Comparing Toxic Comment Classification models: One- Shot, Zero-Shot, Few-Shot and Fine Tuning with Transformers

Abstract:

In this work, I present a comparison of four deep-learning techniques: One Shot, Zero Shot, Few Shot and Fine Tuning, to identify toxic comments from Internet discussions. My main goal is to show the performance of each technique with Transformer models.
For Zero, One and Few shot learning I used GPT-3, T5 and BART models, and for fine-tuning I used GPT-2 model. Experiments were performed on the dataset from Wikipedia (Jigsaw's dataset) Toxic Comment, and the results were compared to the BERT fine-tuning model from paper [6].

Previews work:

Toxic comment classification is a common task that has been done in many ways, such as Logistic Regression, Naive Bayes, SVM, AdaBoost and KNN as part of ML.
Also with DL models such as LSTM, BERT, CNN, etc.

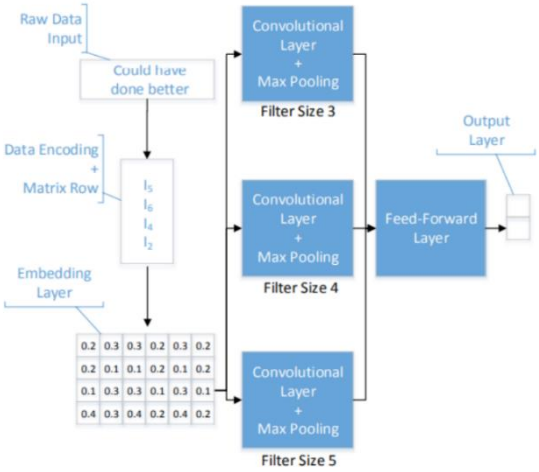
The research on toxic comment classification became active since 2018. It is due to the release of Jigsaw's data set [1]. From this year the number of papers is growing and this is an indicator that this research topic is actual and trendy.

In paper [3], they used various methods (ML and DL) on Jigsaw's data set, one of them was KNN. A KNN classification method was adapted for multilabel classification. MLkNN (Multi-Label k-Nearest Neighbor) builds uses k-NN, finds the nearest examples to a test class and uses Bayesian inference to select assigned labels. they got 58.46% for accuracy.

In paper [4], they simplified the data set to a binary classification of whether the comment is toxic or not toxic. They compare between Naive Bayes (NB) to LSTM and demonstrated that the LSTM solution provides substantial improvement in classification versus a baseline Naive Bayes-based solution. To recall, the True Positive Rate of LSTM was almost 20% higher than Naive Bayes method.

Metric	Naïve Bayes (NB)	Long Short Term Memory (LSTM)
Total Positive Rate (TPR)	48	67
F1	64	73
Precision	94	81
Recall	48	66

In paper [5], they used CNN to classify toxic text and their workflow was like that:



Word embeddings and CNN are compared against the BoW approach for which four well-established text classification methods namely Support Vector Machines (SVM), Naive Bayes (NB), k-Nearest Neighbor (KNN) and Linear Discriminated Analysis (LDA) applied on the designed DTMs. They used Jigsaw's data set and evaluated using Mean values and Standard Deviation across all experiments for Accuracy, Specificity and False discovery rate.

	Accuracy		Specificity		False disc.rate	
	Mean	Std	Mean	Std	Mean	Std
CNN_{fix}	0.912	0.002	0.917	0.006	0.083	0.007
CNN_{rand}	0.895	0.003	0.906	0.015	0.092	0.017
kNN	0.697	0.008	0.590	0.016	0.335	0.010
LDA	0.808	0.005	0.826	0.010	0.179	0.009
NB	0.719	0.005	0.776	0.012	0.250	0.010
SVM	0.811	0.007	0.841	0.012	0.167	0.012

In paper [6], they aimed to analyze the influence of different pre-processing techniques and text representations including standard TF-IDF, pre-trained word embeddings and also explored currently popular transformer models. Experiments were performed on the dataset from the Kaggle Toxic Comment Classification competition (Jigsaw's data set), and the best performing model was compared with similar approaches using standard metrics used in data analysis.

They used BERT models for classification and compared the performance of the model, bare BERT, its DistilBERT and XLNet variants.

