Comparative Analysis of Transformers to Support Fine-Grained Emotion Detection in Short-Text Data

Robert H Frye and David C Wilson
University of North Carolina at Charlotte
Charlotte, NC

Abstract

Understanding a person's mood and circumstances by way of sentiment or finer-grained emotion detection can play a significant role in AI systems and applications, such as in chat dialogue or reviews. Analysis of emotion from text typically requires specialized text or document understanding, and recent work has focused on transformer learning approaches. Common models of these transformers (e.g. BERT, RoBERTa, ELECTRA, XLM-R, and XLNet) have been pre-trained using longer texts of well-written English; however, many application contexts align more directly with social media content or have a shorter format more akin to social media, where texts often bend or violate standard language conventions. To understand the applicability and trade-offs among common transformers within such contexts, our research investigates accuracy and efficiency considerations in fine-tuning transformers for granular emotion detection in short-text data. This paper presents a comparative study investigating the performance of five common transformers as applied in the specific context of multi-category emotion detection in short-text Twitter data. The study explores different considerations for hyperparameter settings in this context. Results show significant fine-tuning benefits in comparison to recommended baselines for the approaches and provide guidance for fine-tuning to support fine-grained emotion detection in short texts.

Introduction

Understanding a person's mood and circumstances by way of sentiment analysis or finer-grained emotion detection can play a significant role in AI systems and applications. Yue et al. (Yue and others 2019), for example, note commercial, political, and public security as three important application areas. Commercial applications can provide valuable reaction-based feedback to merchants, manufacturers, and consumers about the quality and reception of different products. In politics, sentiment and emotion data can help inform political strategies for both candidates and incumbents. And recognizing sentiment-driven trends in social media can help to understand emerging, potentially disruptive world events.

The task of correctly identifying specific emotions portrayed in written text is challenging, even with richer data where the text is longer, well-written and closely follows rules of grammar, spelling, punctuation, and style. Given that texts for user interactions in modern communication are often shorter and more aligned in structure with social media interactions, there are additional challenges to the detection task. For example, Hassan et al. (Hassan, Abbasi, and Zeng 2013) note that "...Twitter harbors a lot of noise, including spam, the short colloquial communication style adopted by users, irrelevant content, and an abundance of neutral content." More generally, noise issues in short-text online communication include: misspellings, inconsistent and repeated punctuation, use of emojis and emoticons, user tags, hashtags, URLs, and case mixing.

For example, consider the first entry from our dataset: "t-minus 10 minutes until interview timeeeee. #nervous". There are differences from formal style in terms of sentence length and general content, including the lack of capitalization, misspelling of time, and use of a hashtag to self-label the tweet. Considerations such as these are important aspects of context in typical short-text online communication, and need to be accounted for in emotion detection, particularly when the leading available text analysis components are foundationally trained on longer, more formal texts.

In building emotion-aware applications, there are important tradeoffs to consider in the use of standard language processing components as part of the architecture. Pre-trained components are commonly available as part of natural language understanding packages. Devlin et al. (2019) note that "Recent empirical improvements due to transfer learning with language models have demonstrated that rich, unsupervised pre-training is an integral part of many language understanding systems." At the same time, however, the corpora used for such training typically focus on longer, well-written sources, such as the English version of Wikipedia and BOOKCORPUS (Zhu and others 2015), CCNews-En (Mackenzie and others 2020), STORIES (Trinh and Le 2019), and OpenWebText (Peterson, Meylan, and Bourgin 2019) datasets. Since the foundational models for common transformers are based on longer text corpora with more formal writing style, there are potential trade-offs when they are applied for understanding in shorter-text social media contexts, either directly or with fine-tuning.

Our long-term research program is investigating hybrid classification models for fine-grained emotion detection in short-text data. As a foundation, it is essential to understand performance and trade-offs in best-practice baseline components. This paper investigates potential trade-offs and applicability of modern, commonly available transformers when applied in short-text media contexts for the task of fine-grained emotion detection. Our key questions were to understand the performance of models that were pre-trained on longer, more formal texts in the shorter-text social media context, as well as to investigate appropriate hyperparamter settings for emotion detection in the short-text context. The remainder of this paper is organized as follows. First, we discuss related work, including transformers and deep learning algorithms for emotion detection in written text, as well as hyperparameter optimization. Second, we present our experimental approach. Finally, we present our results and analysis along with conclusions and future work.

