An evaluation of NLP research examining the robustness and performance of BERT, a multi-layer bidirectional transformer, for text-classification tasks

Brian Ferrell CMSC 516

October 2020

1 Introduction

The aim of this paper is to bring different perspectives on how well BERT performs on text classification tasks within different domains and digging into how to improve BERT with some creative methods. For my own text classification problem, I wanted to see how BERT stacked up against various deep learning methods, therefore, this is a more thematic approach to this review. This evaluation revolves around how BERT does at different text classification tasks and how there are ways of making it even better.

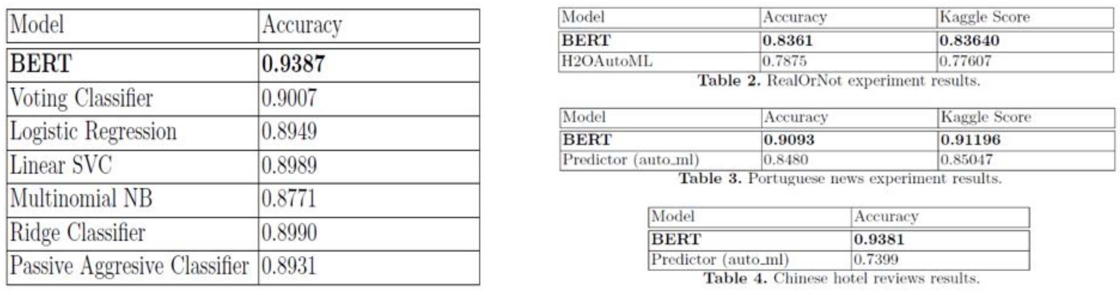
This paper can be broken down into two sections:

- Comparing BERT to classical machine learning methods and other powerful neural networks
- Exposing BERT's vulnerabilities and improving it by using adversarial attacks, training within specific domains, examining the layers within BERT, and parameter reduction techniques (i.e. AlBERT).

2 Comparing BERT on text classification tasks

BERT is widely used when comparing against traditional machine learning and deep learning models. It is likely that BERT will always play a big part in how we model language, and benchmark against even more powerful transformers that are out there, considering the amount of BERT papers there are within this field. This section is meant to show results from literature where authors attempt to use the language model, and fine-tune it to their classification tasks using various datasets. You will find that BERT can be superior whether it is with large datasets, small datasets, short text-lengths, longer text-lengths, or even datasets of different languages. I believe the author's intent with these comparisons is to see if BERT should be the default when it comes to certain NLP tasks, and to examine BERT's dominance over the traditional ways of handling NLP problems. The perception of BERT is that you need large amounts of computing power and data, but that is just not the case in the results shown from the literature.

BERT against machine learning. Gonzalez-Carvajal et al.[1] ran four comparisons on four datasets, using the default pre-trained BERT and several well-known machine learning models. The first experiment uses the famously known IMDB dataset to conduct a sentiment analysis. The second experiment compares BERT to H2OAutoML for a binary text classification task. This dataset (RealorNot tweets) contains tweets about real disasters and tweets that are not about real disasters (ex. use of metaphors). Finally, since BERT continues to outperform their past experiments with the English language, they use a Portuguese news dataset classified into 9 classes for the third experiment and a Chinese hotel reviews dataset for the fourth. Both compared BERT to the AutoML module. They use different ways of splitting the data into training and test sets which I assume is because each dataset had different amounts of data. Below captures the results of the four experiments in the order mentioned above:



Model	Accuracy
BERT	**0.9387**
Voting Classifier	0.9007
Logistic Regression	0.8949
Linear SVC	0.8989
Multinomial NB	0.8771
Ridge Classifier	0.8990
Passive Aggresive Classifier	0.8931

Model	Accuracy	Kaggle Score
BERT	**0.8361**	**0.83640**
H2OAutoML	0.7875	0.77607

Table 2. RealOrNot experiment results.

Model	Accuracy	Kaggle Score
BERT	**0.9093**	**0.91196**
Predictor (auto_ml)	0.8480	0.85047

Table 3. Portuguese news experiment results.

Model	Accuracy
BERT	**0.9381**
Predictor (auto_ml)	0.7399

Table 4. Chinese hotel reviews results.