To evaluate the models, they decided to use the standard metrics used in classification, e.g., accuracy, precision, recall and F1 score.

Table 10. Performance of the transformer models.

	Accuracy	AUC Score		Precision	Recall	F1 Score
BERT-base (cased)	0.8884	0.9624	micro avg.:	0.66	0.64	0.65
			macro avg.:	0.34	0.34	0.34
BERT-base (uncased)	0.8798	0.9700	micro avg.:	0.61	0.71	0.65
			macro avg.:	0.31	0.38	0.34
bare BERT-base (cased)	0.8998	0.9802	micro avg.:	0.69	0.68	0.69
			macro avg.:	0.63	0.51	0.51
bare BERT-base (uncased)	0.8841	0.9839	micro avg.:	0.60	0.78	**0.68**
			macro avg.:	0.60	0.58	**0.57**
bare BERT-large (cased)	0.8795	0.9801	micro avg.:	0.62	0.73	0.67
			macro avg.:	0.53	0.51	0.51
bare BERT-large (uncased)	0.8921	0.9806	micro avg.:	0.67	0.68	0.67
			macro avg.:	0.60	0.41	0.42
DistilBERT (cased)	0.8866	0.9649	micro avg.:	0.63	0.68	0.66
			macro avg.:	0.34	0.37	0.34
DistilBERT (uncased)	0.8649	0.9781	micro avg.:	0.59	0.73	0.65
			macro avg.:	0.48	0.39	0.34
XLNet (cased)	0.9630	0.9530	micro avg.:	0.60	0.70	0.65
			macro avg.:	0.30	0.37	0.33

The BERT-base uncased model provided the best results among the transformer models.

Techniques:

Zero-shot learning, few-shot learning and one-shot learning are all techniques that allow a machine learning model to make predictions for new classes with limited labeled data. The choice of technique depends on the specific problem and the amount of labeled data available for new categories or labels (classes).

Zero-shot learning – there is absolutely no labeled data available for new classes. The goal is for the algorithm to make predictions about new classes by using prior knowledge about the relationships that exist between classes it already knows. In the case of large language models (LLMs) like ChatGPT, for example, prior knowledge is likely to include semantic similarities.

One-shot learning – providing one labeled example. The goal is to make predictions for the new classes based on this single example.

Few-shot learning – there is a limited number of labeled examples. The goal is to make predictions for new classes based on just a few examples of labeled data.

Fine-Tuning - has been the most common approach in recent years and involves updating the weights of a pre-trained model by training on a supervised dataset specific to the desired task. Typically, thousands to hundreds of thousands of labeled examples are used. The main advantage of fine-tuning is strong performance on many benchmarks.[7]

Data:

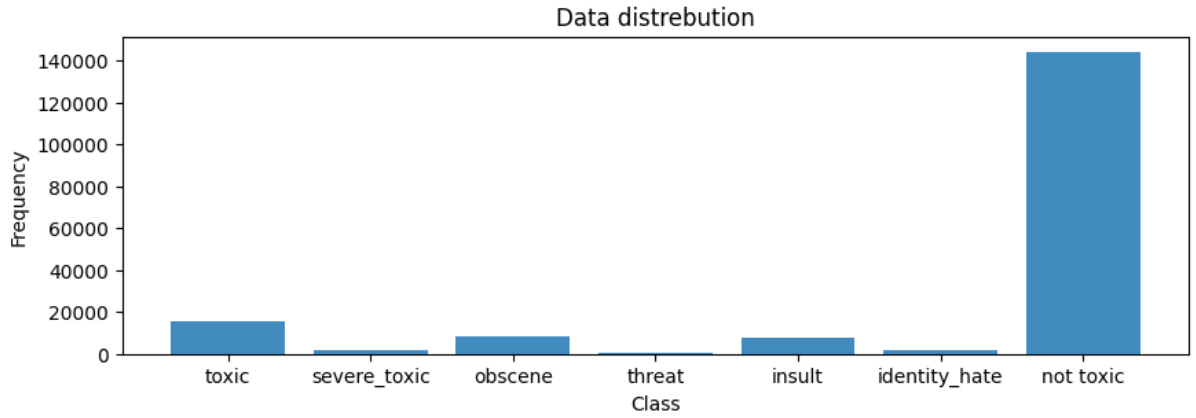
I used the Wikipedia comments dataset (Jigsaw's dataset) which has been labeled by human raters for toxic behavior. The types of toxicity are:

- toxic
- severe toxic
- obscene
- threat
- insult
- identity hate

for example:

	id	comment_text	toxic	severe_toxic	obscene	threat	insult	identity_hate
600	0199d6af27b715f3	I'm also a sock puppet of this account...SUPRI...	1	0	0	1	0	0
601	019ab7fbe83530b6	All the sources needed are provided, all you n...	0	0	0	0	0	0
602	019ab9a91e7fcdbe	Are you fucker mother fucker have nothing to d...	1	0	1	0	1	0
603	019bf50235ccbe37	It's from one of the many books on various ban...	0	0	0	0	0	0
604	019c4ce48e6bfbab	HELLO \n\nYou disgrace to humanity. Stop wasti...	1	0	1	0	1	0
605	019c9815c304f9e9	I've explained my reasoning for this block at ...	0	0	0	0	0	0

The dataset is multilabel and has 159,571 samples.



Algorithms:

Transformer models have revolutionized the field of natural language processing (NLP) by introducing a highly effective approach to language understanding and generation. Among the notable examples of Transformer models are GPT-2, GPT-3, T5, and BART.

GPT-2 (Generative Pre-trained Transformer 2): OpenAI decoder only model that was proposed in Language Models are Unsupervised Multitask Learners paper [8]. It's a causal (unidirectional) transformer pretrained using language modeling on a very large corpus of ~40 GB of text data. GPT-2 is a large transformer-based language model with 1.5 billion

parameters, trained on a dataset of 8 million web pages. Trained with a simple objective: predict the next word, given all of the previous words within some text.

GPT-3 (Generative Pre-trained Transformer 3): the successor to GPT-2, took the capabilities of language generation to a whole new level. The architecture is a decoder-only transformer with a 2048-token-long context and 175 billion parameters. Its massive size enabled it to grasp complex contextual information, making it capable of answering questions, translating languages, and even creating human-like conversational responses.

T5 (Text-to-Text Transfer Transformer): encoder-decoder model, reframing all NLP tasks into a unified text-to-text-format where the input and output are always text strings, in contrast to BERT-style models that can only output either a class label or a span of the input. The text-to-text framework allows using the same model, loss function, and hyperparameters on any NLP task, including machine translation, document summarization, question answering, and classification tasks (e.g., sentiment analysis). Can even apply T5 to regression tasks by training it to predict the string representation of a number instead of the number itself.

BART (Bidirectional and AutoRegressive Transformers): BART uses a standard seq2seq/machine translation architecture with a bidirectional encoder (like BERT) and a left-to-right decoder (like GPT). The pretraining task involves randomly shuffling the order of the original sentences and a novel in-filling scheme, where spans of text are replaced with a single mask token. BART is particularly effective when fine-tuned for text generation but also works well for comprehension tasks. Achieves new state-of-the-art results on a range of abstractive dialogue, question answering, and summarization tasks, with gains of up to 6 ROUGE. [9]

Evaluation:

To evaluate the performance of the models, I used the following methods:

Accuracy: the ratio of the number of correct predictions to the total number of input samples.

$$Accuracy = \frac{True\ Prediction}{Total\ samples}$$

Precision: the ratio of correctly predicted positive observations to the total predicted positive observation.
Definition of some terms for understanding the calculation:

- True Positive (TP) - the correct positive prediction by the model.
- True Negative (TN) - the correct negative prediction by the model.
- False Positive (FP) - the wrong prediction of the positive by the model.
- False Negative (FN) - the wrong prediction of the negative by the model.

Precision calculates by:

$$Precision = \frac{TP}{TP + FP}$$

Recall: the fraction of samples from one class which are predicted correctly by the model.

$$Recall = \frac{TP}{TP + FN}$$

F1 score: the harmonic mean between precision and recall. It tells you how precise your classifier is, as well as how robust it is.

$$F1\ Score = \frac{2 \cdot (Precision \cdot Recall)}{Precision + Recall}$$

AUC (Area Under The Curve) score: provides an aggregate measure of performance across all possible classification thresholds. One way of interpreting AUC is as the probability that the model ranks a random positive example more highly than a random negative example. The AUC score ranges from 0 to 1, where a score of 0.5 indicates a model with random performance, and a score of 1 represents a perfect classifier.

