final fully-connected layer.

2) **LSTM with attention mechanism (LSTM-AT)**: The attention mechanism is integrated to emphasize the semantically important words and sentences [10]. We connect our LSTM model to an attention layer which modifies the weights of different words.

3) **LSTM with sentiment augmented attention (LSTM-SAT)**: Attention models based only on local context may not be able to identify all meaningful information in different tasks. The similarity in performance of the LSTM and the LSTM-AT on this dataset supports this hypothesis. Inspired by [9], we integrate external knowledge, obtained from sentiment lexicons, to improve our attention model.

Sentiment lexicons are dictionaries of opinion words labeled with sentiment scores [3], which are frequently used in lexicon-based sentiment classification approaches. We use the public opinion words in tweets provided by Bing Liu<sup>1</sup>. Based on the assumption that the sentiment score of positive / negative words is related to their frequency in positive / negative tweets, we define the sentiment score  $S(t)$  of the opinion word  $t$  as

$$S(t) = \frac{F_i(t)}{\max_{t' \in i} F_i(t')}$$

where  $i = 1$  for positive opinion words and  $i = -1$  for negative ones.  $F_i(t)$  denotes the frequency of positive / negative  $t$  in positive / negative tweets.

The architecture of the LSTM-SAT is shown in Figure 1. For each tweet  $d_i$ , we generate a sentiment vector  $\vec{SV}_{d_i}$ . The corresponding entries of opinion words in  $\vec{SV}_{d_i}$  are their sentiment scores, while all other entries are set to zero. The embedded tweet  $d_i$  is first fed into the LSTM layer. The output word representations and  $\vec{SV}_{d_i}$  are used as inputs of the SAT layer to produce the document representation  $\vec{R}_{d_i}$ . Finally, a dense layer is connected to generate the prediction.

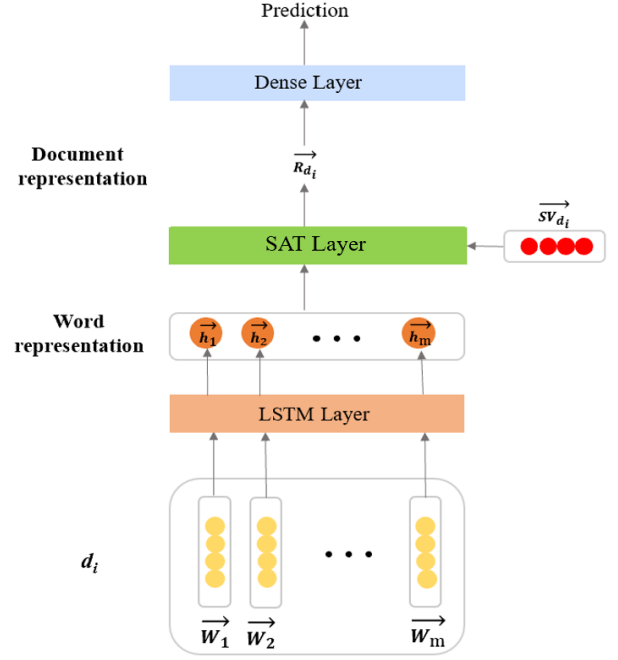


Figure 1: LSTM-SAT Architecture

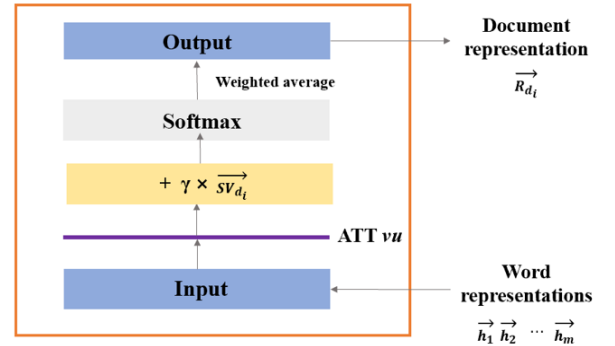


Figure 2: SAT Layer

Figure 2 illustrates the detailed structure of the SAT layer. The adjusted attention weights  $\vec{vu}'$  for document representation generation are defined as:

$$\vec{vu}' = \text{softmax}(\vec{vu} + \gamma * \vec{SV}_{d_i})$$

where  $\gamma$  is a parameter for scale matching between the original attention weights  $\vec{vu}$  and the sentiment scores.

### 3.3 BERT models

BERT is a pre-trained bidirectional transformer model proposed in [4]. Being trained with masked language modeling and next sentence prediction, it is efficient in improving many NLP tasks. We load the pre-trained BERT model and fine-tune its parameters using training data consisting of labeled tweets.

<sup>1</sup><https://www.kaggle.com/nltkdata/opinion-lexicon>

1) **BERT-NN**: The final [CLS] token embedding for every tweet is passed on to a neural network classifier, which has a 768-dimensional linear layer with ReLU activation followed by a second linear layer of 50 dimensions.