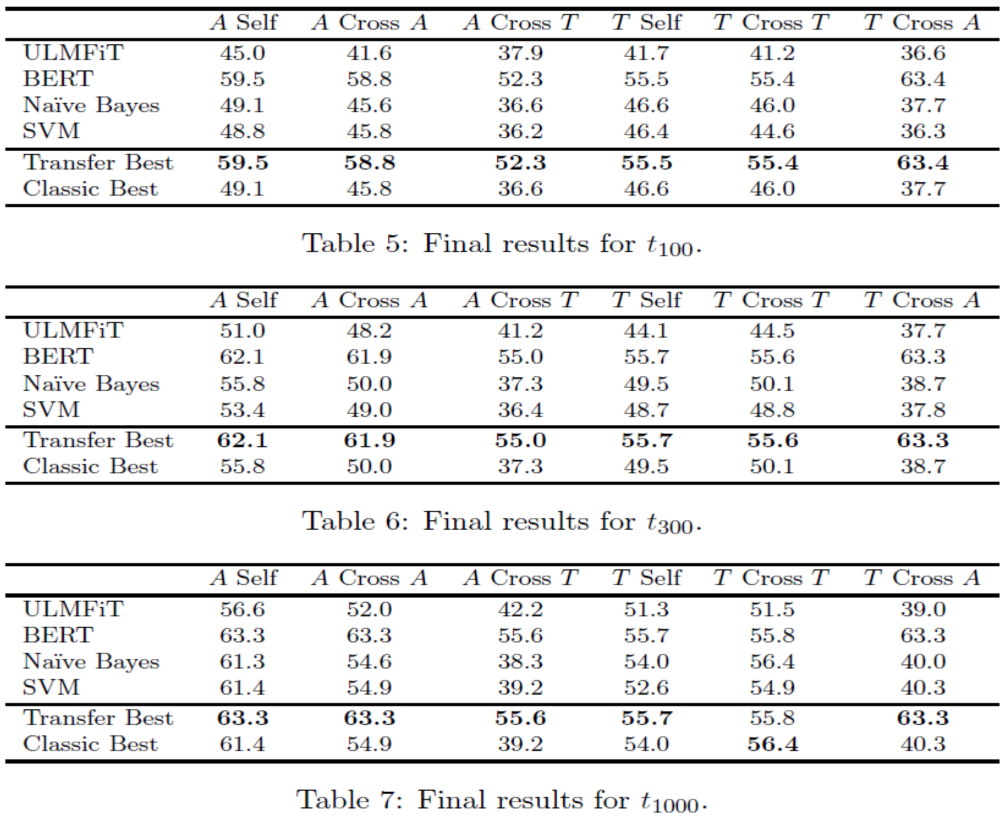
Figure 1: Results from four experiments[1]

In some of the tasks, BERT did not have an enormous lead in terms of the accuracy, so I think it comes down to training time and the effort that was put into the TF-IDF models. This also is just a basic BERT model, and the authors do not mention anything about what learning rate they used for BERT or epochs. They reference the original paper of BERT when talking about parameter tuning, so maybe they went with what the authors of that paper recommend. Another thing to add is the last experiment which uses the Chinese hotel reviews, had the smallest dataset and BERT outperforms the AutoML model (best was the gradient boosting classifier). The only thing missing in this paper was a breakdown of how much data was in each dataset and any limitations of BERT that the authors might have discovered.

In conclusion, the authors admitted that BERT turned out to be far less complicated than implementing the traditional methods. Additionally, it will be interesting to see how the authors improve these tasks in the future.

Low-Shot Classification. The authors Peter Usherwodand and Steven Smit demonstrate in their paper[2] that BERT can be just as amazing on datasets where there are only 100-1000 examples per class. They compare BERT and ULMFiT to a variety of classical methods just like the ones mentioned before, only they address the lack of quantitative studies on smaller datasets. They focus their efforts on five datasets using sentiment-based classification tasks divided into two categories: Amazon reviews and Twitter. For the Amazon reviews they use three datasets containing movie reviews, books reviews, and health and personal-care product reviews, and the other two datasets from Twitter are under subtask a and ce from SemEval2017. Some of the ways they test how BERT, and other models, perform is they use Amazon trained classifiers on other Amazon datasets, and Amazon trained classifiers on Twitter datasets (and vice versa). This is to see how the models perform across multiple domains.

The figure below shows the results from the datasets:



	A Self	A Cross A	A Cross T	T Self	T Cross T	T Cross A
ULMFiT	45.0	41.6	37.9	41.7	41.2	36.6
BERT	59.5	58.8	52.3	55.5	55.4	63.4
Naïve Bayes	49.1	45.6	36.6	46.6	46.0	37.7
SVM	48.8	45.8	36.2	46.4	44.6	36.3
Transfer Best	**59.5**	**58.8**	**52.3**	**55.5**	**55.4**	**63.4**
Classic Best	49.1	45.8	36.6	46.6	46.0	37.7

Table 5: Final results for t_{100}.

