Comparison of Transformer-based Deep Learning Methods for the Paraphrase Identification Task

Oleksandr Marchenko and Vitalii Vrublevskyi

Taras Shevchenko National University of Kyiv, 64/13, Volodymyrska St., Kyiv, 01601, Ukraine

Abstract
The paper highlights the importance of paraphrase identification in various real-world applications, including information retrieval, machine translation, sentiment analysis, and question answering. It emphasises the significance of recognising different wordings that convey the same meaning and how this capability can enhance intelligent systems across domains. The paper aims to explore Transformer-based deep learning methods for paraphrase identification, offering insights into their capabilities and limitations to advance natural language understanding and inform researchers and practitioners in the field. It highlights that Transformer-based models, particularly BERT and its variants, have become state-of-the-art methods due to their ability to capture contextual information effectively and handle diverse linguistic expressions. The subsequent section of the paper will comprehensively compare these Transformer-based methods and discuss techniques for finetuning them for paraphrase detection tasks. Large Language Models were also explored, and a method was described to finetune them on classification tasks.

Keywords [1]
Natural language processing, paraphrase identification, machine learning, transformer-based models, large language models.

1. Introduction

Identifying paraphrasing has become pivotal and challenging in the ever-evolving landscape of natural language processing and understanding. Paraphrasing, a linguistic phenomenon wherein different wordings convey the same meaning, is one of the cornerstones of human communication and comprehension. Understanding and using the power of paraphrase identification have profound implications across various domains, from information retrieval and machine translation to sentiment analysis and question answering.

The ability to discern and differentiate paraphrases is vital in numerous real-world applications. Consider the following examples:

1. **Information Retrieval**: Search engines strive to provide users with the most relevant and comprehensive results. Paraphrase identification can enhance retrieval systems by recognizing different formulations of user queries and retrieving documents that may use different words but convey the same information.

Example: A user searching for "effects of climate change" might also benefit from results that mention "consequences of global warming."

2. **Machine Translation**: In translation, recognising paraphrases is very useful. Accurate paraphrase identification can aid machine translation systems in generating contextually appropriate translations by selecting from many possible wordings.

Example: When translating "Je suis fatigué" from French to English, the system should recognize that both "I am tired" and "I feel exhausted" convey the same underlying meaning.

Information Technology and Implementation (IT&I-2023), November 20-21, 2023, Kyiv, Ukraine
EMAIL: rozenkrans17@gmail.com (Oleksandr Marchenko); vitalii.vrublevskyi@gmail.com (Vitalii Vrublevskyi)
ORCID: 0000-0002-5408-5279 (Oleksandr Marchenko); 0009-0005-7070-9001 (Vitalii Vrublevskyi)

3. **Sentiment Analysis**: Social media and customer reviews are abundant data sources, often full of paraphrased expressions. Paraphrase identification can help sentiment analysis models decipher the sentiment behind various rephrasings of the same sentiment.

Example: Identifying "I absolutely love this product" and "This product is fantastic!" as paraphrases allows for a more nuanced understanding of user sentiment.

4. **Question Answering**: In question-answering systems, paraphrase identification plays a crucial role. It helps match user queries to potential answers, even when the wording varies significantly.

Example: For the question "What are the effects of smoking?" paraphrase identification can assist in recognizing that "What harm does smoking cause?" is a paraphrase of the same inquiry.

In this paper, we delve into paraphrase identification and comprehensively explore Transformer-based deep learning methods for this essential task. By leveraging state-of-the-art models and innovative techniques, we aim to shed light on the capabilities and limitations of these methods in recognising paraphrases, thus contributing to the advancement of natural language understanding and facilitating the development of intelligent systems in a multitude of applications.

In the subsequent sections, we will detail our methodology, dataset, experiments, and results to provide insights that can guide practitioners and researchers in paraphrase identification.

2. Overview of different methods to make paraphrase identification

Various approaches have been explored to tackle this challenge, from traditional methods to current state-of-the-art deep learning techniques.

This section provides an overview of different approaches to paraphrase detection, focusing on the state-of-the-art methods that have garnered considerable attention and achieved remarkable results.

1. ***Traditional Rule-Based Methods***: Traditional paraphrase detection approaches rely on handcrafted rules, linguistic patterns, and syntactic analysis. These methods often use lexical and semantic features [1, 2, 3] to identify paraphrases. While these approaches are interpretable and can work well in some instances, they may struggle with the complexities and nuances of natural language.

