Comparing the Performance and Efficiency of Pre-Trained Transformers with Convolutional Neural Networks

Michael Liam Sinclair
USC Upstate
February 4th, 2024

ABSTRACT
The amount of user-generated data continues to grow at an unprecedented rate in the modern day. Analyzing public sentiment from websites, social media posts, blogs, review sites, etc. requires carefully digesting human text that contains the many nuances of human speech. Machine learning seeks to solve this issue by applying natural language processing to perform sentiment analysis. Sentiment analysis is the automated process of tagging data according to sentiment by identifying the polarity of a body of text. By leveraging sentiment analysis, individuals and organizations can process large datasets at scale in real-time without manual sorting. This research seeks to compare and analyze the performance of a fine-tuned DistilBERT model with a convolutional neural network model on the task of sentiment analysis.

Keywords
Artificial Intelligence, Machine Learning, Neural Network, Convolutional Neural Network, Transformer Architecture, Deep Learning, Sentiment Analysis, Transfer Learning, DistilBERT

1. INTRODUCTION
As the digital age continues to develop and evolve, individuals, companies, and organizations are faced with the need to evolve with it. The internet has given rise to an incomprehensible amount of data including social media, blogs, review sites, and much more. These data sources provide invaluable insights into people's thoughts and feelings about topics and ideas. However, individuals who want to gather insights related to sentiment require a means of processing data in ways that accurately reflect the actual sentiment present. Natural Language Processing (NLP) is a recent technical leap that allows sentiment analysis to be performed on massive sets of data without the need for manual sorting or other labor-intensive measures [15]. Sentiment analysis is an NLP technique that identifies the polarity of a body of text and tags it according to defined sentiment labels, most popular of which are simply positive, negative, and neutral. By utilizing sentiment analysis on user-generated data such as blogs, social media posts, review sites, etc. individuals, companies, and organizations can process large datasets at scale in real-time, allowing them to quickly understand public sentiment and make data-driven decisions. Common applications include analyzing social media mentions to analyze public sentiment, analyzing feedback from surveys and product reviews, and analyzing incoming support tickets to immediately detect and handle upset customers [10].

For this research, one of the models will be a convolutional neural network (CNN). The other model used in this research is a fine-tuned DistilBERT model, which is a distilled version of BERT [13]. Given that DistilBERT is still a large model despite being a distilled version of BERT, this raises the issue of possible computational constraints and resource usage. For this reason, this research will compare the performance of a CNN with a fine-tuned version of DistilBERT. Each model will undergo an efficiency, robustness, generalization, and error assessment. These results, along with the results gathered against the performance metrics, will determine the practicality of deploying each model in practice, considering factors such as latency, scalability, and ease of integration. Sentiment analysis has been chosen as the focal point due to its critical importance in processing and understanding the nuances of human language, which is paramount for accurately gauging public sentiment. By dissecting the strengths and limitations of these models, this study seeks to provide insights into their practical application in real-world scenarios, contributing to the advancement of NLP technologies.

2. LITERATURE REVIEW
Accurate sentiment analysis is not without its challenges and properly identifies key challenges [15] juxtaposed forty-seven research efforts to identify the most pressing challenges and their effects on sentiment analysis accuracy. Review structure played a critical role in selecting challenges to focus on, given that domain-oriented challenges required an understanding of the topic's specific features or keywords.

Many research efforts employed techniques such as POS tagging, lexicon-based methods, Bag-of-Words, and n-grams, each carrying varying success rates. Negation (such as "not good" is the opposite of good), sarcasm, and ambiguity within the text have also been identified as key challenges posed by sentiment analysis that call for further research. [5] also supports the notion that context, ambiguity, sarcasm, domain-specific language, informal language, and slang continue to pose a challenge for sentiment analysis. Two evaluation metrics used to measure the quality of NLP models are BLUE score (for language translation) and GLUE (for comparing performance across various NLP tasks). GLUE is comprised of nine tasks that evaluate different aspects of language understanding, such as sentence relationships, logic, and context. The final GLUE score is an average of scores on each task. The BLUE and GLUE evaluations have aided in developing more sophisticated and accurate NLP models [5].