	A Self	A Cross A	A Cross T	T Self	T Cross T	T Cross A
ULMFiT	51.0	48.2	41.2	44.1	44.5	37.7
BERT	62.1	61.9	55.0	55.7	55.6	63.3
Naïve Bayes	55.8	50.0	37.3	49.5	50.1	38.7
SVM	53.4	49.0	36.4	48.7	48.8	37.8
Transfer Best	**62.1**	**61.9**	**55.0**	**55.7**	**55.6**	**63.3**
Classic Best	55.8	50.0	37.3	49.5	50.1	38.7

Table 6: Final results for t_{300}.

	A Self	A Cross A	A Cross T	T Self	T Cross T	T Cross A
ULMFiT	56.6	52.0	42.2	51.3	51.5	39.0
BERT	63.3	63.3	55.6	55.7	55.8	63.3
Naïve Bayes	61.3	54.6	38.3	54.0	56.4	40.0
SVM	61.4	54.9	39.2	52.6	54.9	40.3
Transfer Best	**63.3**	**63.3**	**55.6**	**55.7**	55.8	**63.3**
Classic Best	61.4	54.9	39.2	54.0	**56.4**	40.3

Table 7: Final results for t_{1000}.

Figure 2: Average accuracy's from the different domains[2]

As you can see, BERT outperforms the machine learning models as well as UMLFiT. The overall results for each of the models are not spectacular, thus more data would be needed; however, the authors could have used cross-validation if they had the resources for it. Authors point out there is less time spent on hyperparameter tuning when it comes to BERT, due to just changing the epochs/learning rates and nothing else.

Approaches for Alzheimer's Disease Detection. Balagopalan et al.[3] compare BERT to feature engineered machine learning models for detecting Alzheimer's Disease in speech transcripts from the ADDReSS Dataset challenge. Since humans find the features within machine learning models, it can be time-consuming and error prone, also requiring expert domain knowledge, which is why they want to test how BERT performs. This is another challenge where the dataset is quite small, consisting of 156 speech samples split 50/50 for either non-Alzheimer's or are Alzheimer's participants, where they are tasked with describing a picture. All the models (including BERT) are trained using 10-fold cross validation, and another cross-validation strategy (not for BERT, because of memory constraints) called leave-one-subject-out CV.

Below shows the results from the training sets, and the held-out test set *(Table four has the LOSO-CV which the authors mention they did not use for BERT due to it being more computationally expensive):*

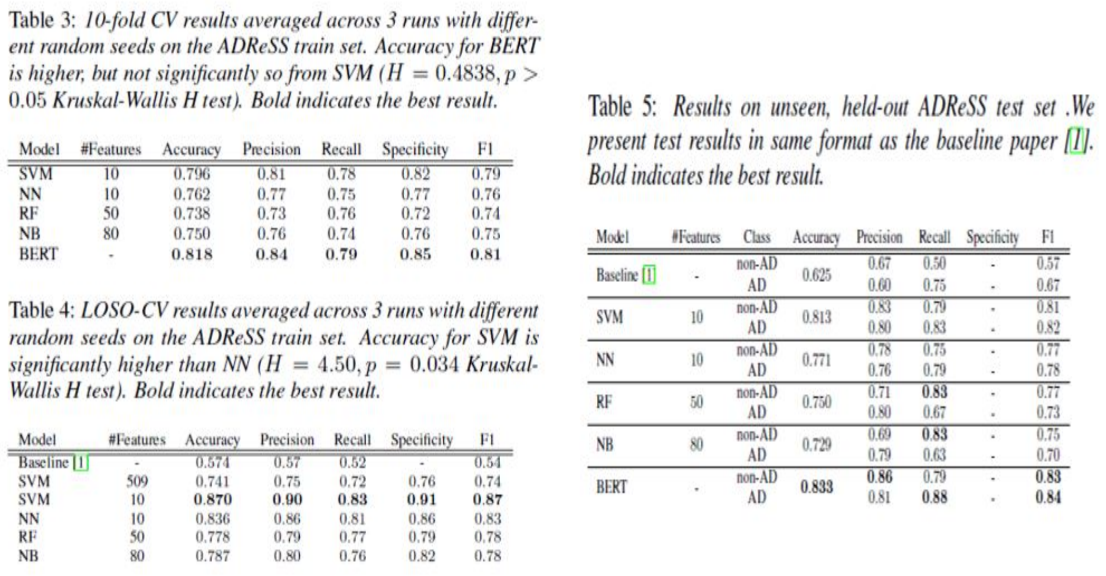


Table 3: *10-fold CV results averaged across 3 runs with different random seeds on the ADReSS train set. Accuracy for BERT is higher, but not significantly so from SVM ($H = 0.4838, p > 0.05$ Kruskal-Wallis H test). Bold indicates the best result.*

