Reproducibility report instructions

Tyler Rada, Quinton Velazquez, Jacob Rowell

{radat19, rowellj17, velazquezq19}@ecu.edu

**Reproducibility Summary**

**Scope of Reproducibility**

Ivan Montero, Nikolaos Pappas, and Noah A. Smith proposed an approach that converts a pre-trained transformer language model into a sentence-level autoencoder. This autoencoder is able to reconstruct its pretraining data. The resulting model improves the performance of the pretrained model on sentence-level tasks while maintaining its performance on multi sentence tasks.

**Methodology**

how much time did it take to produce the results, what hardware you were using and how long it took to train/evaluate.

The authors provided us with a link to their GitHub that had all of their code.

We also had access to the majority of the Datasets that the team used to train the Model.

**Results**

Start with your overall conclusion - where was your study successful and where not successful. Be specific and use precise language, e.g. "we reproduced the accuracy to within 1% of reported value, that upholds the paper’s conclusion that it performs much better than baselines". Getting exactly the same number is in most cases infeasible, so you’ll need to use your judgement call to decide if your results support the original claim of the paper.

**What was easy**

The initial installation was fairly simple. All we needed to do was copy the gitHub onto our computers and run a setup line. It took a few minutes for everything to install but it eventually worked.

First Installation took about 1 hour but it worked.

Second Installation using Ubuntu VM only took a few minutes.

**What was difficult**

Attempting to train the model itself was difficult. We kept running into errors while running the Datasets due to our personal machines not being strong enough.

Another issue we had run into is that if we were to replicate the larger experiments, our computers were not able to handle the information.

Training started off with around 15 hours but then started to throw errors and crashed.

We even tried to use the Ubuntu VM and allocated 40GB of Memory…Still failed.

**Communication with original authors**

Unfortunately we were unable to communicate with the original authors due to time constraints.

**1 Introduction**

Representation learning for text via pre-training a language model on a large scale has become a standard starting point for developing NLP systems. This approach stands in contrast to autoencoders, also trained on raw text, but with the objective of learning to encode each input as a vector that allows full reconstruction.

The authors explored the construction of a sentence-level autoencoder from a pretrained, frozen transformer language model. They adapted the masked language modeling objective as a generative, denoising one, while only training a sentence bottleneck and a single-layer modified transformer decoder.

They demonstrated that the sentence representations discovered their model achieve better quality than previous methods. The previous methods extracted representations from pretrained transformers on text similarity tasks, an example of controlled generation, and single-sentence classification tasks in the GLUE benchmark. All while using fewer parameters than large pretrained models.

**2 Scope of reproducibility**

Explain the claims from the paper you picked for the reproduction study and briefly motivate your choice. We recommend picking the claim that is the central contribution of the paper. To find what this contribution is, try to summarize the most important result of the paper in 1-2 sentences, e.g. "This paper introduces a new activation function X that outperforms a similar activation function Y on tasks Z,V,W".

Make the scope as specific as possible. It should be something that can be supported or rejected by your data. For example, this scope is too broad and lacks precise outcome (what is "strong performance"?): "Contextual embedding models have shown strong performance on a number of tasks across NLP. We will run experiments evaluating two types of contextual embedding models on datasets X, Y, and Z."

This scope is better because it’s more specific and has an outcome that can be either supported or rejected based on your work: "Finetuning Pretrained BERT on SST-2 will have higher accuracy than an LSTM trained with GloVe embeddings." 2.1 Addressed claims from the original paper

Clearly itemize the claims you are testing:

• Claim 1

• Claim 2

• Claim 3

**3.1 Model descriptions**

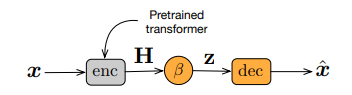
Describe the models used in the original paper, including the architecture, learning objective and the number of parameters.

The authors created AUTOBOT, a new autoencoder model for learning sentence “bottleneck” (i.e., fixed-size) representations from pretrained transformers that is useful for similarity, generation, and classification

The model has two unique components:

(i) a transformation that uses dot product attention to dynamically pool semantic information from the pretrained model’s hidden states into a sentence bottleneck representation

(ii) a shallow transformer decoder that is modified to operate based on the bottleneck representation



**Encoder:** The authors chose to keep the original encoder fixed and train a transformation β that will learn to compress H into a single representation z = β(H; θ), with θ being an additional set of parameters to be learned during finetuning.

**Decoder:** The Transformer decoder architecture expects hidden representations for every token input from the encoder in order for each output candidate to attend to each input token.

**Training considerations:** To avoid training the model from scratch, the creators fine tuned it for 100K optimization steps on a pretraining dataset using the base RoBERTa model.The model is trained using an input reconstruction loss by minimizing the negative log-likelihood computed over the reconstructed inputs.

**3.2 Datasets**

Since RoBERTa’s Datasets are not available publicly, the team had to use the Datasets that were used to train BERT.

GLUE (General Language Understanding Evaluation)- A collection of nine natural language understanding tasks, including single-sentence tasks CoLA and SST-2, similarity and paraphrasing tasks MRPC, STS-B and QQP, and natural language inference tasks MNLI, QNLI, RTE and WNLI.

[glue · Datasets at Hugging Face](https://huggingface.co/datasets/glue)

CHANGE-IT - dataset contains approximately 152,000 article-headline pairs, collected from two Italian newspapers situated at opposite ends of the political spectrum, namely la Repubblica and Il Giornale.

