FOOLING PRE-TRAINED LANGUAGE MODELS: AN EVOLUTIONARY APPROACH TO GENERATE WRONG SENTENCES WITH HIGH ACCEPTABILITY SCORE

Anonymous authors
Paper under double-blind review

ABSTRACT

Large pre-trained language representation models have recently collected numerous successes in language understanding. They obtained state-of-the-art results in many classical benchmark datasets, such as GLUE benchmark and SQuAD dataset, but do they really *understand* the language? In this paper we investigate two among the best pre-trained language models, BERT (Devlin et al. (2018)) and RoBERTa (Liu et al. (2019)), analysing their weaknesses by generating adversarial sentences in an evolutionary approach. Our goal is to discover if and why it is possible to fool these models, and how to face this issue. This adversarial attack is followed by a cross analysis, understanding robustness and generalization proprieties of models and fooling techniques. We find that BERT can be easily fooled, but an augmentation of the original dataset with adversarial samples is enough to make it learn how not to be fooled again. RoBERTa, instead, is more resistent to this approach even if it still have some weak spots.

1 INTRODUCTION

Recently pre-trained language models became popular due to their successes on many benchmark natural language processing (NLP) tasks. These models, such as BERT (Devlin et al. (2018)), needs an expensive pre-training phase, followed by a relatively cheap task specific finetuning. This training procedure allows us to use the same pre-trained model for more tasks, getting closer to a more general understanding of language through the construction of task independent embeddings of words in an unsupervised fashion. Besides BERT, many other different language models have been released recently, obtaining state-of-the-art results in multiple tasks, built as improvement or variations of existing models or pre-training procedures. ROBERTa (Liu et al. (2019)), for example, uses BERT architecture, improving the training procedure.

The proliferation of pre-trained language models was recently followed by works analyzing their behaviors (Jin et al. (2019); Tenney et al. (2019); Michel et al. (2019); Clark et al. (2019)), necessary to discover not only the weak spots and the possible improvements, but also trying to interpret these huge models (composed of hundreds of millions of parameters to train). However, much work has still to be done to obtain useful insights about the behavior these black boxes. Our work can be considered as another step to better understand pre-trained language models.

We aim to answer the following questions:
Q1: Can we fool a language model to classify wrongly sentences generated through an evolutionary algorithm?
Q2: Can we train a language model to recognize fooling sentences by augmenting the training dataset with previously generated fooling sentences?
Q3: Are the obtained fooling sentences universal or model specific? Can the same set fool more than one language model?

Our work is focused on fooling two language models, BERT and RoBERTa, in order to understand if the predictions obtained are reliable or if there are weak spots in the model that could be fatal if applied incautiously. Our algorithm is inspired by Nguyen et al. (2014), where the authors applied a similar evolutionary algorithm to fool ImageNet. Since pre-trained language models can be

considered the corresponding ImageNet for textual data, we wondered if those new models based on transformers have the same weaknesses as the ones based on convolutional neural networks.

The paper is structured as follows: section 2 collects the related work. In section 3, BERT and RoBERTa are briefly described, while in section 4 the datasets are presented. The fooling algorithm is shown in section 5 while section 6 the experiments designed. We conclude in section 7.

2 RELATED WORK

After the huge success of pre-trained language models, numerous papers about their analysis and applications have been published, trying to prove or demystify the goodness of those models.

Interesting analysis has been made trying to prune attention heads from BERT by Michel et al. (2019), observing that good results can be obtained also with smaller models.
Tenney et al. (2019) were able to discover steps of the traditional NLP pipeline inside the pre-trained BERT model, defining two measurements to quantify the effects, and observing qualitatively that the pipeline is dynamically adjusted to disambiguate information.
Clark et al. (2019) perform an exhaustive analysis of BERT's attentions heads, collecting similar patterns such as delimiter tokens, positional offsets, observing linguistic notions of syntax and co-reference.
An analysis on relation representations of BERT is made by Soares et al. (2019), designing a training setup called matching the blanks, relying solely on entity resolution annotations.
Regarding adversarial attacks on text data, Ebrahimi et al. (2017) propose a method based on swapping tokens, Li et al. (2018) develop TextBugger, an extremely effective method to trick Deep Learning-based Text Understanding (DLTU), while Niven & Kao (2019) construct an adversarial dataset exploring spurious statistical cues in the dataset, making state-of-the-art language models achieve random performances. Zhao et al. (2017), instead, use GANs to generate adversarial samples so to evaluate and analyze black-box classifiers.
Our work is inspired by Nguyen et al. (2014), where the authors apply fooling algorithms to convolutional neural networks in order to generate images that humans perceive almost as white noise, but they are predicted by ImageNet as belonging to a specific class with high probability.