"Attention Is All You Need" by [10] marked a significant advancement in machine learning, leading to the development of large language models like BERT. DistilBERT, introduced by [13] in March 2020, is a streamlined version of BERT that is 40% smaller, retains 97% of BERT's language understanding capabilities, and operates 60% faster due to distillation during pre-training. This approach reduced computational costs while maintaining high performance, demonstrated by DistilBERT's GLUE benchmark score of 77.0 compared to BERT's 79.5. Despite its reduced size, with 66 million parameters versus BERT's 110 million, DistilBERT performs comparably to BERT and is more efficient on lower-end devices, proven by its 71% faster operation on an iPhone 7 Plus [13]. The importance of pre-training, as outlined by [15], lies in its ability to enable transfer learning, minimize the need for large labeled datasets, and prevent overfitting on smaller datasets.

As for CNNs, while they are typically suited for tasks with visual data requiring the analysis of complex patterns, they do have several advantages in sentiment analysis when trained properly. [5] highlights CNN's powerful ability to learn hierarchical features, adaptations to various input sizes, and generalizing skills. Some of the limitations of CNNs [5] however are its sensitivity to training data quality and the potential need for a generous amount of data. This introduces cost considerations such as hardware, dataset size, model complexity, and distributed computing efforts to improve efficiency and mitigate costs. Recent developments include attention mechanisms and transfer learning as previously mentioned, as well as capsule networks, quantization, and compression that all aim to enhance CNN's efficiency, reliability, and robustness. CNNs have significantly advanced image recognition fields and continue to evolve for improvements in interpretability, robustness, and the ability to handle more complex tasks [5].

3. METHODOLOGY

To create an effective model, it is critical to consider its architecture, system design, data preparation, and learning techniques. All these factors must be optimized to develop a model that produces accurate and reliable results, and this section will cover each in detail.

3. 1 Convolutional Neural Networks

Natural Language Processing (NLP) predominantly utilizes neural networks that operate similarly to human brains. Through learning from data, these networks enhance their performance. Instead of neurons, NLP networks employ nodes, and their intricate connectivity enables swift data processing.

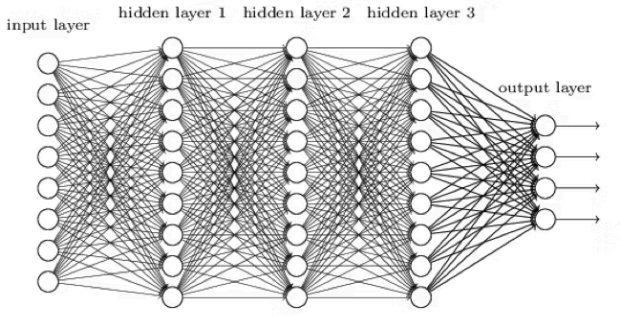

Figure 1. Basic Neural Network Architecture

A neural network has input, hidden, and output layers with nodes that receive input values, act in specific ways based on the data and pass signals to the next layer. Each connection between nodes has a weight that allows the network to emphasize specific nodes. Neural networks are capable of more complex operations and error correction through backpropagation. These features allow neural networks to predict accurately and solve complex problems like a brain [5].

Convolutional neural networks (CNNs) are characterized by convolutional layers that perform feature extraction and pooling layers. In NLP, CNNs learn word or phrase patterns for sentiment analysis. By stacking multiple convolutional layers, CNNs learn patterns of higher complexity and understand higher-level linguistics. CNNs also identify the relationship between adjacent words or n-grams for NLP by using local dependency capture, which is essential for context understanding. Furthermore, the convolution operation uses the same set of weights across the entire input, making CNNs more efficient at processing large datasets and reducing the number of parameters needed to learn features [5].