Related Work

Our research focuses on commonly used deep learning transformer-based approaches and hyperparameter optimization as part of the classification and prediction steps for emotion detection. The approaches discussed have all been applied to the task of sentiment analysis, and each of the transformer approaches specifically references the SST-2 task of GLUE (Wang et al. 2019). The related work here covers both common types of neural networks employed, as well as the specific transformer approaches that have been applied in sentiment analysis and finer-grained emotion detection.

Deep Learning Algorithms

Convolutional neural networks (CNNs) were originally proposed by Lecun et al. (LeCun et al. 1999) for applications in image recognition. CNNs use filters of progressively smaller selections of an image to map the features from each larger filter to the smaller selection in each subsequent layer. A softmax layer is then used to map the preceding layers to the final categories in a classification task. Yoon Kim (Kim 2014) mapped the CNN approach from the field of image recognition to the problem of text classification, and parallel work by other researchers further established the viability of CNNs for text classification problems (Liu et al. 2017; Johnson and Zhang 2015; 2017). As an example for emotion detection, consider the following phrase. “No fan of the Carolina Panthers has ever said watching their games is a relaxing experience.” From this statement, we can infer that a fan may find most Panthers games stressful to watch; however, if we consider only the phrase, “...watching their games is a relaxing experience,” in isolation, we may draw a much different conclusion. Thus, the context between phrases is important for emotion detection.

Recurrent neural networks (RNNs) are designed to maintain some memory of the outcome of previous steps. In RNNs, the output of a previous step in a sequence applies to the next word in a sequence, enabling an RNN to consider the context of an entire word sequence. RNNs are vulnerable to the problem of vanishing or exploding gradients, and thus are limited to remembering only a small number of steps. Hochreiter and Schmidhuber (Hochreiter and Schmidhuber 1997) developed the long short-term memory (LSTM) network, which uses 4 layers per time step, including an input gate, output gate, forget gate, and hidden memory layer to bypass the vanishing or exploding gradient problem inherent to RNNs. Cho et al. (Cho and others 2014) proposed GRU, a simplified version of LSTM which combines the forget and input gates and merges the hidden and cell states. Schuster and Paliwal (Schuster and Paliwal 1997) proposed bidirectional RNNs to capture the context before and after a sequence. In their approach, one RNN processes input in the original order and the other processes the input in reverse order. Ghosh et al. (Ghosh et al. 2016) extended LSTM with the concepts of CNNs (LeCun et al. 1999) by adding memory of the class to each gate in the LSTM layer. In their work on topic modeling classification, C-LSTM improved on the performance of traditional LSTM by approximately 20%.

Transformers

Devlin et al. (Devlin et al. 2019) created BERT (Bidirectional Encoder Representations from Transformers) to capture the bidirectional contextual information inherent in text. BERT was pre-trained using the 800M words of the BooksCorpus (Zhu and others 2015) and 2.5B words of English Wikipedia. BERT was developed using a 30K token vocabulary based on WordPiece embeddings (Wu and others 2016). Part 1 of BERT’s pre-training was unsupervised training, conducted by randomly masking 15% of the input tokens and then predicting the masked tokens – a so-called masked language model (MLM) approach. Part 2 of pre-training was meant to enable an understanding of sentence relationships using the unsupervised training of a next sentence prediction task. BERT generated pre-trained embeddings based on the sum of token, segmentation, and position embeddings, and thus captures information from each pre-training task in the final embedding vector. BERT was trained with batches of 256 tokens. The original BERT approach showed substantial improvement on the General Language Understanding Evaluation (GLUE) (Wang et al. 2019) score by 7.7% absolute improvement, and 91.6% for binary (positive/negative) sentiment prediction when combined with BiLSTM.

Liu et al. (2019) extended the concepts of BERT transformers in RoBERTa (Robustly Optimized BERT Pretraining Approach). RoBERTa was trained with the CC-NEWS dataset, a derivation compiled from Nagel’s CommonCrawl News dataset (Nagel 2016). The RoBERTa approach matched or improved on BERT results in 6 of the 9 GLUE tasks. Liu et al. increased several training hyperparameters for BERT, including the length of training and the batch sizes used. RoBERTa was trained with maximum batch sizes of 512. For ablation testing, Liu et al. removed the next sentence prediction task, adapted the masking pattern dynamically, and trained on longer sequences. In evaluating RoBERTa, the team reported a best accuracy in binary sentiment analysis (the SST-2 task from GLUE) of 92.9%.

