# Comparison of Sentiment Recognition Techniques

Petros Mitseas[1]

[1]National Centre for Scientific Research "Demokritos"

June 16, 2022

## Abstract

Since the introduction of Deep Learning and Transformers, there has been a major leap in Natural Language Processing. In this paper we revisit different text classification techniques from classic NLP and Deep Learning, up to the latest transformer models. Specifically we examine the task of sentiment recognition by evaluating the models on the Stanford Sentiment Treebank and comparing their performance.

## 1 Introduction

Sentiment Recognition is an important language task with many real-life applications. Essentially i can be modeled as a text classification problem, where the input is text (words, phrases, sentences, paragraphs) and the output is a label (binary or multi-level) indicating the respective sentiment. Classical approaches use bag of words modeling to extract features from text, which are propagated to a machine learning classifier [1]. Though with the recent progress in Deep Learning [2] and Transformers [3] it is now possible to extract sentiment from sophisticated language models, trained on large corpora, with the aid of Transfer Learning. These models use contextual representations of words, by also taking into account the surrounding words, demonstrating human-level performance on a variety of language tasks. There are plenty of available datasets for sentiment analysis, namely the Stanford Sentiment Treebank (SST-2 and SST-5) [4], the IMDb movie reviews dataset [5] and the Amazon product reviews dataset. In the following sections, we revisit and compare some of the most used techniques for sentiment extraction, also evaluating their performance on SST-5.

## 2 Methodology

We discuss and evaluate three types of models, for sentiment classification:

- Support Vector Machines
- Bidirectional Long Short Term Memory Networks [6]
- DistilBERT, a lightweight variant of the Bidirectional Encoder Representations from Transformers (BERT) model

### 2.1 Dataset

The models were trained and evaluated on the Stanford Sentiment Treebank 5 dataset (SST-5). The dataset includes 215,154 phrases and 11,855 sentences, extracted from movie reviews. Each sentence is labeled from 0 (very negative) to 4 (very positive). The phrases are labeled with a continuous number from 0.0 (very negative) to 1.0 (very positive), and can be converted to the respective integer values by thresholding. In our experiments we measured the models' performance on both phrases and sentences.

### 2.2 Model Description

#### 2.2.1 Support Vector Machines

Support Vector Machines [7] have been one of the most popular machine learning models, able to manage small to medium size datasets and

a high number of dimensions. It is often used in conjunction with bag of words representations and is often referred to as a baseline model for simple classification. It relies on creating hyperplanes that split the data on higher dimensions, using an efficient method known as the kernel trick.

#### 2.2.2 Long-Short Term Memory Networks

LSTMs [8] are a variant of Recurring Neural Networks. These types of networks can process sequential data such as text, for tasks like classification and language modeling. In detail, the output of the model at each time step is used alongside the next input. Thus the model maintains a "context" determined by the sequence so far. One other benefit of this architecture is that it can handle sequences of variable length, as the model "unfolds" as many times as the sequence's items. The difference between vanilla RNNs and LSTMs is that the latter introduces internal mechanisms like gates and cell states, to overcome the vanishing gradient and instability that RNNs introduce. Until the rise of attention-based models like Transformers, LSTMs and Convolutional Neural Networks had been the de facto approach for managing sequential data like text.

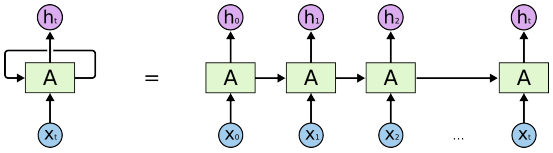


Figure 1: The unrolled RNN.