Model	#Features	Accuracy	Precision	Recall	Specificity	F1
SVM	10	0.796	0.81	0.78	0.82	0.79
NN	10	0.762	0.77	0.75	0.77	0.76
RF	50	0.738	0.73	0.76	0.72	0.74
NB	80	0.750	0.76	0.74	0.76	0.75
BERT	-	**0.818**	**0.84**	**0.79**	**0.85**	**0.81**

Table 4: *LOSO-CV results averaged across 3 runs with different random seeds on the ADReSS train set. Accuracy for SVM is significantly higher than NN ($H = 4.50, p = 0.034$ Kruskal-Wallis H test). Bold indicates the best result.*

Model	#Features	Accuracy	Precision	Recall	Specificity	F1
Baseline [1]	-	0.574	0.57	0.52	-	0.54
SVM	509	0.741	0.75	0.72	0.76	0.74
SVM	10	**0.870**	**0.90**	**0.83**	**0.91**	**0.87**
NN	10	0.836	0.86	0.81	0.86	0.83
RF	50	0.778	0.79	0.77	0.79	0.78
NB	80	0.787	0.80	0.76	0.82	0.78

Table 5: *Results on unseen, held-out ADReSS test set .We present test results in same format as the baseline paper [1]. Bold indicates the best result.*

Model	#Features	Class	Accuracy	Precision	Recall	Specificity	F1
Baseline [1]	-	non-AD	0.625	0.67	0.50	-	0.57
		AD		0.60	0.75	-	0.67
SVM	10	non-AD	0.813	0.83	0.79	-	0.81
		AD		0.80	0.83	-	0.82
NN	10	non-AD	0.771	0.78	0.75	-	0.77
		AD		0.76	0.79	-	0.78
RF	50	non-AD	0.750	0.71	**0.83**	-	0.77
		AD		0.80	0.67	-	0.73
NB	80	non-AD	0.729	0.69	**0.83**	-	0.75
		AD		0.79	0.63	-	0.70
BERT	-	non-AD	**0.833**	**0.86**	0.79	-	**0.83**
		AD		0.81	**0.88**	-	**0.84**

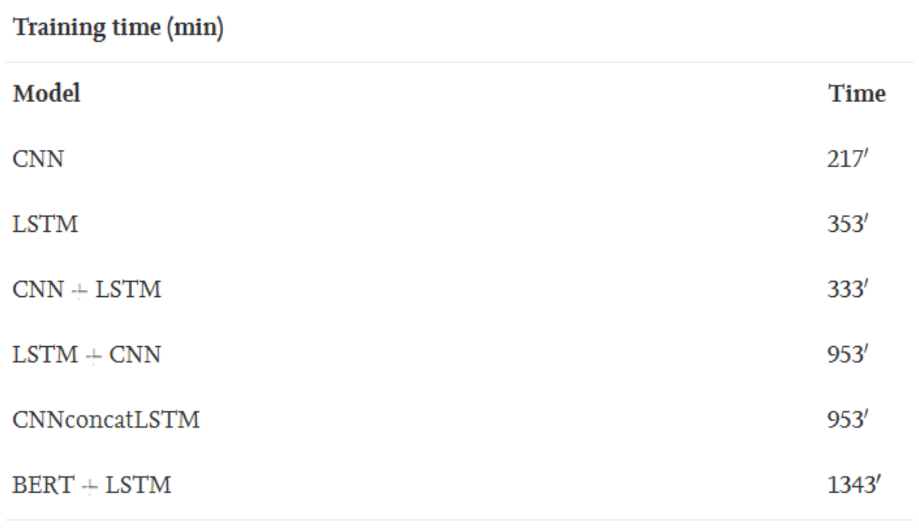
Figure 3: Results from experiments[3]

Once again, BERT outperforms all the models. Had they tried different amounts of folds or used BERT without cross-validation would the results be any different. To conclude, the authors wish to explore ways of combining representations from BERT with those hand-crafted features, which I think would be cool to see if it boosts performance in any way.

BERT being used across different cultures. A team of researchers led by Elizaveta Zinovyeva use BERT for malicious online behavior detection[4]. This is an interesting paper to view how BERT is being used in places outside of America with different sets of values and goals for protecting people's social welfare. I will not show the results from their classification task because they have an exhaustive list of models being compared to BERT and DistilBERT such as GRU, Bi-LSTM/GRU, transformers, attention models, and more using 5-fold cross-validation. I will say that DistilBERT and BERT do better on the number of binary classification tasks they had for each dataset predicting whether something was malicious or not. This task is very unique because how do we draw the line between what is malicious intent versus what is sarcastic. Even if we can differentiate between what is considered malicious (which I am sure we can), what do we do with that kind of power when it comes to people's opinions. It also seems that having the perfect dataset for their task doesn't seem to exist, as they point out the unintended biases in their paper. These biases occur from different ways the datasets are retrieved, and it leads to discriminating against people or groups differently in unfair ways. For example, just mentioning gender, religion, or sexual orientation apparently can have higher "malicious scores" even if the comment is not intended to be hateful. Nonetheless, it is interesting to see how culture affects the uses of BERT and to still see it perform well on this "bad behavior" detection even in varying datasets.