Yang et al. (2020) compared their XLNet algorithm to BERT, and differentiated their approach from BERT in several key ways. XLNet’s training method considers permutations of factorization orders to capture the bidirectional context of tokens within text and maximize the logarithmic likelihood of a token sequence in regards to the permutations. As an autoregressive approach to pre-training, XLNet uses the product rule to factor join probabilities of predicted tokens, thus avoiding the token/mask independence discrepancy in BERT. Finally, XLNet further differentiates its approach from BERT in not introducing tagging noise to the dataset. XLNet was trained on many of the same or similar datasets as BERT and RoBERTa, including CommonCrawl, BooksCorpus, and English Wikipedia, while also training with the ClueWeb 2012-B (extended from (Callan et al. 2009)) and the Giga5 (Parker et al. 2011) datasets. We note that Yang et al. intentionally filtered their training data to remove short, low-quality articles, arguably representative of social media type data. XLNet was trained with 512 token training sequences and, like RoBERTa, dropped the next sentence prediction task when compared to BERT. XLNet reported 94.4% accuracy in the SST-2 binary sentiment prediction task, higher than both BERT and RoBERTa.

Cross-lingual language models (XLM) (Lample and Conneau 2019) were developed to extend the success of pretraining approaches like BERT to multiple languages. XLM was trained using the XNLI dataset (Conneau and others 2018) with 7500 human-annotated samples from 15 languages. XLM differs from BERT by considering text streams truncated at 256 tokens, composed of an arbitrary number of sentences, instead of sentence pairs. XLM attempts to address the imbalance between frequent and rare tokens by sampling the tokens in a stream based on their multinomially distributed weight in proportion to the square root of their inverted frequencies. For translation language modeling, XLM extended the MLM approach to consider pairs of parallel translated sentences, thus generating predictions based on contextual clues from either language.

XLM-RoBERTa (XLM-R), developed by Conneau et al. (Conneau et al. 2020), applied concepts presented by both BERT and XLM. Specifically, XLM-R was trained with MLM using monolingual sample streams and a larger vocabulary than BERT, with 250K tokens, compared to 30K with BERT. 100 different languages were sampled with the same distribution used in XLM, with $\alpha = 0.3$ instead of $\alpha = 0.5$, and without language embeddings. A clean derivation of the CommonCrawl Corpus was used in pretraining XLM-R, with one English version and twelve versions inclusive of other languages. Conneau et al. reported 95.0% accuracy on the SST binary sentiment classification task.

Clark et al. (Clark et al. 2020) developed ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately) to offset a perceived weakness of BERT, namely, that BERT has an imbalance caused by the introduction of masking tokens during the pre-training phase, but not during the fine-tuning phase of training. While the pre-training corpus is not specifically noted in (Clark et al. 2020), the appendix notes running an MLM comparison between ELECTRA-Base and BERT-Base using a combined Wikipedia and BooksCorpus dataset. BERT was actually more accurate in the MLM comparison, with a 77.9% accuracy compared to ELECTRA's 75.5%. ELECTRA attempts to offset the BERT imbalance between training and fine-tuning by replacing some tokens with samples from a proposed distribution created by a small MLM. Pre-training is then conducted to predict every token, whether an original token or a replaced token. This differs from BERT, wherein pre-training is only applied to the masked 15% of all tokens, and enables ELECTRA to be pre-trained faster. ELECTRA reported a top SST accuracy between 89.1% to 96.7%, with accuracy variations determined by the fine-tuning datasets used and the duration of training.

Hyperparameter Optimization

Elshawi et al. (2019) note that the process of hyperparameter optimization is one of the key challenges in developing accurate models specific to a given domain. Murray et al. (2019) examined auto-sizing various components of the transformer architecture, including gradients, attention heads, and feed forward networks, in order to streamline the hyperparameter optimization process. Yang and Shami (2020) completed a comprehensive survey of hyperparameter optimization approaches for traditional machine learning and deep learning models and identified the learning rate, dropout rate, batch size, and number of epochs as key hyperparameters in need of tuning for deep learning algorithms. They describe numerous optimization approaches, including trial and error (Abreu 2019), grid search (Koch et al. 2017), and random search (Bergstra and Bengio 2012), as well as various libraries designed to assist in their implementation.