2. ***Machine Learning-Based Methods***: Machine learning techniques, such as Support Vector Machines, Random Forests, and logistic regression, have been applied to paraphrase identification. These methods involve feature engineering, where various linguistic and statistical features are extracted from text pairs to train classifiers [4]. While effective to some extent, these approaches may require substantial feature engineering efforts and may not capture higher-level semantic relationships.

3. ***Transformer-Based Models***: Transformer-based models, such as BERT [5] (Bidirectional Encoder Representations from Transformers). Its variants have revolutionised paraphrase detection. These models are pre-trained on large text corpora and can capture contextual information effectively. Finetuning these models on paraphrase identification datasets has consistently achieved state-of-the-art results. BERT's ability to understand the context and nuances of language has made it a go-to choice for many researchers.

Among the diverse approaches to paraphrase detection, Transformer-based models, mainly variants of BERT, have demonstrated remarkable performance and have become the de facto state-of-the-art methods. These models, pre-trained on vast text corpora, capture local and global contextual information, allowing them to discern fine-grained semantic relationships between sentences. Additionally, advancements in finetuning techniques, data augmentation, and larger model architectures have further boosted their performance.

BERT-based models have achieved top-tier results in benchmark paraphrase identification datasets such as the Microsoft Research Paraphrase Corpus (MRPC) [6] and the Quora Question Pairs dataset [7]. Their ability to handle diverse linguistic expressions, syntactic variations, and domain-specific language makes them versatile and applicable to various paraphrase-detection tasks.

In the subsequent section of this paper, we will delve into a comprehensive comparison of Transformer-based deep learning methods, including BERT and its variants, to evaluate their performance and show their suitability for different paraphrase identification scenarios. We will also explore techniques for finetuning these models to achieve state-of-the-art results in paraphrase detection.

3. Transformer models overview

We selected a few different transformer models for the analysis. In this section, we want to go over each of them and present ideas behind each. We will start from the original transformer model, and all the rest will be listed alphabetically.

- BERT [5] short for "Bidirectional Encoder Representations from Transformers". BERT focuses on pre-training deep bidirectional representations by considering both left and right context across all layers. It can be finetuned with a single additional output layer and do question answering or language inference without any task-specific modifications to its architecture. Details: it uses absolute position embeddings; it was trained with masked language modeling and next sentence prediction objectives.
- ALBERT [8] short for "A lite BERT" was created to overcome the challenges of increasing model size in pre-training natural language representations, including GPU/TPU memory limitations, longer training times, and potential model degradation. To tackle these issues, the authors proposed two parameter-reduction techniques to reduce memory usage and speed up BERT training. Details: it uses absolute position embeddings and repeating layers, resulting in a small memory footprint (but compute costs remain the same).
- DistilBERT [9] short for "A distilled version of BERT". This model is trying to minimize params of large-scale pre-trained models like BERT. Their approach is different from previous research, because they tried to apply the distillation techniques [10] during the pre-training phase. As a result, they reduced the size by 40% while retaining 97% of its language understanding capabilities. Details: the authors introduce a triple loss function that combines language modelling, distillation, and cosine-distance losses; it has been trained to predict the exact probabilities of the larger model.
- BART [11] short for "Bidirectional and Auto-Regressive Transformer". It is a sequence-to-sequence model that uses a Transformer-based architecture, like BERT, for encoding but also includes a left-to-right decoder like GPT. The model was learning intending to reconstruct artificially corrupted text. It is capable of both text generation and comprehension tasks. Details: it uses absolute position embeddings.
- ELECTRA [12] short for "Efficiently Learning an Encoder that Classifies Token Replacements Accurately". This model uses a novel pre-training approach called "replaced token detection" instead of the masked language modelling used in BERT. This new pre-training task is efficient since it operates over all input tokens rather than just the masked subset. The efficiency gains are exceptionally prominent for smaller models, with an example of a model trained on one GPU for 4 days outperforming previous on the GLUE [13] natural language understanding benchmark. Details: no changes were made to the underlying model BERT (ELECTRA is the pre-training approach); a small masked language model was used to pretrain the model.
- MobileBERT [14] is a bidirectional transformer based on the BERT model but compressed and accelerated. The main idea behind this model is to reduce the size of big models to make it possible to use them on mobile devices. The resulting model is task-agnostic like the original BERT, allowing it to be applied to various NLP tasks through finetuning. Details: it uses absolute position embeddings; the teacher model was created on the base of the BERT_LARGE model, and knowledge transferring was applied.
- RoBERTa [15] short for "Robustly optimised BERT approach". This model explores how the performance of the original BERT can be optimised using more data and longer training times. It also proposes to change the training objective - remove the next sentence prediction part. Their improved pre-training procedure achieved state-of-the-art results even without multi-task finetuning. This approach also highlights the importance of hyperparameter selection and its significant impact on performance. Details: RoBERTa has the same architecture as BERT but uses a different pre-training scheme.
- I-BERT [16] short for "Integer BERT". This model addresses the resource-intensive nature of Transformer-based models like BERT and RoBERTa by introducing a novel quantisation approach. It is based on memory footprint reduction via lower bit precision representation [17]. The authors focus on integer-only quantisation throughout the inference process, eliminating the need for floating-point arithmetic. This approach relies on lightweight integer-only approximations for nonlinear operations