3.2 Convolutional Neural Network Layers

The Convolutional Neural Network built for this research employs an architecture with two convolutional layers. Given that the model's objective is sentiment analysis followed by binary classification, an approach utilizing fewer convolutional layers is preferable to reduce the risk of overfitting and reduce model complexity [5].

The model architecture in question consists of several key layers, each designed to extract meaningful patterns from text data. The first layer is an Embedding layer, which leverages pre-trained GloVe weights to capture general linguistic representations. Specifically, the GloVe weights are set to a dimensionality of 200 to enable the model to capture a wide scope of semantic meanings. This layer is non-trainable to preserve these meanings.

Following the Embedding layer is a Conv1D layer, which is configured with 64 filters and a kernel size of 5. It utilizes the ReLU activation function to extract local patterns within text sequences. To ensure stable learning throughout training, a Batch Normalization layer follows the convolutional layer. The Batch Normalization layer is then followed by a GlobalMaxPooling1D layer, which serves to reduce dimensionality and computational complexity.

The next layer in the network is a Dense layer of 32 units that applies L2 regularization to reduce the risk of overfitting, which is further applied to the following Dropout layer with a rate of 0.5 for regularization. Finally, the output layer for this model is a Dense output layer that employs the softmax activation function to produce probability distributions for binary classification.

This particular design for the CNN was chosen for its efficiency in text processing and its proven effectiveness in similar sentiment analysis tasks, as supported by existing literature.

3.3 Transformers

In 2017, Vaswani et al. [16] introduced Transformers, a new approach to deep learning for NLP. They use word embeddings to represent words as vector values, and positional embeddings to encode the position of each token in a sequence. This allows the

model to capture the order and position of words in a sequence, which is important for transformers where words are not processed in a sequence. Transformers use a seq2seq neural network architecture with encoder and decoder layers. Each encoder layer extracts features from a sequence, and each decoder layer uses these features to produce an output sequence. Attention is used for encoding and decoding within transformers [16].

Attention helps the model understand the context of a word by considering words that go before and after it. An example of this would be how the word bark can be connected to dogs while also being connected to tree bark. The context of other words in the sequence helps the model understand different meanings of words and process them correctly. Self-attention is an entirely different mechanism that works with the sequence that is being encoded, while attention works with the sequence that is being generated by the decoder. Both attention and self-attention allow the model to weigh the importance of different parts of the input sequences against each other. This is done by computing a set of attention weights that indicate the relevance of each element in the input to every other input and allows the model to capture long-range dependencies as well as learn to recognize patterns that span multiple elements. Multi-head attention allows the network to learn multiple ways of weighing the input sequence against itself [16].

3.4 DistilBERT

As previously mentioned DistilBERT was chosen as the model to be used due to its reduced size and complexity while still retaining most of BERT's performance. The approach to fine-tuning DistilBERT for sentiment analysis differs from that of CNN given that the model is already created and only requires fine-tuning. The 'AutoModelForSequenceClassification' class from the Hugging Face Transformers library adapts DistilBERT for sequence classification by adding a softmax layer on top for class prediction, similar to the softmax layer used for the CNN. The training techniques utilized for DistilBERT fine-tuning are very similar to those used for training the CNN and will be covered in Section 3.6 [13].

3.5 Tokenization and Word Embeddings

Tokenization is a preprocessing step that transforms text into a format that models and algorithms can process. Tokenization involves dividing the text into individual words or symbols, followed by other cleaning steps such as removing punctuation or normalization (converting all characters to lowercase for consistency). Word embeddings are dense vectors that capture semantic meanings and relationships behind words, allowing algorithms to better understand text data. They are used as the input layer in many machine-learning models and are often generated using large datasets [14].

The CNN and DistilBERT use different word embeddings. CNN uses pre-trained GloVe embeddings which capture global word-word co-occurrence statistics to produce word vectors. DistilBERT offers contextual embeddings using attention mechanisms and transformer architecture, allowing it to perform more nuanced language comprehension. GloVe embeddings used in the research are a pre-trained set trained on 2 billion tweets with a vocabulary of 1.2 million and a dimensionality of 200. These embeddings are used to pre-load the model with a rich understanding of word semantics and are designated as trainable [11].