It is notable that while each of the various transformer algorithms we considered provided some insight into how their models were developed, details on hyperparameter selection explored were limited. For example, in (Devlin et al. 2019), Appendix A.3 notes a constant dropout of 0.1, batch sizes of 16 or 32, Adam learning rates of 5e-5, 3e-5, or 2e-5, and either 2, 3, or 4 learning epochs, stating that these values were found to "...work well across all tasks...". In addition, it is noted that datasets with more than 100k training samples were less sensitive to hyperparameter choices, and that fine-tuning was generally fast enough that it was reasonable to simply iterate through the different hyperparameter options and pick the model with the best results. We interpret this as a recommendation that a basic grid search (Koch et al. 2017) approach is expected to be sufficient for hyperparameter optimization.

Methodology

Key questions in our comparative analysis were to understand the performance of models that were pre-trained on longer, more formal texts in the shorter-text social media context, as well as to investigate appropriate hyperparamter settings for emotion detection in the short-text context.

Data

Experimental analysis employed a 1.2M tweet dataset (Wang et al. 2012) as labeled for emotion detection in (Ranganathan and Tzacheva 2019), which we refer to as the HT (HarnessingTwitter) dataset. Obtaining Twitter datasets can be challenging, as Twitter only allows the sharing of tweet IDs between researchers, requiring each research team to independently scrape the required tweets by ID. For this reason, the original HT dataset contained more than 2M tweets; however, the content for only 1.2M remained accessible from Twitter at the time of our study. The HT dataset contained samples of 7 total emotions, distributed as shown in Table 1:

joy	349,419	thankfulness	72,505
sadness	299,412	fear	65,010
anger	261,806	surprise	11,978
love	153,017		

Table 1: Sample sizes for emotions in HT dataset

Preprocessing Data was preprocessed as follows. First, we removed URLs, username references (e.g. @POTUS), hashtags (e.g. #happy), and numbers from the content of each tweet, replacing each complete reference with an empty string. Next, we cast all text to lowercase, unescaped any HTML escape strings, removed punctuation duplicates, and replaced contractions with full phrases (e.g. "what's" was replaced with "what is,""ve" was replaced with " have," and "n't" was replaced with " not."), and stripped extra whitespace from the beginning and end of each string. Finally, we lemmatized the verbs in each tweet.

Software Libraries

Our experiments employed standardized platform implementations of different transformers within the Simple Transformers library, which leverages the following common Python libraries: *scikit-learn - train-test splits and analytics* (Pedregosa and others 2011); *HuggingFace's Transformers - basis of Simple Transformers* (Wolf and others 2020); *Simple Transformers - easy implementation of transformers* (Rajapakse 2019); *Keras Tensorflow - basis of HuggingFace's Transformers* (Chollet and others 2015); *Pandas - dataframe manipulation* (pandas development team 2020); *NLTK - preprocessing text* (Loper and Bird 2002); and *Numpy - array-based math* (Harris and others 2020).

Experimental Setup

We selected BERT (Devlin et al. 2019), RoBERTa (Liu and others 2019), XLM-R (Conneau et al. 2020), XLNet (Yang et al. 2020), and ELECTRA (Clark et al. 2020) transformers as a representative sampling of modern transformer approaches. In all cases, our training approach was to fine-tune the base models available from the Keras Tensorflow libraries for the algorithms above with a 70/30 split of the 1.2M tweet HT dataset and a different random seed than those used in our 10-fold cross validation testing. The Simple Transformers library (Rajapakse 2019) has a default dropout of 0.1. This aligns with the dropout rate consistently employed in related work, and so was kept constant here.