#### 2.2.3 Bidirectional Encoder Representations from Transformers (BERT)

In 2017, researchers from Google proposed the Transformer architecture that relied solely on the attention mechanism to tackle a number of language tasks [9]. The model was able to surpass the state of the art models of the time while discarding the issues of recurrent based architectures. In 2018 Devlin et al. proposed BERT, an attention-based Transformer architecture, to train deep bidirectional representations from unlabeled texts [10]. In its base form, BERT features 12 layers of transformers blocks, each with a hidden size of 768 and 12 self-attention heads. This corresponds to 110M trainable parameters. The original BERT was trained on a large english dataset (BookCorpus and English Wikipedia) on two tasks: Masked Language Modeling and Next Sentence Prediction. It achieved an average performance of 79.6%, a SOTA score at the time. The strength of BERT lies in its ability to adapt to different tasks with little effort, by switching the last layer ("head" of the transformer) with a randomly initialized one, and training only that layer with the given dataset. This is called fine-tuning and is one of the most successful examples of transfer learning in machine learning. In our experiments on Sentiment Classification we used a lightweight variant of BERT (40% smaller), called DistilBERT [11].

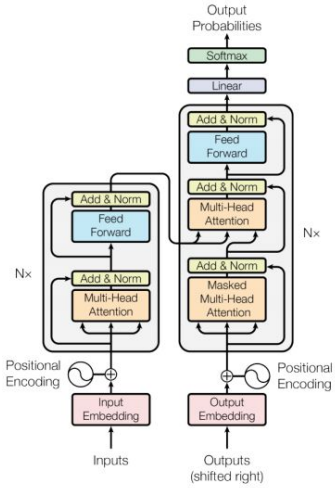


Figure 2: The architecture of a transformer.

### 2.3 Model Inputs

Bag of words models, such as TF-IDF matrices, lead to very sparse feature representations, especially on bigger datasets. These sparse matrices can be very inefficient to calculate and store, in terms of computation time and memory. For this reason, it is common to use word vectors, where each word of the sentence is mapped into a fixed size vector. These representations are created by training a neural network on a large corpora and extracting the hidden network's values. The most common pretrained word vectors include GloVe [12] and Gensim word2vec [13]

In our experiments, the SVM and Bi-LSTM models use as input the pretrained word embed-

| Model | Time (hh:mm:ss) | Accuracy (phrases) | F1 (phrases) | Accuracy (sentences) | F1 (sentences) |
|---|---|---|---|---|---|
| SVM | 00:00:24 | 58.9 | 52.9 | 31.0 | 25.2 |
| Bi-LSTM | 00:07:30 | 67.3 | 65.7 | 58.5 | 57.0 |
| DistilBERT | 01:49:00 | 70.5 | 70.2 | 69.5 | 69.1 |

Table 1: The performance of the three models on the SST-5 dataset

dings provided by the SpaCy library [14]. Since each word of a sentence is represented by a vector, we need a way to aggregate these vectors in order to get a representation of the whole sentence, before feeding it to the SVM classifier. A trivial method that we followed is to take the average of the vectors. This is only required for the SVM model, since LSTMs naturally handle sequential data. In the case of BERT, the input words are converted into subwords that match a predefined vocabulary size of about 30,000 tokens.

### 2.4 Experimental Results

The models were evaluated on the SST-5 and implemented using Pytorch [15] and Huggingface [16]. The training was carried out on phrases and the evaluation both on phrases and sentences. The aggregated results are presented on table 1. It is shown that the DistilBERT model demonstrated the highest scores of all models, while the simple SVM had the worst performance. This is expected as the latter does not take into account the order and the context of words. For example the sentences:

- Good product, not bad at all
- Bad product, not good at all

, are mapped to the same input vector in the SVM classifier, since the average of the words' embedding vectors is the same. Regarding the training time, the SVM becomes inefficient after a few thousand training samples, while not improving much on the prediction capability (table 1).