Comparing different models for sentiment analysis of drug reviews. Colón-Ruiz et al.[5] present comparisons on different advanced NLP models on sentiment analysis tasks for drug reviews. The models used for this task are the following: LSTM, CNN, SVM, Hybrid models, and BERT+LSTM. The dataset used is taken from Drugs.com, which comprises of drug reviews with their corresponding score (reflects patient's degree of satisfaction with the drug). They do this task in two ways: Classify ratings based on a ten-point scale, and then reduce the ten-point scale to a three-point scale by combining the ratings into three levels of polarity(negative, positive, and neutral). For models other than BERT they use different approaches for their word embeddings: one was trained on PubMed, PMC, and Wikipedia, and the other was trained on tweets about drugs (Colon-Ruiz, 2020). Refer to the paper for the results of both embedding styles and different datasets, because they too had a large list. To break it down though, the LSTM followed by a CNN had better results on the 10-class dataset, and BERT+LSTM had best results on the 3-class one. Overall, the 3-point scale task improves performance for obvious reasons.

(Results of the models containing F1 scores and confusion matrices can be found at this link:https://www.sciencedirect.com/science/article/pii/S1532046420301672 since they are hard to copy/paste in a readable way)
They also break down the time it took to train each model:



Training time (min)

Model	Time
CNN	217′
LSTM	353′
CNN + LSTM	333′
LSTM + CNN	953′
CNNconcatLSTM	953′
BERT + LSTM	1343′

Figure 4: Training time for experiments[5]

They ran the BERT+LSTM model for 200 epochs, and had a learning rate of .001, which I would like to know why they steered away from the BERT author's recommendation of 3-5 epochs and a much smaller learning rate (between 5 · 10 -5 and 1 · 10 -5). They also could have used another BERT that relates better to their domain like ClinicalBERT or BioBERT, and also I would've liked to have seen BERT being used by itself without the LSTM attached just to see if it was even necessary to do so. Overall, BERT still does well, despite how much time it took them to train it due to the LSTM being added on.

BERT models applied to radiological text classification. Ranti et al.[6] leverage the use of a medical domain corpus combined with a BERT model on a radiological multi-label text classification problem. The challenges, when it comes to electronic health records, is that they are hard to read, hard to classify, hard to do research on, and can be inaccessible sources due to high proportions of the data being unstructured (Ranti, 2020). To help in the automation of classifying and segmenting the textual electronic health report data, the authors randomly sampled 1,977 CT scanner reports (out of 97K) from the hospital's database system, which were hand-labeled by a team of physicians and neurosurgical/radiology residents. The hand-classified labels are an independent binary multi-label dataset, which means there are multiple labels that can correspond to a single CT report. There can be a 0 or a 1 in front of these 13 labels: normal, hemorrhage, stroke/infarction/ischemia (infarction, venous thrombosis), vascular abnormality (aneurysm, arteriovenous malformation), chronic small vessel disease, periventricular white matter changes, ventricular abnormalities (i.e. hydrocephalus), atrophy (brain), bone abnormality, bony sinus disease, foreign objects, maxillary sinus disease, and carotid siphon calcification. They compare the regular BERT to BioBERT (Ranti, 2020), and a few other models on precision and recall (also included is the ROC score per label). Their label counts and frequencies can be found in Table 1 of the paper, which breaks down the test, evaluation, and training sets for all the 13 labels.

Below are the results of all the models:

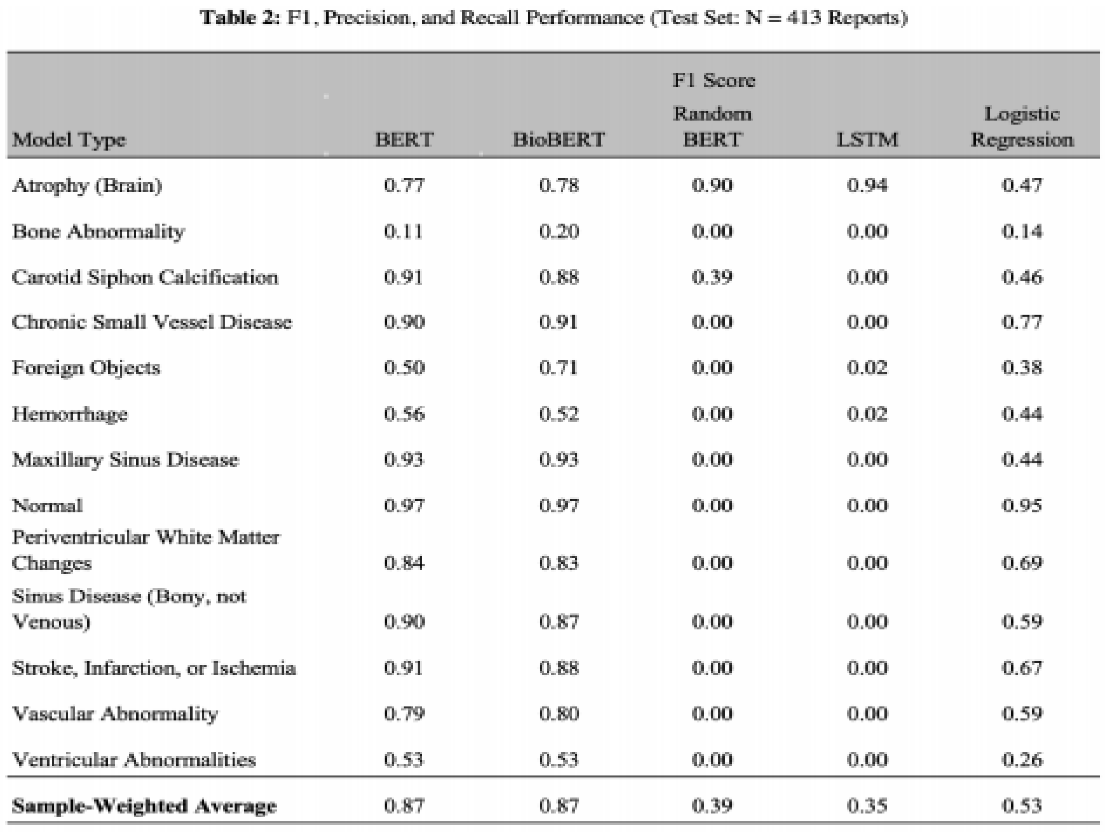


Table 2: F1, Precision, and Recall Performance (Test Set: N = 413 Reports)

	F1 Score				
Model Type	BERT	BioBERT	Random BERT	LSTM	Logistic Regression
Atrophy (Brain)	0.77	0.78	0.90	0.94	0.47
Bone Abnormality	0.11	0.20	0.00	0.00	0.14
Carotid Siphon Calcification	0.91	0.88	0.39	0.00	0.46
Chronic Small Vessel Disease	0.90	0.91	0.00	0.00	0.77
Foreign Objects	0.50	0.71	0.00	0.02	0.38
Hemorrhage	0.56	0.52	0.00	0.02	0.44
Maxillary Sinus Disease	0.93	0.93	0.00	0.00	0.44
Normal	0.97	0.97	0.00	0.00	0.95
Periventricular White Matter Changes	0.84	0.83	0.00	0.00	0.69
Sinus Disease (Bony, not Venous)	0.90	0.87	0.00	0.00	0.59
Stroke, Infarction, or Ischemia	0.91	0.88	0.00	0.00	0.67
Vascular Abnormality	0.79	0.80	0.00	0.00	0.59
Ventricular Abnormalities	0.53	0.53	0.00	0.00	0.26
Sample-Weighted Average	0.87	0.87	0.39	0.35	0.53

Figure 5: Results for experiments[6]

They broke the results down by showing the accuracy per label and sample-weighted average, then compared the results using logistic regression, LSTM, and three different BERT models. There was no significant difference between BERT and BioBERT (Ranti, 2020), but the Random BERT model (parameters were randomly assigned) did even worse than the logistic regression model. This study demonstrates once again the power of transfer learning and BERT's robustness, but with the twist of including a combined medical and general domain corpus.

Takeaways. There are several things you can take from this section. One, condensing classes to fit a more general spectrum might be best. Two, the use of cross-validation should not be ignored especially for small datasets. Three, BERT performs very well across multiple domains. Four, just try things out and see what works, as there is no guarantee that BERT or a specific domain of BERT is best for your task.

3 Can BERT get any better?

In this section we look at how to improve BERT through several ways. Sometimes BERT gets a lot of praise for what it can do (i.e. what I just did for the first 6 pages). Here we explore tactics to expose weaknesses in BERT, view the layers within BERT, and see how BERT can be faster and more scalable by reducing the amount of parameters it has.

Review of Literature

Adversarial Attacks. At first, I didn't understand why purposefully changing sentences to cause prediction accuracy to go down was so important, until I did some additional research into three papers that discuss strategies to make BERT misclassify data. The reason behind these attacks is to expose BERT's weaknesses and help understand what the model gets wrong with just a few simple changes to the text. These simple changes would leave you to believe that BERT would still get the correct answer, because it is just synonym replacement or inserting words that are still contextually correct to the classification, but as a result it does not.