Recommended Parameter Testing We trained our initial models and our extended parameter testing models with a selection of data from the HT dataset, using a 70/30 train/test split and a random seed of 21. We created models from 2, 3, and 4 training epochs for each of our selected transformers, with learning rates of 2e-5, 3e-5, and 5e-5. We used batch sizes of 8 for all permutations of models, epochs, and learning rates. The relatively smaller batch size (vs. 16 or 32) was used because some models gave rise to buffer overrun errors with the Nvidia Geforce RTX 2080 GPU used in experimentation. Moreover, the batch size selection follows Masters and Luschi (2018), who found "...that using small batch sizes for training provides benefits both in terms of range of learning rates that provide stable convergence and achieved test performance for a given number of epochs."

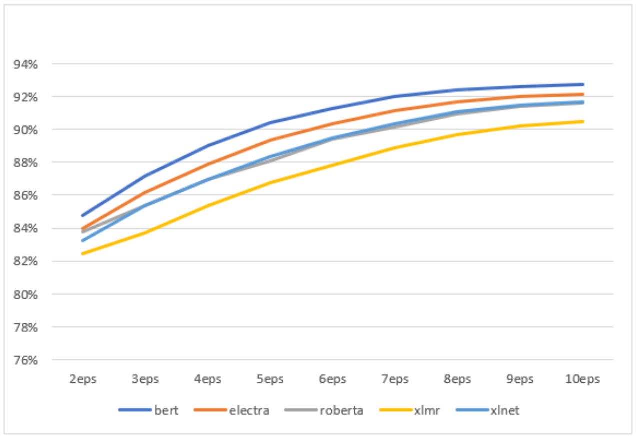

Figure 1: Accuracy increase curve flattens at 9-10 epochs.

Extended Parameter Testing We extended parameter testing by adding additional training epochs until increases in the plotted accuracy curve became relatively flat. Thus, we trained additional models for each transformer up to 10 epochs, across all learning rates. As a long-range comparison point, we also trained one BERT model to 50 epochs.

Results and Discussion

For comparative analysis, we selected accuracy as our primary metric. We considered previous research (Salman and Liu 2019; Sedik and others 2020; Safder et al. 2020) comparing validation loss and accuracy to assess how well a model generalizes and avoid overfitting, as well as others who posit that sufficiently large datasets generate models wherein the flat part of a power-law learning curve describes a region of irreducible error (Hestness et al. 2017). To assess how well our models generalize, We performed 10-fold cross validation with a 70/30 split to provide robust sampling across the entire dataset and determined that the accuracy scores were stable across folds, with maximum deviations from the average generally within $\pm 0.05\%$.

We performed 10-fold cross-validation testing across the models we trained with a grid search using the BERT-recommended parameters. Results are shown in Table 2. In our domain of fine-grained emotion detection from social media texts, we found that the higher learning rate, 5e-5, yielded more accurate results than the smaller 2e-5 and 3e-5 rates. This trend held across all transformer models, with BERT yielding the most accurate results after 4 epochs at an

Model	**2eps**	**3eps**	**4eps**	**5eps**	**6eps**	**7eps**	**8eps**	**9eps**	**10eps**
bert	84.75%	87.14%	89.03%	90.39%	91.30%	92.00%	92.38%	92.63%	92.72%
electra	83.95%	86.19%	87.87%	89.32%	90.37%	91.16%	91.69%	92.02%	92.15%
roberta	83.78%	85.37%	86.95%	88.10%	89.43%	90.16%	90.93%	91.44%	91.62%
xlmr	82.42%	83.73%	85.35%	86.77%	87.80%	88.89%	89.70%	90.23%	90.45%
xlnet	83.26%	85.34%	86.97%	88.34%	89.46%	90.35%	91.06%	91.50%	91.66%

Table 2: Accuracy per epoch by transformer model at 5e-5 learning rate.

average accuracy of 89.03%. We also determined that the increase in accuracy was most prevalent in the BERT models, with a difference of approximately 2% increase for BERT and ELECTRA and 1.0-1.5% for other models.

We also noted that accuracy continued to show substantial increases across training epochs. For example, BERT at the 5e-5 learning rate improved nearly 2% in accuracy from the 3rd to 4th training epoch. Given the rate of increase was still decreasing relatively slowly between the 3rd and 4th epochs, we performed additional grid search testing to determine where the rate of increase was low enough that the increase in accuracy was prohibitively expensive when compared to the increased training time. We found that the rate of accuracy increase across all models and epochs tended to flatten significantly at about the 9-10 epoch range. See Figure 1 for comparison.