The Bi-LSTM's training took 3 epochs / 450 seconds to complete, on a 80/10/10 train/validation/test split of the phrase dataset. The model is able to understand the order of words. For example the two sentences above (where the SVM failed) were labeled "positive"

| Samples | Accuracy | F1 | Time (sec) |
|---|---|---|---|
| 237 | 0.50 | 0.35 | 0.02 |
| 1186 | 0.55 | 0.47 | 0.3 |
| 2372 | 0.55 | 0.46 | 1.1 |
| 4745 | 0.56 | 0.50 | 4.2 |
| 11862 | 0.57 | 0.51 | 24.9 |

Table 2: SVM performance vs sample size

and "negative" respectively. It can also handle negations in short sentences such as: "I don't think that the movie was good", which was labeled correctly as "negative". However, the model starts to lose context on bigger sentences:

- "I think that the movie was worth watching": labeled positive
- "I think that the movie, which my father told me about, was worth watching": labeled neutral

Finally it is shown that DistilBERT greatly outperforms the other models, at the cost of training speed. The resulting predictions are remarkable. For example the following sentences are labeled correctly, demonstrating the transformer's ability to maintain context using the attention mechanism:

- "I think that the movie, which my father told me about last night when we returned home, was amazing": very positive
- "I don't think that the movie, which my father told me about last night when we returned home, was amazing": negative

The transformer's superior performance can be illustrated by examining how the detected sentiment changes, as we modify the following story:

- "The chicken crossed the road": neutral

- "The chicken crossed the road, it got struck by luck": positive
- "The chicken crossed the road, it got struck by a car": negative
- "The chicken crossed the road, it got struck by a car, but it survived": neutral
- "The chicken crossed the road, it got struck by a car, but it survived and now it's rich": positive

| - | very neg | neg | neu | pos | very pos |
|---|---|---|---|---|---|
| very neg | 58 | 212 | 9 | 0 | 0 |
| neg | 26 | 467 | 129 | 10 | 1 |
| neu | 1 | 122 | 189 | 75 | 2 |
| pos | 0 | 10 | 75 | 379 | 46 |
| very pos | 0 | 0 | 12 | 187 | 200 |

Table 3: Confusion matrix for BiLSTM model, on sentences

| - | very neg | neg | neu | pos | very pos |
|---|---|---|---|---|---|
| very neg | 169 | 108 | 2 | 0 | 0 |
| neg | 81 | 480 | 59 | 12 | 1 |
| neu | 6 | 105 | 190 | 85 | 3 |
| pos | 0 | 6 | 25 | 391 | 88 |
| very pos | 0 | 0 | 4 | 88 | 307 |

Table 4: Confusion matrix for DistilBERT, on sentences

# 3 Conclusion

Sentiment Recognition is a fundamental test for language models. We have shown that advancements in deep learning such as BERT, enable developing models that demonstrate human-like performance, utilizing the power of transfer learning.

# References


[1] Tony Mullen and Nigel Collier. Sentiment analysis using support vector machines with diverse information sources. In *In Proceedings of Conference on Empirical Methods in Natural Language Processing*, 2004.

[2] Nishit Shrestha and Fatma Nasoz. Deep learning sentiment analysis of amazon.com reviews and ratings. *International Journal on Soft Computing, Artificial Intelligence and Applications*, 8(1):01–15, feb 2019.

[3] Manish Munikar, Sushil Shakya, and Aakash Shrestha. Fine-grained sentiment classification using bert, 2019.

[4] Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA, October 2013. Association for Computational Linguistics.

[5] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[6] Zhiyong Cui, Ruimin Ke, Ziyuan Pu, and Yinhai Wang. Deep bidirectional and unidirectional lstm recurrent neural network for network-wide traffic speed prediction, 2018.

[7] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[8] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[9] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2017.

[10] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2018.

[11] Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter, 2019.

[12] Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543, 2014.

[13] Radim Rehurek and Petr Sojka. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2), 2011.

[14] Matthew Honnibal and Ines Montani. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear, 2017.

[15] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[16] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. Huggingface's transformers: State-of-the-art natural language processing, 2019.