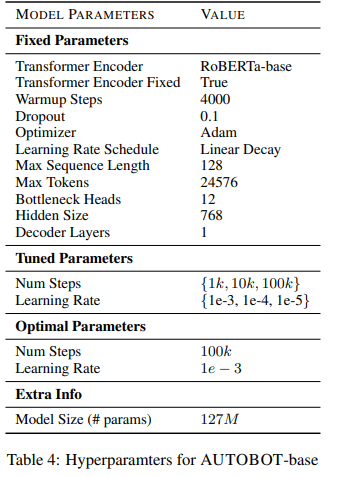
[gsarti/change\_it · Datasets at Hugging Face](https://huggingface.co/datasets/gsarti/change_it)

Both links provide the adequate source to run the Datasets. Additionally we also used the Dataset found with the TensorFlow website. TensorFlow provides the source code and examples.

[glue | TensorFlow Datasets](https://www.tensorflow.org/datasets/catalog/glue)

**3.3 Hyperparameters**

The Authors had provided the Hyperparameters that they had used.



**3.4 Implementation**

The authors provided us with a link to their GitHub that had all of their code.

The languages consisted of: Python (Most of it was done in this language), JavaScript, Shell, and C++.

[ivanmontero/autobot: Implementation of the paper 'Sentence Bottleneck Autoencoders from Transformer Language Models' (github.com)](https://github.com/ivanmontero/autobot)

**3.5 Experimental setup**

Explain how you ran your experiments, e.g. the CPU/GPU resources and provide the link to your code and notebooks.

The authors extracted the sentences from the BooksCorpus and English Wikipedia datasets to recreate the BERT dataset, and use RoBERTa-base’s pretrained tokenizer.

Unfortunately we did not have access to RoBERTa’s pre-training data in order to use their tokenizer.

**3.6 Computational requirements**

For all of the experiments involving base models, the authors used a computation cluster with 5 NVIDIA RTX 2080 TI GPU, 11GB GPU memory, and 128GB RAM.

For large models, we use a computation cluster with 4 NVIDIA TITAN RTX GPUs, 24GB GPU memory and 256GB RAM

**4 Results**

Start with a high-level overview of your results. Does your work support the claims you listed in section 2.1? Keep this section as factual and precise as possible, reserve your judgment and discussion points for the next "Discussion" section.

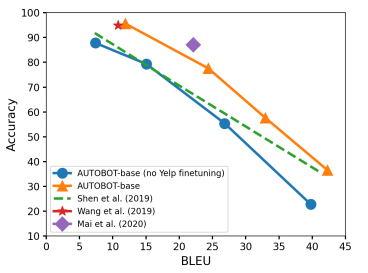
Go into each individual result you have, say how it relates to one of the claims and explain what your result is. Logically group related results into sections. Clearly state if you have gone beyond the original paper to run additional experiments and how they relate to the original claims.

Tips 1: Be specific and use precise language, e.g. "we reproduced the accuracy to within 1% of reported value, that upholds the paper’s conclusion that it performs much better than baselines".

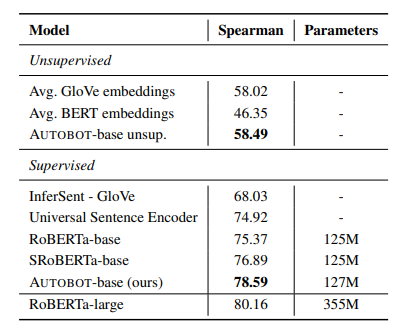
Getting exactly the same number is in most cases infeasible, so you’ll need to use your judgment call to decide if your results support the original claim of the paper.

Tips 2: You may want to use tables and figures to demonstrate your results.

**4.1 Result 1**

****

**4.2 Result 2**

****

**4.3 Additional results not present in the original paper**

Describe any additional experiments beyond the original paper. This could include experimenting with additional datasets, exploring different methods, running more ablations, or tuning the hyperparameters. For each additional experiment, clearly describe which experiment you conducted, its result, and discussions (e.g. what is the indication of the result).

**5 Discussion**

Describe larger implications of the experimental results, whether the original paper was reproducible, and if it wasn’t, what factors made it irreproducible.

Give your judgement on if you feel the evidence you got from running the code supports the claims of the paper. Discuss the strengths and weaknesses of your approach - perhaps you didn’t have time to run all the experiments, or perhaps you did additional experiments that further strengthened the claims in the paper.

**5.1 What was easy**

Describe which parts of your reproduction study were easy. E.g. was it easy to run the author’s code, or easy to re-implement their method based on the description in the paper. The goal of this section is to summarize to the reader which parts of the original paper they could easily apply to their problem.

Tips: Be careful not to give sweeping generalizations. Something that is easy for you might be difficult to others. Put what was easy in context and explain why it was easy (e.g. code had extensive API documentation and a lot of examples that matched experiments in papers).

**5.2 What was difficult**

Describe which parts of your reproduction study were difficult or took much more time than you expected. Perhaps the data was not available and you couldn’t verify some experiments, or the author’s code was broken and had to be debugged first. Or, perhaps some experiments just take too much time/resources to run and you couldn’t verify them. The purpose of this section is to indicate to the reader which parts of the original paper are either difficult to re-use, or require a significant amount of work and resources to verify.

Tips: Be careful to put your discussion in context. For example, don’t say "the maths was difficult to follow", say "the math requires advanced knowledge of calculus to follow".

**5.3 Recommendations for reproducibility**

Unfortunately due to time constraints, we were not able to reach out to the original authors.

**References**

[Sentence Bottleneck Autoencoders from Transformer Language Models (aclanthology.org)](https://aclanthology.org/2021.emnlp-main.137.pdf)