Here is an example of what I mean from the paper introducing TextFooler (Jin, D et al)[7]:

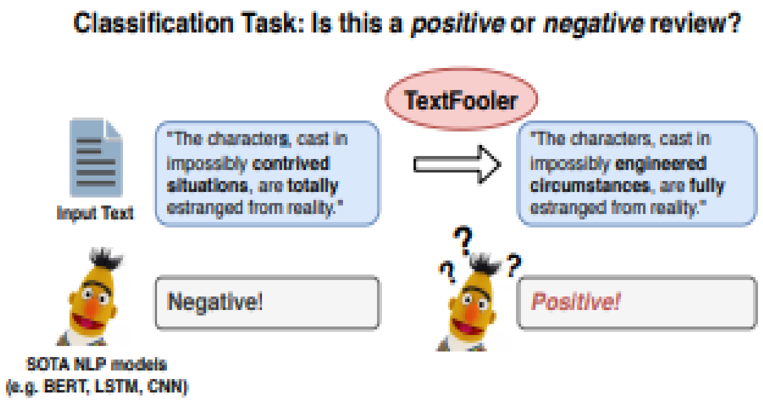
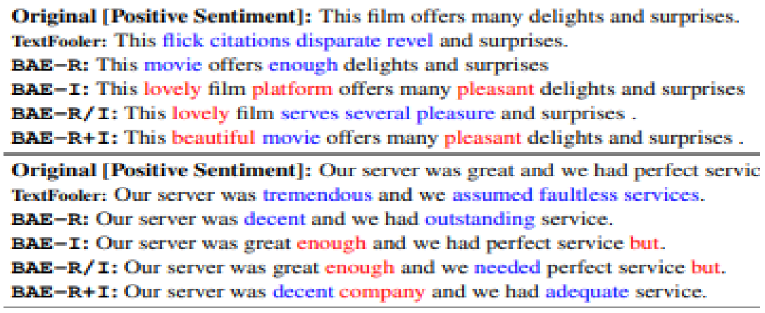

Classification Task: Is this a positive or negative review?
TextFooler
Input Text
"The characters, cast in impossibly contrived situations, are totally estranged from reality."
"The characters, cast in impossibly engineered circumstances, are fully estranged from reality."
Negative!
Positive!
SOTA NLP models (e.g. BERT, LSTM, CNN)

Figure 6: Textfooler[7]

In this example, we see that the words are changed to words that are synonymous to each other, but BERT gets it wrong. This TextFooler algorithm is a great baseline for comparing different results, but the better ways of running adversarial attacks are through semantically similar words that best fit the overall context of the sentence. Two examples of algorithms that leverage the power of BERT to expose BERT would be BAE: BERT-based Adversarial Examples (Siddhant Garg et al.[8]) and BERT-ATTACK: Adversarial Attack Against BERT Using BERT (Linyang Li et al.[9]).

Below is an example of the four strategies that can be implemented from the BAE algorithm paper:



Original [Positive Sentiment]: This film offers many delights and surprises.
TextFooler: This flick citations disparate revel and surprises.
BAE-R: This movie offers enough delights and surprises
BAE-I: This lovely film platform offers many pleasant delights and surprises
BAE-R/I: This lovely film serves several pleasure and surprises .
BAE-R+I: This beautiful movie offers many pleasant delights and surprises .

Original [Positive Sentiment]: Our server was great and we had perfect servic
TextFooler: Our server was tremendous and we assumed faultless services.
BAE-R: Our server was decent and we had outstanding service.
BAE-I: Our server was great enough and we had perfect service but.
BAE-R/I: Our server was great enough and we needed perfect service but.
BAE-R+I: Our server was decent company and we had adequate service.

Figure 7: BAE[9]

Creating adversarial examples is becoming more popular, and I think when testing the robustness of BERT, it doesn't get more robust than picking BERT apart limb from limb just to see where it messes up on semantically similar sequences. There are even more examples that can be found here: https://github.com/QData/TextAttack, which is a Python framework that has 15+ attack strategies (like the ones listed above) as well as data augmentation techniques and model training.

Other improving BERT strategies. There are a wide range of solutions to make BERT perform better. For instance, Sun, C et al.[10] attempt to fine-tune BERT on 8 different text classification datasets, examine learning-rates for specific layers, dealing with longer sequences, layer selection, low-shot learning problems, and more. According to the authors, different layers of BERT capture different levels of semantic and syntactic information, and they examine which of those layers is best for final prediction (Sun, 2019). They conclude that the top layer of BERT is more useful when it comes to text classification, further pre-training to fit your specific domain can significantly boost performance, and BERT shows significant improvement on small datasets.