2) **BERT for sequence classification (BERT-Seq)**: Instead of using the neural network described in BERT-NN, this classifier has a linear layer before pooling the output.

### 3.4 Others

1) **Data preprocessing**: We apply regular expressions to remove URLs, non-ASCII characters, redundant spaces and replace special characters like "&". For LSTM and BERT models, the length of the tweets is normalized to 128.

2) **Word embedding**: The Word2Vec embedding method [5] is used with an embedding dimension of 200, minimal frequency of 5 and window size of 5.

3) **Majority vote**: To lower the prediction variance, we take the majority vote of the best 7 test prediction results of the LSTM-SAT and BERT models as our final prediction.

## 4 IMPLEMENTATION

The GloVe word embedding is implemented using Stanford's pre-trained vectors specifically for Twitter data<sup>2</sup>. Both machine learning classifiers are implemented using scikit-learn<sup>3</sup>. The Word2Vec embedding is implemented using the Gensim package<sup>4</sup>. All LSTM models are implemented using Tensorflow [1]. The stacked BLSTM model was inspired by the tutorial from the Tensorflow website<sup>5</sup>. The BERT models are pretrained ones from Huggingface<sup>6</sup>. The fine-tuning of BERT models are inspired by the public tutorials<sup>7,8</sup>. Our code is available at the Github repository<sup>9</sup> and is submitted along with this report.

## 5 EXPERIMENT AND RESULTS

We shuffle the data and split it into a training and a validation set.

We use both full data containing 2.5 million tweets and partial data containing 100,000 tweets in order to compare performance for machine learning baseline models. The training set and validation set are split in a ratio of 9 to 1.

1) **GloVe + Random Forest (RF)**: GloVe embeddings of 50, 100, and 200 dimensions are employed in experiments on both full and partial data. The random classifier has 100 trees. Table 1 demonstrates the highest validation score of each combination of settings.

Table 2 illustrates the computational costs of running the models on partial and full data using 50d GloVe embeddings.

**Table 1: Comparison of Accuracy on Different Settings**

Acc. %	GloVe Dimensions		
Data Size	50d	100d	200d
partial	76.05	75.83	76.3
full	76.75	-	-

**Table 2: Comparison of Running Time on Different Data Size**

Data Size	Running Time
partial	1085 sec
full	18078 sec

2) **GloVe + Support Vector Machine (SVM)**: This experiment implements an SVM with two different kernel functions. We run the SVM using a linear kernel on the full data due to its high speed nature. Meanwhile, we also run an SVM using RBF kernels on only the partial data. This is due to the fact that the maximum training time of 24h was exceeded on the full data without the model achieving convergence. The GloVe embedding has a dimension of 200.

Table 3 shows the overall performance of both kernels, where the evaluation accuracy is represented by the highest validation score in training, and the test accuracy is the corresponding public score reported by Kaggle.

**Table 3: Comparison of Performance on Different Kernels**

Kernel	Data Size	Evaluation Acc. %	Test Acc. %	Runtime
Linear	full	78.51	77.18	12480 sec
RBF	partial	80.1	79.54	39214 sec

The LSTM and BERT models on the other hand are run on the full data. The training set contains 2,400,000 tweets while the validation set contains the remaining 100,000 tweets.

The stacked BLSTM model is trained with a batch size of 128 and a learning rate of  $5e^{-4}$  for 2 epochs. The other LSTM models use a batch size of 128 and a learning rate of 0.01. For the LSTM-SAT, the hyper-parameter  $\gamma$  is set to 0.8. The best evaluation results appear around the 15th training epoch. The training batch size for both BERT models is 32 with a learning rate of  $5e^{-5}$ . The models are trained for 2 epochs.

Our models are trained on the Leonhard cluster. The maximum training time is 24h. To ensure our experiment results are reproducible, all random seeds are fixed.

The experiment results are shown in Table 4. The evaluation accuracy is the best accuracy achieved during training. The testing accuracy is the corresponding public score reported by Kaggle.

## 6 DISCUSSION

It is worth noting from Table 1 and 2 that for random forest classifiers, different combinations of experimental settings may result in similar performance in terms of accuracy, but create a tenfold difference in running time.