As shown, when we extended our number of fine-tuning epochs to 10, accuracy increased 0.1% between the 9th and 10th epoch to 91.98%, whereas the accuracy increased nearly 3 times as much between the 8th and 9th epoch, with an increase of 0.28%. To assess how much accuracy increased with additional training epochs, we trained a 50 epoch model at the 3e-5 rate. Even when we extended our model to 50 epochs, the accuracy only improved to 93.41%, for an increase of 1.41% when compared to the 3e-5 model at 10 epochs. Given that each additional training epoch for our BERT models required approximately an additional hour of computational time (for our setup), an average accuracy increase per epoch of 0.036% (assuming an even distribution instead of diminishing returns) could become quite expensive in terms of computational time, especially when BERT and ELECTRA were the least expensive to train, and training other models usually required 1.5-2 hours per epoch.

Conclusions and Future Work

Based on our experiments and analysis, we note the following recommendations for training transformers to detect fine-grained emotions in social media text.

- Consider BERT as a preferred baseline model for fine-grained emotion detection
- Batch size 8 performs well and is likely to be efficient enough for many tasks
- Fine-tuning for 9 epochs provides a reasonable balance for accuracy (over 90%) and efficiency, even on single GPU machines
- Target a learning rate of 5e-5, as smaller learning rates seem to provide lower accuracy
- Setting recommendations are valid for BERT, RoBERTa, ELECTRA, XLM-R, and XLNet, as we observed similar trends in accuracy gains across all models

Our focus was on fine-grained emotion detection beyond positive / negative / neutral, and substantive datasets are fairly limited. Thus the analysis here is limited to results from the HT dataset. In this work, our focus was on accuracy versus validation loss, whereas future work includes examination of other metrics (precision, recall, f-measure) and further validation to confirm general recommendations for applying transformers, specifically within the context of emotion detection in short-texts. As part of ongoing research, we are applying this analysis to investigate ensemble predictions from various approaches to fine-grained emotion detection. Finally, we observed variations in previous work on level of detail across parameterization and fine-tuning, and we plan to continue work toward enhancing community standards on threshold criteria in applying transformer approaches.

Copyright

References

Abreu, S. 2019. Automated architecture design for deep neural networks. *CoRR* abs/1908.10714.

Bergstra, J., and Bengio, Y. 2012. Random search for hyper-parameter optimization. *Journal of machine learning research* 13(2).

Callan, J.; Hoy, M.; Yoo, C.; and Zhao, L. 2009. Clueweb09 data set.

Cho, K., et al. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation.

Chollet, F., et al. 2015. Keras. https://github.com/fchollet/keras.

Clark, K.; Luong, M.-T.; Le, Q. V.; and Manning, C. D. 2020. Electra: Pre-training text encoders as discriminators rather than generators.

Conneau, A., et al. 2018. Xnli: Evaluating cross-lingual sentence representations. *arXiv preprint arXiv:1809.05053*.

Conneau, A.; Khandelwal, K.; Goyal, N.; Chaudhary, V.; Wenzek, G.; Guzmán, F.; Grave, E.; Ott, M.; Zettlemoyer, L.; and Stoyanov, V. 2020. Unsupervised cross-lingual representation learning at scale.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Elshawi, R.; Maher, M.; and Sakr, S. 2019. Automated machine learning: State-of-the-art and open challenges.

Ghosh, S.; Vinyals, O.; Strope, B.; Roy, S.; Dean, T.; and Heck, L. 2016. Contextual lstm (clstm) models for large scale nlp tasks.

Harris, C. R., et al. 2020. Array programming with NumPy. *Nature* 585(7825):357–362.

Hassan, A.; Abbasi, A.; and Zeng, D. 2013. Twitter sentiment analysis: A bootstrap ensemble framework. In *2013 international conference on social computing*. IEEE.

Hestness, J.; Narang, S.; Ardalani, N.; Diamos, G.; Jun, H.; Kianinejad, H.; Patwary, M.; Ali, M.; Yang, Y.; and Zhou, Y. 2017. Deep learning scaling is predictable, empirically. *arXiv preprint arXiv:1712.00409*.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural computation* 9(8):1735–1780.

Johnson, R., and Zhang, T. 2015. Effective use of word order for text categorization with convolutional neural networks.