3 PRETRAINED LANGUAGE MODELS

Even if our proposed approach can be applied to any sentence classifier, we selected two pretrained language models that have recently obtained state-of-the-art results to test their true effectiveness: BERT (Devlin et al. (2018)) and RoBERTa (Liu et al. (2019)). We chose those models also because the pre-trained weights and the hyperparameters used to finetune them are publicly available. However, weights finetuned for specific tasks are not, requiring us to perform it.

3.1 BERT

BERT (Bidirectional Encoder Representations from Transformers (Devlin et al. (2018))) is a deep state-of-the-art language representation model based of Transformers (Vaswani et al. (2017)) trained in an unsupervised way on 3.3 bilion tokens of English text. The model is designed to be finetuned to a specific tasks inserting an additional final layer, without substantial task-specific architecture modifications. We use the LARGE version of BERT, consisting in 24 layer, 1024 hidden dimension, 16 heads per layer, for a total of 340M parameters.

3.2 ROBERTA

RoBERTa (Liu et al. (2019)) is an improvement of BERT model through an accurate selection of hyper-parameters and pre-training techniques. The result is a state-of-the-art model with the same architecture as BERT, optimized by carefully investigating all the design choices made to train the model. The full set of hyperparameters used to both train and test the model is reported in the original paper, while the full code is available online.

4 DATASET

The General Language Understanding Evaluation (GLUE) benchmark (Wang et al. (2018)) is a collection of 9 English datasets for training, evaluating, and analyzing natural language understanding systems. The tasks cover a broad range of domains, data quantities, and difficulties, and are either single-sentence classification or sentence-pair classification tasks. The official website contains a submission server and a leaderboard, used in this paper to evaluate the performances of the models.

For the purpose of this paper, we selected one of the nine datasets in the GLUE benchmark: the Corpus of Linguistic Acceptability (CoLA). It consists of English acceptability judgments drawn from books and journal articles on linguistic theory. Each example is a sequence of words annotated with whether it is a grammatical English sentence. The evaluation metric used is Matthews correlation coefficient, as in the official paper (Warstadt et al. (2018)).

5 FOOLING ALGORITHM: EVOLUTIONARY GENERATIVE ALGORITHM

We propose the following algorithm designed to generate sentences that fool a language model is an evolutionary algorithm, inspired by Nguyen et al. (2014). These algorithms are inspired by Darwinian evolution, where the individuals face selection and reproduce mixing their features. The survivors are selected through the definition of fitness: the best individuals are the ones that obtain the highest score and will survive, transmitting their attributes.

New sentences are generated by randomly mutating existing ones, a fitness is calculated in order to decide which sentence survives and which is removed. Repeating the loop many times, we expect to generate always better sentences, meaning sentences with high fitness score.

The key idea is to define the fitness function through a language model. For example, BERT finetuned for CoLA task takes as input a sentence x_i and outputs a score $s_i \in [0, 1]$ representing a confidence if the given sentence is grammatically correct or not. Using BERT finetuned as the fitness function implies that a sentence is more likely to "survive" if it is evaluated grammatically correct by BERT.

The full algorithm is summarized in Algorithm 1. Firstly, a set of 1000 sentences is randomly generated by extracting words from a distribution. The length of each sentence (i.e. the number of words) is randomly picked by a Poisson distribution with mean equal to the mean length of sentences in the training set ($\mu = 7.696$), in order to obtain an initial dataset similar to the one that the model has been trained with. We set the minimum length of sentences as 4. The fitness of the first generation of sentences is evaluated using the finetuned language model. Then, sentences are sorted by their score and the best 10% are selected and duplicated 10 times each, obtaining again a set of 1000 sentences. The obtained set of sentences is mutated randomly, making sure that at least one copy of the previous generation is kept intact. The following mutations are implemented:

1. Replacement: a random set of words is replaced by other words picked from the same distribution as the initial set (a word is mutated with probability $\alpha = 0.2$);
2. Deletion: a random word is deleted (with probability $\beta = 0.1$);
3. Insertion: a random word is inserted in a random place of the sentence (with probability $\gamma = 0.3$).