3.6 Learning Techniques

To optimize training and mitigate risks such as overfitting, many learning techniques have been applied for training both models. Early stopping, learning rate scheduling, and dropout are critical techniques in training deep learning models as they help in managing overfitting, ensuring efficient learning, and assisting in achieving a model that generalizes well to unseen data

Early Stopping

Early stopping is a technique that attempts to stop training before the model begins to overfit. Early stopping works by monitoring a specific metric, such as validation loss, during training and if the metric stops improving for a certain number of epochs, or steps, training is halted. This prevents overfitting by stopping training when the model no longer improves on the validation dataset and saves computational resources that would otherwise be wasted.

Learning Rate Scheduling

Learning rate scheduling adjusts the learning rate during training by decreasing it based on certain rules or metrics, such as after a certain number of epochs or when the improvement plateaus. This technique helps both models better learn by allowing finer adjustments as they get closer to the optimal solution. Additionally, learning rate scheduling prevents the training process from stalling or getting stuck in local minima by adapting the step size.

Dropout

is a regularization technique that also aims to prevent overfitting by randomly setting a fraction of input units to zero at each update during training time, which helps to break up happenstance correlations in the training data. By randomly dropping units, it ensures that no single unit becomes too reliant on the output makes the model more robust and improves generalization. Dropout is a manually defined technique applied to the CNN, and for DistilBERT it is implemented in the transformer architecture itself.

4. IMPLEMENTATION

The entirety of this research is programmed in Python and run using Google Colab's V100 GPU for consistency. The NVIDIA V100 Tensor Core GPU utilizes the following specifications:

Clock rate	1.53 GHz
Number of Cores	80
Memory (GB)	16 GB
Max Memory Bandwidth	900 GB/sec

Numerous libraries are used for key steps and processes for development, and these libraries enable for the efficient and optimal development of each model.

TensorFlow and Keras

TensorFlow is an open-source machine learning library widely used for building and training neural networks. Within the TensorFlow library Keras, a high-level neural networks API, is used for tokenization, model initialization, and CNN training [6].

Scikit-learn
Scikit-learn is a Python library for machine learning that provides various algorithms and tools for data mining, data analysis, classification, model evaluation, etc [6].

Pandas
Pandas is a Python library for data manipulation and analysis and provides data structures and operations for manipulating numerical tables and time series. For this research, the DataFrame data structure is used to cleanse, preprocess, and prepare the dataset for training and evaluation [6].

NumPy
NumPy is a fundamental package for scientific computing that offers support for large multidimensional arrays and matrices, as well as a variety of mathematical functions to operate on them. NumPy serves as the backbone for numerical computations within this research [6].

Pickle5
Pickle5 is an updated version of the pickle module for Python with improvements and optimizations for serializing and deserializing Python object structures. It is utilized in this research for saving and loading models and data transformations [6].

Datasets
Datasets is a lightweight library provided by Hugging Face and provides easy access to large-scale datasets and evaluation metrics for NLP. This library streamlines data preprocessing, loading, and caching for DistilBERT training after the dataset is loaded by Pandas [4].

Accelerate
Accelerate is a library developed by Hugging Face that allows for GPU acceleration and abstracts the complexity of accelerating computation. This allows for more efficient training of DistilBERT on the large dataset [4].

Transformers Library
The Transformers library, developed by Hugging Face, provides the DistilBERT model as well as thousands of other pre-trained models for a wide range of NLP tasks as well as a simple interface for interacting with and using transformer models. The Transformers library is used in this research for obtaining the pre-trained model, training, model use, testing, and evaluations [4].

Hugging Face's Trainer API
The Trainer API from Hugging Face simplifies the process of training, evaluating, and fine-tuning machine-learning models for NLP tasks. It integrates with the Transformers library and offers an interface for training DistilBERT on custom datasets with support for the learning techniques used in this research [4].