<sup>2</sup><https://nlp.stanford.edu/projects/glove/>

<sup>3</sup><https://scikit-learn.org/stable/>

<sup>4</sup><https://github.com/RaRe-Technologies/gensim>

<sup>5</sup>[https://www.tensorflow.org/text/tutorials/text\\_classification\\_rnn](https://www.tensorflow.org/text/tutorials/text_classification_rnn)

<sup>6</sup>[https://huggingface.co/transformers/model\\_doc/bert.html](https://huggingface.co/transformers/model_doc/bert.html)

<sup>7</sup><https://skimai.com/fine-tuning-bert-for-sentiment-analysis/>,

<sup>8</sup><https://mccormickml.com/2019/07/22/BERT-fine-tuning/>

<sup>9</sup>[https://github.com/NE1997/CIL\\_Project](https://github.com/NE1997/CIL_Project)

Table 4: Experiment Results on Different Models

Model	Evaluation Acc. %	Test (Public) Acc. %
GloVe+RF (partial)	76.3	75.82
GloVe+SVM (partial)	80.1	79.54
Stacked BLSTM	85.49	85.64
LSTM-AT	86.23	85.66
LSTM-SAT	87.03	86.22
BERT-NN	90.11	89.20
BERT-Seq	90.04	89.42
Major Vote		89.58

There are two interesting observations we can derive from Table 2. On the one hand, the linear kernel function is 3 times faster than the RBF kernel function, despite the former being run on the full data which is 25 times of the size of the partial data. On the other, the SVM using RBF kernels outperforms the SVM using linear kernels despite the data size difference. This could indicate that the sentiments contained in the tweets might not be easily linearly separated.

We also observe from Table 4 that LSTM models outperform all baseline ML models. Among these, the LSTM-AT doesn't show significant performance improvements compared to the stacked BLSTM without attention. After integrating the sentiment information from a sentiment lexicon, the LSTM-SAT better highlights the opinion words and outperforms the other two LSTM models.



**Figure 3: An example tweet with attention weights**

Figure 3 demonstrates an example tweet and the attention weights from the LSTM-AT (upper line) and LSTM-SAT (lower line). A deeper color represents larger attention weights. We can see that the LSTM-AT tends to put more attention on the number "6". After adjusting the attention weights by adding the external sentiment knowledge, the LSTM-SAT is able to highlight the negative opinion words "sucks" and "late".

We can also see that the BERT models further improve on the results of the LSTM models. One reason for this could be that BERT models benefit from more meaningful embeddings based on large amounts of pretraining data.

## 7 CONCLUSION

Our results present an improvement over the performance of any individual model. These findings show that training an ensemble of models that capture different aspects of the training data in different ways can help improve performance beyond what any model can provide individually.

## REFERENCES

- [1] Martin Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} symposium on operating systems design and implementation ({OSDI} 16)*. 265–283.
- [2] Niki Parmar Jakob Uszkoreit Llion Jones Aidan N. Gomez Łukasz Kaiser Ilia Polosukhin Ashish Vaswani, Noam Shazeer. 2017. Attention is all you need. *arXiv:1706.03762* (2017).
- [3] Huimin Chen, Maosong Sun, Cunchao Tu, Yankai Lin, and Zhiyuan Liu. 2016. Neural sentiment classification with user and product attention. In *Proceedings of the 2016 conference on empirical methods in natural language processing*. 1650–1659.
- [4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805v2* (2019).
- [5] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).
- [6] Mike Schuster, Kuldip K. Paliwal, and A. General. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* (1997).
- [7] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to Sequence Learning with Neural Networks. *arXiv:1409.3215* (2014).
- [8] Duyu Tang, Bing Qin, and Ting Liu. 2015. Document modeling with gated recurrent neural network for sentiment classification. In *Proceedings of the 2015 conference on empirical methods in natural language processing*. 1422–1432.
- [9] Rong Xiang, Ying Jiao, and Qin Lu. 2019. Sentiment augmented attention network for cantonese restaurant review analysis. In *ACM*. 9.
- [10] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. 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*. 1480–1489.



Eidgenössische Technische Hochschule Zürich  
Swiss Federal Institute of Technology Zurich

## Declaration of originality

The signed declaration of originality is a component of every semester paper, Bachelor's thesis, Master's thesis and any other degree paper undertaken during the course of studies, including the respective electronic versions.

Lecturers may also require a declaration of originality for other written papers compiled for their courses.

I hereby confirm that I am the sole author of the written work here enclosed and that I have compiled it in my own words. Parts excepted are corrections of form and content by the supervisor.

**Title of work** (in block letters):

**Authored by** (in block letters):

*For papers written by groups the names of all authors are required.*

**Name(s):**

**First name(s):**

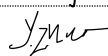
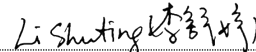
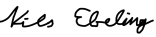

With my signature I confirm that

- I have committed none of the forms of plagiarism described in the '[Citation etiquette](#)' information sheet.
- I have documented all methods, data and processes truthfully.
- I have not manipulated any data.
- I have mentioned all persons who were significant facilitators of the work.

I am aware that the work may be screened electronically for plagiarism.

**Place, date**

**Signature(s)**

	Ying Jiao
	
	Li Shuting 李淑婷
	
	Kels Ebeling
	

*For papers written by groups the names of all authors are required. Their signatures collectively guarantee the entire content of the written paper.*