The parameters that governs the mutations are selected intuitively in order to explore the sentences space as much as possible, without producing too many variations to the original sentences. We observed that the results are not strongly dependent on those parameters. However, they determine the exploration capabilities of the sentences' space: higher values of α, β and γ imply a chaotic path, with "longer steps" among generations, while low values imply "short steps", slowing but not compromising the algorithm.

Once the sentences are mutated, we obtain a new set of sentences, more or less similar to the 100 ones selected before from the original set. Iterating this procedure few times, we expect the scores to increase until they reach similar values to the ones obtained when real correct sentences are given as input.

We are aware that this procedure could not be able to explore the full sentences' space, being extremely dependent on the first generation set. However, this procedure is not expected to be exhaustive, but to find at least a few fooling samples.

FOOLING ALGORITHM
Initialize 1000 random sentences of length $L \sim Poisson(\mu)$;
for *L loops* **do**
    scores $\leftarrow$ LM(sentences);
    Sort sentences by scores;
    Replicate the best 100 sentences 10 times;
    for *each sentence* **do**
        Replace each word with probability α with a random word;
        Delete a random word with probability β;
        Insert a word with probability γ in a random position;
    end
end

Algorithm 1: Evolutionary fooling algorithm

6 Experiments and Results

6.1 Initial Finetuning

One of the main advantages of BERT is the generalization ability between different tasks, obtained by an unsupervised pre-training procedure, followed by the insertion of a task specific layer and a complete or partial finetuning step. Since our goal is to test and fool BERT, we need to obtain a model as similar as possible to the one used to reach the state-of-the-art results on one or more tasks. Using the same hyper-parameters as the ones described in the official paper, we finetuned the models for a specific task (CoLA), obtaining performances similar to the ones reported in the official GLUE benchmark leaderboard, implying that the model we have trained is similar to the official one. We remark that this step is necessary since the finetuned models are not available to download, but only pre-trained ones are.

6.2 Qualitative Fooling of Original BERT

Firstly, the Fooling algorithm (Algorithm 1) described above is applied using BERT as a fitness estimator.

We delineate here the expected outcomes. If the fooler is not able to generate sentences that BERT classifies with high score even after many iterations, it could mean that the generation process is too random and it must be refined. The algorithm has been chosen to be as simple as possible and as random as possible in order to explore the "sentences space" in an general unsupervised way. If this strategy does not work, better generation techniques should be chosen, sacrificing the total randomness of this method.

Instead, if the algorithm is able to generate sentences that BERT classifies with high score, there are two possible reasons. The generator could actually generate correct sentences. In this case, BERT is not fooled because it is actually scoring correctly the input sentences. Since the generation algorithm is extremely simple, we can almost surely assert that BERT cannot be fooled through this evolutionary procedure.
Otherwise, if the sentences generated are random sentences, as expected, given the nature of the fooling algorithm, we succeeded by finding a set of sentences composed by random words that BERT classifies as correct. The selection of a simple generating algorithm is made to increase the probability that the generated sentences are actually random.

Since we are interested in sentences that fool the language model, we evaluate only the generated sentences that obtain from the model a score higher than 0.98. We set this threshold not too low to be surprised if a random sentence is predicted with a so high probability, even if every sentence that obtains a score higher than 0.5 will be classified as grammatical. To evaluate the performance, the selected sentences are qualitatively evaluated by manually reading them.

In table 1, a sample of fooling sentences and their respective scores are reported to be manually evaluated. It can be easily noticed that, even if the score computed by BERT is high, the sentences are collections or random words, meaning that BERT is not able to understand the real difference between correct sentences and selected random collections of words.

Table 1: Sample of wrong sentences generated using the fooling algorithm on BERT

Sentence	**Score**
block characters rude cedar plum pronoun climbers	0.993
Pfizer Dracula articles lousy scissors firemen Genie Letters	0.990
umps teacher Abbey Gillespie brave scones answer exercises	0.980

However, the outcome of the fooling algorithm is not always positive. Different runs converge to different results. It is not rare to generate correct sentences, mostly when sentences are short, depending on the initial random set. An example is shown in table 2, where a sample of sentences from a different run is reported. The generated sentences are grammatically correct or almost, meaning that we are not surprised if the model classifies them as grammatical with high scores even if they are not completely correct. We report samples of correct ones and almost correct ones generated from the same run to show qualitatively the results of procedures that we consider not successful and we don't use them for the following analysis.

