Self-Supervised Representation Learning

Kedarnath P
MTech Computer Science & Automation
(SR NO: 20902)
kedarnathp@iisc.ac.in

Abstract

This project compares the performance of self-supervised representation learning (SSRL) models, specifically fine-tuned BERT, with traditional supervised learning models such as Naive Bayes, Linear SVC, and LSTMs on the sentiment classification task using the IMDb dataset. We also evaluate the performance of different pre-trained Word2Vec models, including custom-trained CBOW and Skip-gram models, Gensim library's CBOW and Skip-gram models, and Google's pre-trained Skip-gram model. The dataset is divided into different training splits (10%, 40%, and 80%) to evaluate the models' performance under varying amounts of labeled data. BERT consistently achieves the highest test accuracy and lowest test loss across all splits, while pre-trained Word2Vec models perform reasonably well in conjunction with a linear SVM classifier.

1 Introduction

This project explores the performance of self-supervised representation learning (SSRL) using fine-tuned BERT models and pre-trained Word2Vec models for sentiment analysis using the IMDb dataset. The primary goal is to understand the benefits and limitations of SSRL, like BERT and pre-trained Word2Vec models, and traditional supervised learning approaches in different scenarios. By comparing these approaches, the study provides valuable insights into their effectiveness, guiding the development of efficient learning algorithms and the selection of appropriate methodologies for different tasks and data availability scenarios.

2 Project Approach

The project compares SSRL models such as BERT, pre-trained Word2Vec models, and supervised learning models like LSTM, Naive Bayes, and Linear SVC on the sentiment classification task using the IMDb dataset. The dataset is divided into different training splits (10%, 40%, and 80%) to evaluate the models' performance under varying amounts of labeled data. The remaining data is split between validation and test sets.

2.1 Bi-directional Encoder Representations from Transformers (BERT)

We fine-tune the pre-trained BERT model for sentiment classification on the IMDb dataset, comparing its performance to traditional supervised learning models. BERT, developed by Google AI, is a powerful pre-trained language model that captures deep language understanding.

2.2 Pre-trained Word2Vec Models

We train and evaluate Word2Vec models on the IMDb dataset using linear SVM as the classifier. Models are evaluated using 80%, 40%, and 10% of training data. We preprocess the data, train

Word2Vec models, and train and evaluate SVM classifiers. We compare custom-trained CBOW and Skip-gram models, Gensim library's CBOW and Skip-gram models, and Google's pre-trained Skip-gram model.

3 Literature survey Overview

3.1 BERT

The paper "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" by Devlin et al. (2019) introduces BERT (Bidirectional Encoder Representations from Transformers), a new pre-trained language model that has significantly advanced the state-of-the-art performance across various natural language processing (NLP) tasks.

BERT's architecture is based on the Transformer model, which was proposed by Vaswani et al. (2017) for machine translation. The key feature of BERT is its bidirectional nature, allowing the model to learn contextual representations by capturing information from both the left and right context of each token in a sentence. This bidirectional context learning is achieved using a technique called Masked Language Modeling (MLM), where a certain percentage of input tokens are randomly masked, and the model is trained to predict the masked tokens based on the context provided by the remaining unmasked tokens.

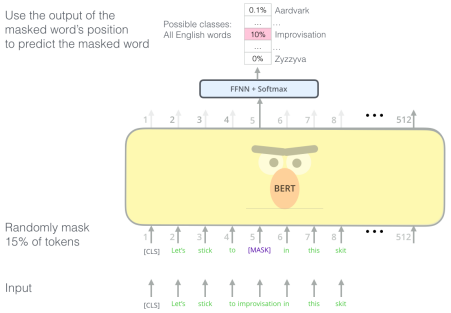

Figure 1: Masked Language Modelling(MLM)

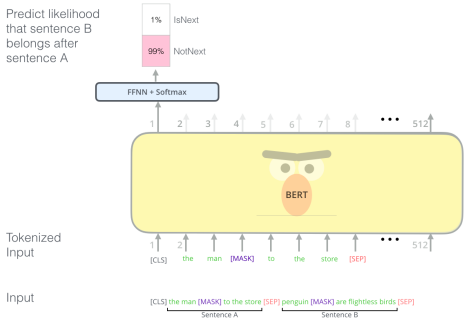

Figure 2: Next Sentence Prediction(NSP)

In addition to MLM, BERT is also pre-trained on a task called Next Sentence Prediction (NSP). In this task, the model is trained to predict whether a given pair of sentences are consecutive or not. NSP helps the model understand the relationship between sentences and improve its understanding of sentence-level context.