Speaking of pre-training to fit a specific domain, the author Emily Alsentzer et al.[11] improve BERT by creating BERT models that are geared towards clinical text. They also prove that domain-specific pre-trainings improve performance for classification tasks. The purpose of this task was to further downstream BERT with the clinical data that can be used for even more specific classification problems because BERT, which is pre-trained on BooksCorpus and Wikipedia, can be too broad for specific problems.

The last thing to mention when it comes to improving BERT, would be what Lan, Z et al.[12] from Google Research and Toyota Technological Institute at Chicago created to make BERT more scalable. They introduced what is called "ALBERT", which stands for "a lite BERT", and it is meant to be a lot more efficient due to its parameters being reduced significantly. The model's advantage is the reduction of parameters, which in some cases is not only faster than BERT but has better benchmark accuracy. The authors introduce two parameter reduction techniques to lower GPU/TPU memory usage while increasing the training speed of BERT. These parameter reduction strategies alone actually hurt accuracy, but they are meant to be scaled up to what is needed to improve.

Takeaways. I hope you see that these strategies can be very beneficial in improving the overall robustness of BERT, and maybe it gives the idea of trying out ALBERT to see if it works for your task. To conclude this section, BERT does a better job when pre-trained to a specific domain, exposing its vulnerabilities can provide insight in how to expand your corpus to fit all possible samples in your dataset, and ALBERT can be used and scaled up when necessary for whatever task you are working on.

4 Conclusion

I wanted to find literature between 2019-2020, so that this review had the most relevant aspects of the power of BERT and how to improve it. Of course, everyone's tasks look differently, whether it is the size of data, the type of data, the length of sequences, or even the domain of what the data is capturing; however, I gave a honest review of several tasks BERT can perform on, and how to even make it better. I think areas for future research will be in the pre-training aspect of BERT, with trying to fit specific domains for BERT, since that really is how to significantly improve results on text classification tasks of all dataset sizes.

The important aspects of this literature review are as follows:

- BERT can be fine-tuned towards text classification tasks and perform very well against other machine learning/deep learning models without having expert domain knowledge
- Exposing BERT's vulnerabilities is important and is becoming a big research area
- Pre-training BERT to fit a specific domain might be your best option for text classification tasks
- Being a data scientist is hard but rewarding. You see so many unique uses for these powerful tools that are shaping the world we live in

References

[1] Santiago González-Carvajal and Eduardo C Garrido-Merchán. Comparing bert against traditional machine learning text classification. *arXiv preprint arXiv:2005.13012*, 2020.

[2] Peter Usherwood and Steven Smit. Low-shot classification: A comparison of classical and deep transfer machine learning approaches. *arXiv preprint arXiv:1907.07543*, 2019.

[3] Aparna Balagopalan, Benjamin Eyre, Frank Rudzicz, and Jekaterina Novikova. To bert or not to bert: Comparing speech and language-based approaches for alzheimer's disease detection. *arXiv preprint arXiv:2008.01551*, 2020.

[4] Elizaveta Zinovyeva, Wolfgang Karl Härdle, and Stefan Lessmann. Antisocial online behavior detection using deep learning. *Decision Support Systems*, page 113362, 2020.

[5] Cristóbal Colón-Ruiz and Isabel Segura-Bedmar. Comparing deep learning architectures for sentiment analysis on drug reviews. *Journal of Biomedical Informatics*, 110:103539, 2020.

[6] Daniel Ranti, Katie Hanss, Shan Zhao, Varun Arvind, Joseph Titano, Anthony Costa, and Eric Oermann. The utility of general domain transfer learning for medical language tasks. *arXiv preprint arXiv:2002.06670*, 2020.

[7] Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. *arXiv*, pages arXiv–1907, 2019.

[8] Siddhant Garg and Goutham Ramakrishnan. Bae: Bert-based adversarial examples for text classification. *arXiv preprint arXiv:2004.01970*, 2020.

[9] Linyang Li, Ruotian Ma, Qipeng Guo, Xiangyang Xue, and Xipeng Qiu. Bert-attack: Adversarial attack against bert using bert. *arXiv preprint arXiv:2004.09984*, 2020.

[10] Chi Sun, Xipeng Qiu, Yige Xu, and Xuanjing Huang. How to fine-tune bert for text classification? In *China National Conference on Chinese Computational Linguistics*, pages 194–206. Springer, 2019.

[11] Emily Alsentzer, John R Murphy, Willie Boag, Wei-Hung Weng, Di Jin, Tristan Naumann, and Matthew McDermott. Publicly available clinical bert embeddings. *arXiv preprint arXiv:1904.03323*, 2019.

[12] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. Albert: A lite bert for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.