Table 2: Sample of correct and "almost" correct sentences generated using the fooling algorithm on BERT

Correct sentences	**Score**
Students admired metal art gently	0.998
We're expecting frustrating unflattering airport Experts poisonous	0.997
citizens nearly begged rice	0.997
"Almost" correct sentences	**Score**
NY restaurant student Shelly dove lack	0.998
Calvin nearly begged wooden	0.998
Maxwell's kicking made polarity News Harold	0.997

In figure 1, we show the distribution of scores during a fooling run of $L = 9$ loops that succeeded. At the beginning, the sentences randomly generated obtain a low score. During the process, the distribution flatten, finally reaching also scores of about 1.

We show in figure 2 the distribution of scores when a fooling run fails. In this case, the sentences generated are grammatically correct. After few iterations, the distribution of scores is the union of the distribution of correct sentences, with scores near to 1, and the distribution of their random variations, with scores near to 0.

This difference between behaviors can be an indicator of the performance of the fooling algorithm. Without manually looking at the sentences generated, if we observe a distribution of scores similar to figure 1, we can assert that the algorithm succeeded, while if the distribution is similar to the one in figure 2, the generated sentences are probably correct.

6.3 ADVERSARIAL TRAINING

Once BERT has been fooled, the next goal is to better understand if the problem is about the dataset or about BERT itself. Thus, we tried to augment the dataset with sentences that fool BERT and verify if the model is able to generalize or not. We are answering the question: can we teach BERT how to recognize fooling sentences?

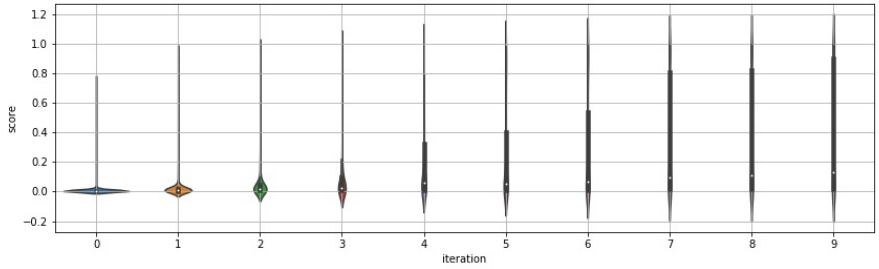

Figure 1: Example of distributions of the BERT scores when fooling sentences are found

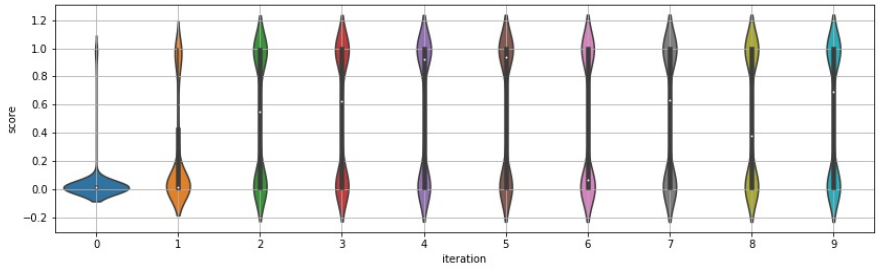

Figure 2: Example of distributions of the BERT scores when fooling sentences are not found

After running the fooling algorithm on the same initial BERT model n times, we collect all the sentences that obtain a BERT score greater than a fixed threshold of 0.98 and we merge those as negative samples to the original CoLA dataset, obtaining an augmented dataset. Finally, we finetune the original pre-trained BERT model using this new augmented dataset. We call the obtained model *Improved BERT* in the following sections and we check the performance of the new model calculating the score through the official GLUE benchmark leaderboard, obtaining a result similar to the original score.

6.4 Qualitative Fooling of Improved BERT

Once verified that the improved model is similar to the original one in the official CoLA task, we apply the fooling algorithm to it, evaluating qualitatively the generated sentences.
This is necessary to understand if improving the dataset is enough to make a model so robust to not be fooled again or if the model itself is not able to generalize enough. If the fooler cannot fool the improved BERT (obtaining low scores or generating real sentences) then it means that the dataset was not built to deal with random sentences. By adding samples of random sentences we succeeded at teaching the model not to be fooled again. Otherwise, if the fooling algorithm can still fool the improved version of BERT, we can conclude that, even if the model was presented with some random sentences in the training set, the space of random sentences is still too sparse to be fully generalized. The model, then, is not good enough to deal with this kind of issue and a new model has to be designed to overcome the problem.