BERT is pre-trained on a large corpus of text data, specifically the BooksCorpus (800 million words) and English Wikipedia (2,500 million words). The pre-training process allows BERT to learn the structure of the language and understand context, which is then fine-tuned for specific NLP tasks using smaller labeled datasets. This transfer learning approach reduces the need for extensive labeled data and computational resources when adapting the model to various tasks.

The paper demonstrates that BERT significantly outperforms existing methods on a range of NLP benchmarks, including the General Language Understanding Evaluation (GLUE) benchmark, the Stanford Question Answering Dataset (SQuAD v1.1), and the CoNLL-2003 Named Entity Recognition (NER) task. BERT's performance can be attributed to its bidirectional nature, the use of the Transformer architecture, and its extensive pre-training on large-scale unsupervised text data

3.2 Word2Vec

Word2Vec is a popular technique for training word embeddings, which are dense vector representations of words. It was introduced by Mikolov et al. in 2013 and has since been widely adopted in various natural language processing tasks. There are two main architectures for Word2Vec: Continuous Bag-of-Words (CBOW) and Skip-gram.

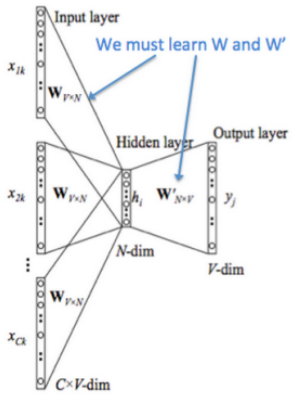

Figure 3: CBOW: Predicting a center word form the surrounding context

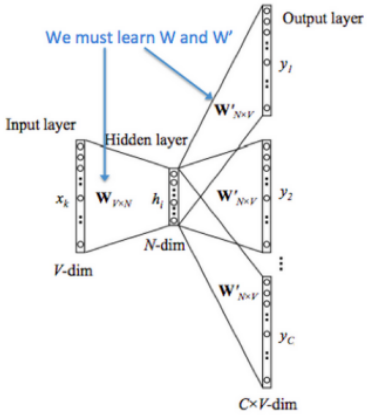

Figure 4: Skip-Gram: Predicting surrounding context words given a center word

CBOW: This architecture predicts the target word based on the context words within a given window. It is faster to train and generally performs better on frequent words.

Skip-gram: This architecture predicts the context words given a target word. It is slower to train but typically performs better on infrequent words and larger datasets.

4 Methodology

4.1 Pre-MidTerm Work

An overview of the different approaches for sentiment classification on the IMDB movie review dataset :

1. **Fine-tuned BERT:**
   The Implemented code fine-tunes a BERT model on the IMDb dataset to perform sentiment analysis. Let's break down the code into the main components.
   - *Imports:* Import the required libraries for the project.
   - *Load and preprocess the data:* Load the IMDb dataset, encode the sentiment labels, preprocess the reviews by removing HTML tags, non-ASCII characters, emojis, mentions, URLs, and punctuation marks. Then convert the text to lower case.
   - *Tokenize the text:* Tokenize the preprocessed text using BERT tokenizer, which returns input IDs and attention masks for each review.
   - *Create PyTorch dataset and dataloaders:* Define a custom TextDataset class, create train, validation, and test datasets and dataloaders using the preprocessed and tokenized data.
   - *Initialize the BERT model:* Load the pre-trained BERT model for sequence classification with two output labels (positive and negative sentiment). The BERT architecture for sequence classification includes the following layers:
     - BertEmbeddings: Combines token embeddings, position embeddings, and segment embeddings.
     - BertEncoder: Consists of multiple transformer layers to process the input sequence.
     - BertPooler: Applies a linear transformation followed by a Tanh activation to the first token's hidden state (the [CLS] token).
     - Dropout: Applies dropout regularization to the pooled output.
     - Linear: Fully connected layer to produce logits for the two output labels (positive and negative sentiment).
   - *Set up training parameters:* Define the number of training epochs, optimizer, and learning rate scheduler.