Experiments:

In this part, I will present all the experiments I did with their results. It will be divided into the different Transformer models and the techniques I used. I created a balanced test dataset of 200 samples from the full dataset for the evaluation of the models.

GPT-3 & T5:

For GPT-3 and T5 Transformers, I used Zero shot, One shot and Few shot learning.

Zero-shot learning:

I created a prompt to feed the model and asked it to classify a comment and provided the labels. For example:

```
Classify the following comment as toxic or not and the types of toxicity:

 I'm also a sock puppet of this account...SUPRISE!!
-sincerely,
           The man that will track you down from the Internet and kill you

Labels: ['toxic', 'severe_toxic', 'obscene', 'threat', 'insult', 'identity_hate']
```

I converted the model answer into array of zeros and ones as the labels structure of the dataset by searching each label in the output.

The results are:

	Accuracy	*Precision*	*Recall*	*F1 score*	*AUC score*
ZS GPT-3	*48.5%*	*micro avg: 41%* *macro avg: 39%*	*micro avg: 73%* *macro avg: 67%*	*micro avg: 52%* *macro avg: 43%*	*72%*
ZS T5	*24.5%*	*micro avg: 21%* *macro avg: 20%*	*micro avg: 72%* *macro avg: 59%*	*micro avg: 33%* *macro avg: 26%*	*53.1%*

One-shot learning:

For one shot learning I added one example to the prompt:

```
Classify the following comment as toxic or not and the types of toxicity.
With the following labels: ['toxic', 'severe_toxic', 'obscene', 'threat', 'insult', 'identity_hate']

For example:
Comment: COCKSUCKER BEFORE YOU PISS AROUND ON MY WORK
Labels: toxic, severe_toxic, obscene, insult.

Comment to classify:
Are you fucker mother fucker have nothing to do but block University computers. Go and suck cocks
```

The results are:

	Accuracy	*Precision*	*Recall*	*F1 score*	*AUC score*
OS GPT-3	*52.5%*	*micro avg: 63%* *macro avg: 58%*	*micro avg: 70%* *macro avg: 78%*	*micro avg: 66%* *macro avg: 61%*	*84.1%*
OS T5	*19%*	*micro avg: 23%* *macro avg: 22%*	*micro avg: 87%* *macro avg: 91%*	*micro avg: 37%* *macro avg: 31%*	*62.9%*

Few-shot learning:

For Few shot learning I added 10 examples to the prompt and the results are:

	Accuracy	*Precision*	*Recall*	*F1 score*	*AUC score*
FS GPT-3	*49.5%*	*micro avg: 60%* *macro avg: 50%*	*micro avg: 76%* *macro avg: 70%*	*micro avg: 67%* *macro avg: 54%*	*78.9%*
FS T5	*12.5%*	*micro avg: 33 %* *macro avg: 16%*	*micro avg: 90%* *macro avg: 50%*	*micro avg: 48%* *macro avg: 24%*	*49.9%*

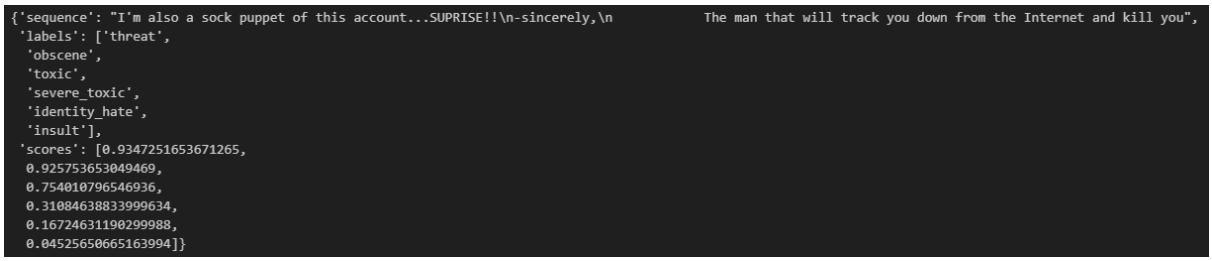
BART:

With BART model I used different methods for Zero shot, One shot and Few shot learning, that don't use the prompt for classification.