5. EXPERIMENTAL SETUP
The following sections detail the data preparation, evaluation metrics, and experiment steps that have been taken to ensure replicability and document the entire research process.

5.1 Data Preparation
The dataset used for training both models is the Sentiment140 dataset with 1.6 million tweets. This dataset was obtained through Kaggle and prepared by [3]. The dataset is preprocessed by first removing any URLs, mentions, and hashtags. Any emojis are then converted to text using the Emoji library in Python, and finally, the text is processed into a data frame to be used for training (Pandas data structure). The prepared dataset is converted from the pandas data frame to a Hugging Face dataset for compatibility with training the DistilBERT model. The dataset is then further prepared by converting sequences into features, padding the sequences to a fixed length of 100 tokens, and tokenization (the DistilBERT training uses a maximum length of sequences of 128 tokens). The tokenizer for both models is initialized with a maximum vocabulary of 20,000 and a <OOV> token for words out of vocabulary. One-hot encoding is then applied with two classes, positive and negative, and the dataset is split into training and testing sets.

For the CNN, the GloVe embeddings are then iterated over the following data preprocessing where each line is split into values, the words are extracted, the values are converted to a NumPy array, and finally, the word embedding is stored in the dictionary. The dimensionality of the word embeddings for the CNN is set to 200 and the values from the set of word embeddings are applied to the embedding matrix that is used for training.

5.1 Evaluation Metrics
Several evaluation metrics are applied to both models during and following training that provide valuable insights into each model's performance.

Loss and Accuracy
Loss and accuracy are both key metrics that are used during training to evaluate the model's performance on generalizing unseen data, and the values of each metric influence the learning rate as well as possible early stopping applied. Accuracy is a straightforward measurement of the proportion of total predictions that were correctly identified, and loss, or cost quantifies how much predictions made by the model deviate from the actual labels [7].

F1 Score
The F1 score combines precision, the proportion of true positive results in all positive predictions, and recall, the proportion of true positive results in all actual positives, into a single metric [7].

AUROC and AUPRC
AUROC (Area Under the Receiver Operating Characteristic Curve) quantify the trade-off between the true positive rate and the false positive rate at various threshold settings. AUROC provides a single measurement of overall model performance across all classification thresholds with 0.5 indicating predictions are no better than random change, and 0.1 indicating perfect predictions. AUPRC (Area Under the Precision-Recall Curve) measures the relationship between precision and recall for every possible cut-off with higher areas indicating a high proportion of positives while minimizing false positives [7].

Efficiency
Efficiency, in this context, refers to how resourcefully the model performs in terms of computation time and memory usage. The most efficient model achieves high performance with minimal computational resources and time. Efficiency in this research is calculated by the various ratios of processing time and resource consumption with the accuracy and performance that the model provides [7].

Robustness
Robustness measures each model's ability to maintain high performance when exposed to noisy data, adversarial examples, or

conditions it wasn't specifically trained. Adversarial testing, noise injection, and out-of-distribution performance are the evaluations that will be applied to each model to measure robustness [7].

5.2 Experiment Steps

Each model follows the same general processes for data preparation, model training, and model evaluation, though the individual steps do contain variations.

Convolutional Neural Network

The experiment process begins with installing and importing the required libraries: TensorFlow, scikit-learn, Pandas, NumPy, and Pickle5 followed by preparing the Google Colab environment. The data is then loaded and preprocessed, including tokenization and padding of text data, and the data is split into training and testing sets, and the GloVe embeddings are prepared as described above.

The model is then constructed as a Sequential model with the layers previously outlined, and hyperparameters (dropout, L2 regularization, and learning rate) that have been tuned using Hyperopt, a Python library. The model is compiled with the Adam optimizer and categorical cross-entropy loss. The callbacks mentioned previously are implemented with the addition of model checkpoints (saving the best-performing weights) and a learning rate scheduler that reduces the learning rate on performance plateaus. The CNN is then trained on the test set and evaluated on the test set to obtain the loss and accuracy. The model is then further evaluated using the evaluations and evaluation metrics previously mentioned using scikit-learn.