- *Fine-tune the model:* Train the model on the IMDb dataset for the specified number of epochs. The training loop includes zeroing gradients, calculating loss, backpropagating, clipping gradients to avoid gradient explosion, updating model parameters, and updating the learning rate scheduler. After each epoch, evaluate the model on the validation dataset.
- *Visualize training and validation loss/accuracy:* Plot the training and validation loss and accuracy for each epoch.
- *Evaluate on the test dataset:* Use the fine-tuned model to make predictions on the test dataset and calculate the test loss and accuracy. Print the classification report and plot the confusion matrix.

2. **Naive Bayes:** Naive Bayes is a probabilistic machine learning model based on Bayes' theorem. It assumes that the features are conditionally independent, given the class label. In this approach, the IMDB dataset is preprocessed using TfidfVectorizer to create a Term Frequency-Inverse Document Frequency (TF-IDF) representation of the reviews. Then, the MultinomialNB class from the sklearn library is used to create a Naive Bayes model for sentiment classification. The model is trained on the training set and evaluated on the test set, with accuracy as the performance metric.
3. **Linear Support Vector Machine (SVC):** Support Vector Machines (SVM) are supervised learning models that can be used for classification tasks. Linear SVC is an SVM model with a linear kernel. In this approach, the IMDB dataset is preprocessed using TfidfVectorizer to create a Term Frequency-Inverse Document Frequency (TF-IDF) representation of the reviews. Then, the LinearSVC class from the sklearn library is used to create a Linear SVC model for sentiment classification. The model is trained on the training set and evaluated on the test set, with accuracy as the performance metric.
4. **LSTM:** LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) that is capable of learning long-term dependencies in sequences. In this approach, the IMDb dataset is preprocessed by tokenizing the reviews, building a vocabulary, and padding the sequences to have the same length. The model architecture consists of the following layers:
   - Embedding: Maps each token in the input sequence to a dense vector of fixed size (in this case, 256). The embedding layer learns these dense vector representations during training.
   - LSTM: A LSTM layer with 512 hidden units, batch_first set to True, and dropout rate of 0.25. This layer can capture the long-term dependencies in the input sequence. The model has two LSTM layers stacked on top of each other.
   - Dropout: Applies dropout regularization with a probability of 0.3 to help prevent overfitting.
   - Fully connected (Linear): A fully connected layer that maps the output of the LSTM layer to a single output unit. This layer is used for binary classification (positive or negative sentiment).
   - Sigmoid: A sigmoid activation function that squashes the output of the fully connected layer to a value between 0 and 1, representing the probability of the positive sentiment class.

   The model is trained using the Adam optimizer and BCELoss as the loss function. Early stopping is implemented based on the validation loss to prevent overfitting. Finally, the model is evaluated on the test set, and the classification report and confusion matrix are displayed.

4.2 Post-MidTerm Work

We train and evaluate different Word2Vec models on the IMDB dataset using linear SVM as the classifier. The models are evaluated using three different percentages of training data: 10%, 40%, and 80%.

Stage I: We pre-train custom Implemented Word2Vec models on SST2(Stanford Sentiment Treebank) dataset. We train a CBOW model and a Skip-gram model with negative sampling.

Stage 2: IMDB sentiment classification using pre-trained word embeddings:

1. Load the IMDB dataset and preprocess the reviews, including cleaning up HTML elements, removing non-ASCII characters, emojis, mentions, URLs, punctuation, and converting text to lowercase.
2. Load the pre-trained word2vec models on SST2-Stanford Sentiment Treebank dataset(CBOW and skip-gram with negative sampling:implemented from scratch, gensim library's CBOW and skip-gram with negative sampling, and Google's pre-trained skip-gram model).
3. For each pre-trained word2vec model, convert the IMDB reviews into word embeddings by averaging the word vectors in each review.
4. Train and evaluate a Linear SVM classifier on each set of embeddings to classify the sentiment of the IMDB reviews for each model.