Zero-shot learning:

I used BART NLI-based Zero Shot Text Classification. a method for using pre-trained NLI models as a ready-made zero-shot sequence classifiers. The method works by posing the sequence to be classified as the NLI premise and to construct a hypothesis from each candidate label. This method is surprisingly effective in many cases, particularly when used with larger pre-trained models like BART.

The way it works is that you enter a sequence to classify and the candidate labels, and the model outputs probability scores from 0 to 1 for each label. For example:



I classified the output scores as 0 if it's lower than 0.5 or 1 if it's higher than 0.5 and created an array with the labels structure of the dataset.

The results are:

	Accuracy	*Precision*	*Recall*	*F1 score*	*AUC score*
ZS BART	*22%*	*micro avg: 33%* *macro avg: 32%*	*micro avg: 91%* *macro avg: 87%*	*micro avg: 48%* *macro avg: 42%*	*71.4%*

One-shot learning:

For One shot learning I fine-tuned BART classification model with one sample without using prompt to see if it can get decent results without training with a big dataset and number of epochs.

The results are:

	Accuracy	*Precision*	*Recall*	*F1 score*	*AUC score*
OS BART	*1.5%*	*micro avg: 25%* *macro avg: 17%*	*micro avg: 95%* *macro avg: 67%*	*micro avg: 40%* *macro avg: 25%*	*50%*

Few-shot learning:

For Few shot learning I fine-tuned BART classification model with 10 balanced samples.

The results are:

	Accuracy	*Precision*	*Recall*	*F1 score*	*AUC score*
FS BART	*3.5%*	*micro avg: 35%* *macro avg: 12%*	*micro avg: 66%* *macro avg: 33%*	*micro avg: 46%* *macro avg: 17%*	*50%*

GPT-2:

In this part, I did fine-tuning to the GPT-2 Transformer model with the full train set.
Since GPT-2 is a decoder transformer, the last token of the input sequence is used to make predictions about the next token that should follow the input. This means that the last token of the input sequence contains all the information needed in the prediction. I can use that information to make a prediction in a classification task instead of a generation task. In other words, I will use the last token embedding to make a prediction with GPT-2.

I split the data into training (111,700 samples), validation (31,914 samples), and test (15,957 samples). I used the GPT-2 For Sequence Classification architecture from hugging face which is the GPT-2 Model with a sequence classification head on top (linear layer).

I used ClearML platform to track my experiments and document the model architecture, configuration, loss, and accuracy graphs.

For the training process I used the following hyperparameters:

- Loss Function: BCEWithLogitsLoss.
- Optimizer: AdamW.
- Initial Learning Rate: 2e-5 and drops with get_linear_schedule_with_warmup.
- Number of epochs: 4.
- Max tokens length: 60.

The following graphs show the training process, accuracy and loss:

The test results are:

	Accuracy	*Precision*	*Recall*	*F1 score*	*AUC score*
GPT-2 fine-tuning	*90%*	*micro avg: 73%* *macro avg: 53%*	*micro avg: 67%* *macro avg: 43%*	*micro avg: 70%* *macro avg: 44%*	*96.1%*

Discussion:

To compare the different approaches I studied in this work, I made the following table:

	Accuracy	*Precision*	*Recall*	*F1 score*	*AUC score*
ZS GPT-3	*48.5%*	*micro avg: 41%* *macro avg: 39%*	*micro avg: 73%* *macro avg: 67%*	*micro avg: 52%* *macro avg: 43%*	*72%*
ZS T5	*24.5%*	*micro avg: 21%* *macro avg: 20%*	*micro avg: 72%* *macro avg: 59%*	*micro avg: 33%* *macro avg: 26%*	*53.1%*
ZS BART	*22%*	*micro avg: 33%* *macro avg: 32%*	*micro avg: 91%* *macro avg: 87%*	*micro avg: 48%* *macro avg: 42%*	*71.4%*
OS GPT-3	*52.5%*	*micro avg: 63%* *macro avg: 58%*	*micro avg: 70%* *macro avg: 78%*	*micro avg: 66%* *macro avg: 61%*	*84.1%*
OS T5	*19%*	*micro avg: 23%* *macro avg: 22%*	*micro avg: 87%* *macro avg: 91%*	*micro avg: 37%* *macro avg: 31%*	*62.9%*
OS BART	*1.5%*	*micro avg: 25%* *macro avg: 17%*	*micro avg: 95%* *macro avg: 67%*	*micro avg: 40%* *macro avg: 25%*	*50%*
FS GPT-3	*49.5%*	*micro avg: 60%* *macro avg: 50%*	*micro avg: 76%* *macro avg: 70%*	*micro avg: 67%* *macro avg: 54%*	*78.9%*
FS T5	*12.5%*	*micro avg: 33 %* *macro avg: 16%*	*micro avg: 90%* *macro avg: 50%*	*micro avg: 48%* *macro avg: 24%*	*49.9%*
FS BART	*3.5%*	*micro avg: 35%* *macro avg: 12%*	*micro avg: 66%* *macro avg: 33%*	*micro avg: 46%* *macro avg: 17%*	*50%*
GPT-2 fine-tuning	*90%*	*micro avg: 73%* *macro avg: 53%*	*micro avg: 67%* *macro avg: 43%*	*micro avg: 70%* *macro avg: 44%*	*96.1%*
BERT-base (uncased) from paper [6]	*87.9%*	*micro avg: 61%* *macro avg: 31%*	*micro avg: 71%* *macro avg: 38%*	*micro avg: 65%* *macro avg: 34%*	*97%*

We can see that the best results for Zero shot technique are with GPT-3 model which is the latest model of them all. It's also true for One and Few shot.
For fine tuning approach I compared between the GPT-2 model and the BERR-base model from paper [6], and we can see that regarding to Accuracy, Precision, Recall macro avg and F1-score the GPT-2 model that I trained has the better edge. But the AUC score and Recall micro avg is slightly higher in the BERT model.

Overall, the GPT-2 model shows good results with fine-tuning, but there is still room for improvement.

Conclusion:

In conclusion, this work compares four deep-learning techniques—One Shot, Zero Shot, Few Shot, and Fine Tuning—specifically focusing on their performance in identifying toxic comments from Internet discussions using Transformer models. The experiments conducted utilized GPT-3, T5, BART models for Zero, One, and Few Shot learning, while GPT-2 model was employed for Fine Tuning. The dataset from Wikipedia (Jigsaw's dataset) Toxic Comment was used for evaluation, and the obtained results were compared to the BERT fine-tuning model outlined in paper [6]. The findings of this study provide valuable insights into the effectiveness of each technique.

References:

[1] M. Yao, C. Chelmis, D.-S. Zois, Cyberbullying Ends Here: Towards Robust Detection of Cyberbullying in Social Media, The Web Conference 2019 - Proceedings of the World Wide Web Conference, WWW 2019, San Francisco, USA, 2019, pp. 3427–3433.

[2] Andročec, Darko. "Machine learning methods for toxic comment classification: a systematic review" Acta Universitatis Sapientiae, Informatica, vol.12, no.2, 2020, pp.205-216. https://doi.org/10.2478/ausi-2020-0012

[3] A. N. M. Jubaer, A. Sayem and M. A. Rahman, "Bangla Toxic Comment Classification (Machine Learning and Deep Learning Approach)," *2019 8th International Conference System Modeling and Advancement in Research Trends (SMART)*, Moradabad, India, 2019, pp. 62-66, doi: 10.1109/SMART46866.2019.9117286.

[4] Zaheri, Sara, Jeff Leath, and David Stroud. "Toxic comment classification." *SMU Data Science Review* 3.1 (2020): 13.

[5] Georgakopoulos, Spiros V., et al. "Convolutional neural networks for toxic comment classification." *Proceedings of the 10th hellenic conference on artificial intelligence*. 2018.

[6] Maslej-Krešňáková, V.; Sarnovský, M.; Butka, P.; Machová, K. Comparison of Deep Learning Models and Various Text Pre-Processing Techniques for the Toxic Comments Classification. *Appl. Sci.* 2020, *10*, 8631. https://doi.org/10.3390/app10238631

[7] Brown, Tom, et al. "Language models are few-shot learners." Advances in neural information processing systems 33 (2020): 1877-1901.

[8] Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., & Sutskever, I. (2019). Language models are unsupervised multitask learners. OpenAI blog, 1(8), 9.

[9] Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2019). BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. ArXiv. /abs/1910.13461