Table 3 shows qualitatively the fooling sentences of an improved version of BERT. It can be noticed that the sentences are still random collections of words but shorter. In figure 3, the distribution of scores is presented, showing a not so marked separation of distributions as in figure 2, belonging to a run where the high score sentences are correct, but also not so flat as the one in figure 1, where the high scores sentences are random collection of words.

Table 3: Sample of correct sentences generated using the fooling algorithm on improved BERT

Sentence	Score
handle introduced manuscript handkerchief beans	0.994
bear pesto camera bugs abound	0.994
boat presentation freedom boxes	0.993

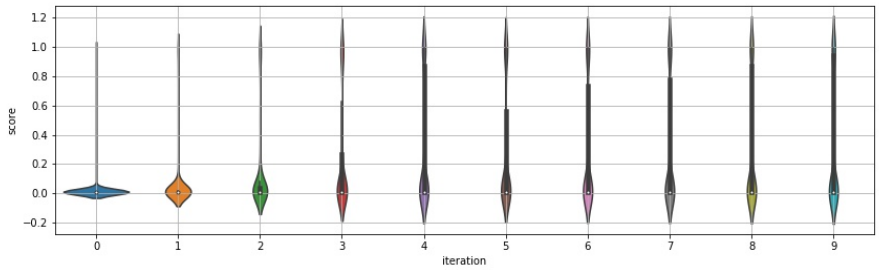

Figure 3: Example of distributions of the improved BERT scores when fooling sentences are not found

6.5 Robustness Experiments

Now we investigate the robustness of the models, to understand if the fooling datasets are universal or not, performing the two following experiments.

6.5.1 Fooling two versions of BERT

Obviously, different finetuning of the same pre-trained BERT model, with the same dataset, can lead to different models. These models will obtain similar performance on the GLUE test set, but we want to understand if they also obtain similar performances when subject to adversarial attacks. Firstly, a fooling set of sentences is obtained using the first version of BERT. Then, the second version of BERT is tested on the generated set. We performed this experiment using 3 different finetuned BERT models and 5 different set of fooling sentences. The models are always fooled, obtaining 0% accuracy, meaning that the generated sentences are general, and not dependents of the randomness of the training procedure.

6.5.2 Fooling Improved BERT

To test if the improved model, trained with adversarial samples merged to the original dataset, learnt how to recognize a fooling sentence, we designed the following experiment. We generate 5 sets of fooling sentences using the first version of BERT and we perform 5 separate adversarial trainings obtaining 5 different improved models. We tested each model with the remaining 4 sets of generated fooling sentences. The improved models are never fooled, obtaining a 100% accuracy, meaning that if the model is presented with samples of random sentences during its finetuning phase, it will be able to recognize every sentence generated in a similar way.

6.6 Fooling RoBERTa

The following analysis regards the comparison of BERT with another pre-trained language model: RoBERTa (Liu et al. (2019)). We aim to understand if pretrained language models can be fooled in the same way and if they have different weak spots. We repeat the whole fooling procedure as described earlier for BERT, using RoBERTa to evaluate the fitness of sentences.

Firstly, the finetuned model is tested using the official GLUE benchmark leaderbord, obtaining a score similar to the official one.

Then, a fooling dataset is generated using algorithm 1, and a sample of it is shown in table 4.

Table 4: Sample of sentences generated using the fooling algorithm on RoBERTa

Sentence	Score
treading monkeys won next	0.993
Yeltsin used spaghetti selfish	0.991
mauve dungeon dangers tickle Bingley	0.990

A qualitative inspection of the generated sentences suggests that, even if there is a big fraction of sentences that can be considered grammatically correct even if meaningless, there could be still found samples of random sentences classified wrongly with high scores. These sentences are usually shorter than the ones generated using BERT. We can interpret this result as a strength of RoBERTa, since the longer random sentences are discarded in the firsts iterations, selecting the shorter ones that are easier to be generated correctly using a random procedure. Tuning the parameter γ, that defines the probability of randomly deleting words, we are still able to direct the algorithm in the direction we want, so to generate fooling sentences.