5 Observations

The Implementation can be found here: code link

We analyze the performance of various models (BERT, Naive Bayes, Linear SVC, and LSTM) and (custom-trained Word2Vec models on the SST2 dataset) for IMDB sentiment classification with different training, validation, and test splits. The key observations are as follows:

1. BERT consistently achieves the highest test accuracy and lowest test loss across all splits due to its powerful pre-trained representations. As the percentage of training data increases, BERT's performance further improves.
2. Baseline models like Naive Bayes and Linear SVC show competitive results, especially with smaller training datasets. However, they do not surpass BERT's performance even with more training data.
3. LSTM underperforms compared to other models, but its performance improves with more training data. It requires a larger dataset to achieve comparable results. As generally, LSTMs are more prone to overfitting when trained with small datasets.

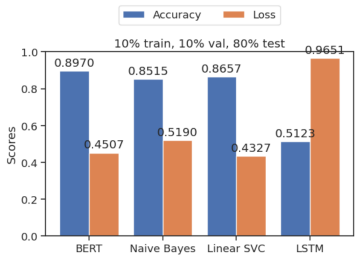

Figure 5: With 10% training, 10% validation, and 80% testing data.

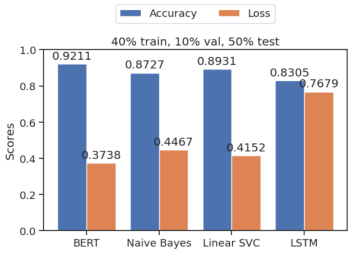

Figure 6: With 40% training, 10% validation, and 50% testing data.

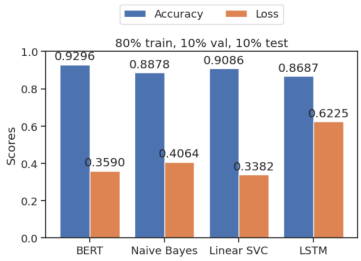

Figure 7: With 80% training, 10% validation, and 10% testing data.

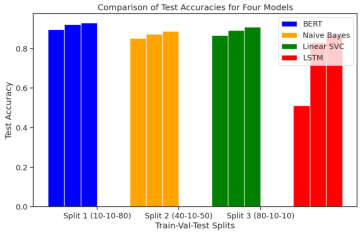

Figure 8: Performance of all four models across the three training splits.

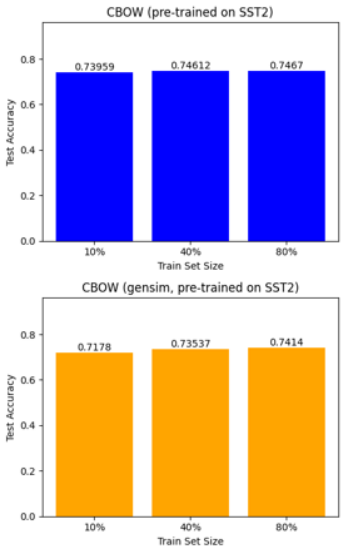

Figure 9: CBOW Comparision (custom vs gensim)

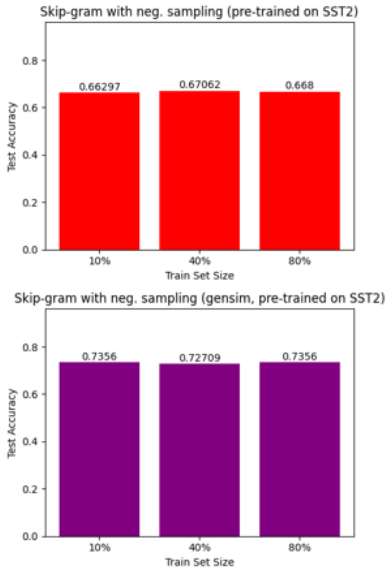

Figure 10: Skip-gram with Neg. Sampling Comparision (custom vs gensim)