Johnson, R., and Zhang, T. 2017. Deep pyramid convolutional neural networks for text categorization. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 562–570.

Kim, Y. 2014. Convolutional neural networks for sentence classification. *CoRR* abs/1408.5882.

Koch, P.; Wujek, B.; Golovidov, O.; and Gardner, S. 2017. Automated hyperparameter tuning for effective machine learning. In *proceedings of the SAS Global Forum 2017 Conference*, 1–23. SAS Institute Inc. Cary, NC.

Lample, G., and Conneau, A. 2019. Cross-lingual language model pretraining.

LeCun, Y.; Haffner, P.; Bottou, L.; and Bengio, Y. 1999. Object recognition with gradient-based learning. In *Shape, contour and grouping in computer vision*. Springer.

Liu, Y., et al. 2019. Roberta: A robustly optimized bert pretraining approach.

Liu, J.; Chang, W.-C.; Wu, Y.; and Yang, Y. 2017. Deep learning for extreme multi-label text classification. In *Proceedings of the 40th International ACM SIGIR Conference*.

Loper, E., and Bird, S. 2002. Nltk: The natural language toolkit. In *In Proceedings of the ACL Workshop on Effective Tools and Methodologies for Teaching Natural Language Processing and Computational Linguistics*.

Mackenzie, J., et al. 2020. Cc-news-en: A large english news corpus. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 3077–3084.

Masters, D., and Luschi, C. 2018. Revisiting small batch training for deep neural networks. *CoRR* abs/1804.07612.

Murray, K.; Kinnison, J.; Nguyen, T. Q.; Scheirer, W. J.; and Chiang, D. 2019. Auto-sizing the transformer network: Improving speed, efficiency, and performance for low-resource machine translation. *CoRR* abs/1910.06717.

Nagel, S. 2016. Common crawl. Data retrieved from http://https://commoncrawl.org/2016/10/news-dataset-available/.

pandas development team, T. 2020. pandas-dev/pandas: Pandas.

Parker, R.; Graff, D.; Kong, J.; Chen, K.; and Maeda, K. 2011. English gigaword fifth edition, linguistic data consortium. *Google Scholar*.

Pedregosa, F., et al. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Peterson, J.; Meylan, S.; and Bourgin, D. 2019. Open clone of openai's unreleased webtext dataset scraper used to train gpt-2. https://github.com/jcpeterson/openwebtext.

Rajapakse, T. C. 2019. Simple transformers. https://github.com/ThilinaRajapakse/simpletransformers.

Ranganathan, J., and Tzacheva, A. 2019. Emotion mining in social media data. *Procedia Computer Science* 159:58–66.

Safder, I.; Hassan, S.-U.; Visvizi, A.; Noraset, T.; Nawaz, R.; and Tuarob, S. 2020. Deep learning-based extraction of algorithmic metadata in full-text scholarly documents. *Information Processing Management* 57(6):102269.

Salman, S., and Liu, X. 2019. Overfitting mechanism and avoidance in deep neural networks. *arXiv preprint arXiv:1901.06566*.

Schuster, M., and Paliwal, K. K. 1997. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing* 45(11):2673–2681.

Sedik, A., et al. 2020. Deploying machine and deep learning models for efficient data-augmented detection of covid-19 infections. *Viruses* 12(7):769.

Trinh, T. H., and Le, Q. V. 2019. A simple method for commonsense reasoning.

Wang, W.; Chen, L.; Thirunarayan, K.; and Sheth, A. P. 2012. Harnessing twitter" big data" for automatic emotion identification. In *2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing*, 587–592. IEEE.

Wang, A.; Singh, A.; Michael, J.; Hill, F.; Levy, O.; and Bowman, S. R. 2019. Glue: A multi-task benchmark and analysis platform for natural language understanding.

Wolf, T., et al. 2020. Huggingface's transformers: State-of-the-art natural language processing.

Wu, Y., et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation.

Yang, L., and Shami, A. 2020. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing* 415:295–316.

Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.; and Le, Q. V. 2020. Xlnet: Generalized autoregressive pretraining for language understanding.

Yue, L., et al. 2019. A survey of sentiment analysis in social media. *Knowledge and Information Systems* 60(2):617–663.

Zhu, Y., et al. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, 19–27.