6.7 Comparison between BERT and RoBERTa

In the last experiment, we aim to compare the performances of both models, using sentences generated fooling one of them to test the other. If the tested model is able to correctly predict that the sentences are all incorrect, we can conclude that the fooling sentences are characteristic of a specific model (the first one, used to generate them), not general difficult sets of random words. Instead, if there is an overlap of fooling sentences, meaning that the tested model is not able to predict that all of the input sentences, that previously fooled the first model, are incorrect, we can conclude that the generated set is general, it is a portion of difficult set of words that the language models selected are not able to predict yet, due to their architectures or due to the training set used. The results are obtained testing 3 finetuned BERT and RoBERTa models with 5 generated set of fooling sentences each. We observe an asymmetric behaviour: RoBERTa obtains 100% accuracy when tested on sentences generated fooling BERT, while BERT obtains a 0% accuracy when tested on sentences generated fooling RoBERTa. Since we observe that the set of sentences fooling RoBERTa is a subset of the set of sentences fooling BERT, we can conclude that RoBERTa is a stronger model than BERT against this kind of adversarial attacks. Even if both models can be fooled, the smaller the set of fooling sentences, the harder it is to generate them.

7 Conclusion

A simple generative algorithm to test the robustness and prediction capabilities of some of the best pre-trained language models is proposed and tested. The described procedure is able to find sentences of random words classified by state-of-the-art language models with high precision as grammatically correct. Augmenting datasets with samples of random sentences helps increasing the prediction abilities of the models, but the fooling procedure still works when applied to these models. A crossed test between BERT and RoBERTa is performed, showing the superiority of the latter model. The results suggests that, even if such models are able to obtain surprising results on many benchmark datasets, they are still far from truly understanding languages.

Future work will involve similar experiments on other language models (such as GPT-2 (Radford et al. (2018)) and XLNet(Yang et al. (2019))) to understand if different architectures influence the prenformances and tasks (multi-sentences tasks, such as MRPC dataset (Dolan & Brockett (2005)) in GLUE benchmark), obtaining more statistically relevant analysis.

References

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. What does BERT look at? an analysis of bert's attention. *CoRR*, abs/1906.04341, 2019. URL `http://arxiv.org/abs/1906.04341`.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Bill Dolan and Chris Brockett. Automatically constructing a corpus of sentential paraphrases. In *Third International Workshop on Paraphrasing (IWP2005)*. Asia Federation of Natural Language Processing, January 2005. URL https://www.microsoft.com/en-us/research/publication/automatically-constructing-a-corpus-of-sentential-paraphrases/.

Javid Ebrahimi, Anyi Rao, Daniel Lowd, and Dejing Dou. Hotflip: White-box adversarial examples for NLP. *CoRR*, abs/1712.06751, 2017. URL http://arxiv.org/abs/1712.06751.

Di Jin, Zhijing Jin, Joey Tianyi Zhou, and Peter Szolovits. Is BERT really robust? natural language attack on text classification and entailment. *CoRR*, abs/1907.11932, 2019. URL http://arxiv.org/abs/1907.11932.

Jinfeng Li, Shouling Ji, Tianyu Du, Bo Li, and Ting Wang. Textbugger: Generating adversarial text against real-world applications. *CoRR*, abs/1812.05271, 2018. URL http://arxiv.org/abs/1812.05271.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

Paul Michel, Omer Levy, and Graham Neubig. Are sixteen heads really better than one? *CoRR*, abs/1905.10650, 2019. URL http://arxiv.org/abs/1905.10650.

Anh Mai Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 427–436, 2014.

Timothy Niven and Hung-Yu Kao. Probing neural network comprehension of natural language arguments. *CoRR*, abs/1907.07355, 2019. URL http://arxiv.org/abs/1907.07355.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2018. URL https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf.

Livio Baldini Soares, Nicholas FitzGerald, Jeffrey Ling, and Tom Kwiatkowski. Matching the blanks: Distributional similarity for relation learning. *CoRR*, abs/1906.03158, 2019. URL http://arxiv.org/abs/1906.03158.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. BERT rediscovers the classical NLP pipeline. *CoRR*, abs/1905.05950, 2019. URL http://arxiv.org/abs/1905.05950.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL http://arxiv.org/abs/1706.03762.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. *CoRR*, abs/1804.07461, 2018. URL http://arxiv.org/abs/1804.07461.

Alex Warstadt, Amanpreet Singh, and Samuel R. Bowman. Neural network acceptability judgments. *CoRR*, abs/1805.12471, 2018. URL http://arxiv.org/abs/1805.12471.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. *CoRR*, abs/1906.08237, 2019. URL http://arxiv.org/abs/1906.08237.

Zhengli Zhao, Dheeru Dua, and Sameer Singh. Generating natural adversarial examples. *CoRR*, abs/1710.11342, 2017. URL http://arxiv.org/abs/1710.11342.