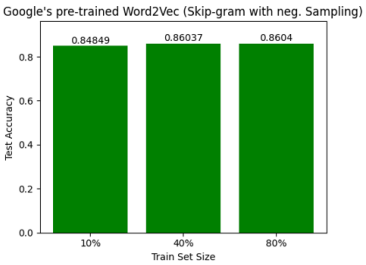

Figure 11: Google's Word2Vec (Skip-gram with Neg. sampling)

Model	10% Train Data	40% Train Data	80% Train Data
LSTM	0.5123	0.8305	0.8687
Naive Bayes (NB)	0.8515	0.8727	0.8878
Linear SVC	0.8657	0.8931	0.9086
BERT	0.8970	0.9211	0.9296
Word2Vec CBOW (Scratch)	0.7396	0.7461	0.7467
Word2Vec CBOW (Gensim)	0.7178	0.7354	0.7414
Word2Vec Skip-Gram (Scratch)	0.6630	0.6706	0.6680
Word2Vec Skip-Gram (Gensim)	0.7356	0.7271	0.7356
Google's Pre-trained Word2Vec	0.8485	0.8604	0.8604

Figure 12: Performance of all models across the three splits.

4. Custom-trained CBOW model achieves the highest test accuracy compared to the CBOW model from the Gensim library, indicating better-suited training parameters for sentiment classification on the IMDb dataset.
5. Custom-trained Skip-gram with negative sampling model underperforms compared to the Gensim model. Differences in performance could be due to varying implementations, training parameters, initialization, or hyperparameters.
6. Google's pre-trained Skip-gram model significantly outperforms all other word2vec models tested, attributed to its extensive training dataset (Google News) and large vocabulary, allowing a better understanding of semantics and relationships between words for accurate sentiment classification.

6 Results

BERT consistently achieves the highest test accuracy across all data splits, showcasing the effectiveness of self-supervised representation learning. Supervised learning models, such as Naive Bayes and Linear SVC, which were trained using TF-IDF representations, also perform reasonably well. Linear SVC slightly outperforms Naive Bayes in all data splits, but both are outperformed by BERT. LSTM shows varying performance, with significant improvements as the training data size increases. CBOW and Skip-gram models, both implemented from scratch aas well as using Gensim with Linear SVM as a classifier yield relatively lower results compared to other models like BERT & Linear SVC, and Naive Bayes.

7 Conclusions

This study showcases the effectiveness of self-supervised representation learning techniques in natural language processing tasks, with BERT emerging as the most effective model, followed by Linear SVC and Naive Bayes. The project also demonstrates the power of custom-trained and pre-trained Word2Vec models, such as CBOW and Skip-gram, in sentiment classification tasks. The performance advantage of BERT and Google's pre-trained Skip-gram model can be attributed to their pre-trained representations and self-supervised learning mechanisms, which enable them to capture complex language patterns more effectively.

On the other hand, Supervised learning algorithms like the Linear SVC and Naive Bayes classifiers show reasonable performance, but they do not benefit from the self-supervised learning mechanism. As a result, their performance is not as impressive as BERT.

In conclusion, this project highlights the importance of leveraging pre-trained representations and self-supervised learning mechanisms for achieving high performance in natural language processing tasks. BERT consistently outperforms traditional machine learning models and LSTM-based approaches, while Google's pre-trained Skip-gram model achieves the highest test accuracy across all Word2Vec models.

8 Future Work

Further experiments can be conducted by exploring the potential of other self-supervised learning models, such as RoBERTa or GPT, and comparing them to BERT in similar tasks. Further investigations into other NLP tasks and fine-tuning models for domain-specific applications could provide insights into the generalizability of self-supervised representation learning techniques. Additionally, other architectures and training techniques for word embeddings, such as GloVe and fastText, can be explored. Investigating the use of transfer learning and fine-tuning with pre-trained models could further improve the performance of sentiment classification tasks and extend the analysis to other NLP tasks and datasets to better understand the performance and generalizability of different pre-trained Word2Vec models.

References

[1] BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding

[2]Efficient Estimation of Word Representations in Vector Space

[3] Reference for Implementations:
Hugging Face
scikit-learn
Few Youtube tutorials.