DistillBERT

The DistilBERT training process begins with installing and importing the Transformers, Datasets, Accelerate, Torch, Pandas, NumPy, and git-lfs (for handling large files) libraries. The Google Colab environment is then prepared, followed by dataset preprocessing and preparation as previously described and the dataset is split into training and testing sets.

The DistilBERT model is then initialized for sequence classification with two labels (positive and negative). The data collater is also initialized for dynamic padding of tokenized inputs. The Trainer API is then configured using 'TrainingArguments' with configurations for evaluation strategy, learning rate scheduling, batch sizes, and early stopping criteria. The 'Trainer' is then initialized with the model, arguments, datasets, tokenizer, data collator, and the compute metrics function. The model is then trained using the 'Trainer' instance and evaluated on the test dataset. Following training, the model is evaluated using the evaluations and evaluation metrics previously described using the Trainer API as well as scikit-learn.

6. REFERENCES

[1] Arora, S. (2022, July 7). Sentiment Analysis Using Python. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2022/07/sentiment-analysis-using-python/

[2] Gavrilova, Y. (2023, April 10). Transformers in ML: What They Are and How They Work. Serokell Software Development Company. https://serokell.io/blog/transformers-in-ml

[3] Go, A., Bhayani, R. and Huang, L., 2009. Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford, 1(2009), p.12.

[4] Hugging Face. (n.d.). 🤗 Transformers. Huggingface.co. Retrieved March 1, 2024, from https://huggingface.co/docs/transformers/en/index

[5] Khurana, D., Koli, A., Khatter, K., & Singh, S. (2022). Natural Language processing: State of the art, Current Trends and Challenges. Multimedia Tools and Applications, 82(3), 3713–3744. https://doi.org/10.1007/s11042-022-13428-4

[6] Logunova, I. (2023, October 31). A Guide to Transfer Learning. Serokell Software Development Company. https://serokell.io/blog/guide-to-transfer-learning

[7] Mishra, A. (2018, February 24). Metrics to Evaluate your Machine Learning Algorithm. Medium; Towards Data Science. https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234

[8] Moez Krichen. (2023). Convolutional Neural Networks: A Survey. Computers, 12(8), 151–151. https://doi.org/10.3390/computers12080151

[9] Mohey El-Din Mohamed Hussein, D. (2018). A Survey On Sentiment Analysis Challenges. Journal of King Saud University - Engineering Sciences, 30(4), 330–338. ScienceDirect. https://doi.org/10.1016/j.jksues.2016.04.002

[10] Pascual, F. (2022, February 2). Getting Started with Sentiment Analysis using Python. Huggingface.co. https://huggingface.co/blog/sentiment-analysis-python#3-building-your-own-sentiment-analysis-model

[11] Pennington, J., Socher, R., & Manning, C. D. (2014). GloVe: Global vectors for word representation. 1532–1543. http://www.aclweb.org/anthology/D14-1162

[12] Roy, R. (2023, February 23). Best Python libraries for Machine Learning. GeeksforGeeks. https://www.geeksforgeeks.org/best-python-libraries-for-machine-learning/

[13] Sanh, V., Debut, L., Chaumond, J., & Wolf, T. (2020). DistilBERT, a distilled version of BERT: smaller, faster, cheaper, and lighter. Arxiv.org. https://doi.org/10.48550/arXiv.1910.01108

[14] shristikotaiah. (2024, January 5). Word Embeddings in NLP. GeeksforGeeks. https://www.geeksforgeeks.org/word-embeddings-in-nlp/

[15] Subramanyam, K., Katikapalli, Ajit, R., & Sivanesan, S. (2021). AMMUS : A Survey of Transformer-based Pretrained Models in Natural Language Processing. ArXiv.org. https://doi.org/10.48550/arXiv.2108.05542

[16] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. ArXiv.org. https://doi.org/10.48550/arXiv.1706.03