like GELU, SoftMax, and Layer Normalization. As a result, I-BERT achieves similar or slightly higher accuracy than full-precision models. Details: I-BERT has the same architecture model as BERT.

- DeBERTa [18] short for "Decoding-enhanced BERT with disentangled attention". The model builds upon RoBERTa and incorporates two innovative techniques. First, it employs a disentangled attention mechanism where each word is represented by two vectors, one encoding its content and the other its position. Attention weights between words are computed using disentangled matrices based on content and relative positions. Second, it uses an enhanced mask decoder to predict masked tokens during pre-training. Details: the DeBERTa model with 1.5 billion parameters surpasses the human performance on the SuperGLUE [19] benchmark.
- SqueezeBERT [20] is a model that is inspired by SqueezeNet [21] - computer vision model. SqueezeBERT is a bidirectional transformer model similar to BERT. Authors try to apply techniques usually used in computer vision models to minimise the model's size and improve its speed. Details: the critical distinction between SqueezeBERT and BERT lies in the architectural choice of using grouped convolutions instead of fully connected layers for the Q, K, V, and FFN layers.

All models described in this section will be analysed more deeply in the experiments section by their size, structure and performance on the paraphrase identification task.

Key metrics that will be used to compare models:

- Accuracy and F1 score.
- Size of the model.
- Number of sentences that the model can process in a second.

4. Dataset

Quite a few different corpora are commonly used for the paraphrase identification task. We will use Microsoft Research Paraphrase Corpus (MSRPC) [6]: is a corpus containing pairs of sentences for which it was manually indicated whether they are paraphrases. Dolan and Brockett created it at Microsoft Research, and has become one of the most common datasets for this task. It contains 5,801 sentence pairs taken from various sources, including news articles, encyclopaedia articles, and web pages. Multiple annotators were used for each pair of sentences to ensure the quality of the labels, and disagreements were resolved by majority vote.

5. Experiments

The main goal of this paper is to compare different models and their effectiveness to identify paraphrases. All finetuning for models described in this section was done using NVIDIA Tesla T4 with 16 GB GPU RAM and 50 GB System RAM using the Google Collab [22] platform. Given restrictions in computational resources, we limited amount of all models and used this experimental set-up:

- Each model was finetuned only 5 epochs.
- We used generic global parameters for finetuning for all models.
- Each model was finetuned five times, and we reported the mean and standard deviation. It was done to show how stable the model training is.

It is important to note that some of the models in the original papers achieved better results than reported here. There are a couple of reasons why this happened:

- Each model has different checkpoints depending on the aimed size of the final model and the vocabulary/training dataset used. In most cases, the best results were shown using large models we do not use because of computational restrictions.
- Given a model, one can always try improving the results using different hyperparameter optimisation techniques. We did not try to do such optimisation because of the high cost.

We used Hugging Face [23] library to access model checkpoints and to finetune those models on MRPC [6] dataset. All the models listed above are built using transformer architecture, but they have different configurations in terms of size, layers, size of the embeddings and vocabulary size.

We can group models based on their size into two groups – small and regular (we do not want to use the "large" word since we do not explore large transformer models in this paper due to their enormous size). Small models: ALBERT base, DistilBERT, Google Electra small discriminator, Google

MobileBERT, SqueezeBERT. Regular models: BERT base, Facebook BART base, Google Electra base discriminator, I-BERT RoBERTa base, Microsoft DeBERTa base, RoBERTa base.

Table 1
Mapping between model names and checkpoints used for finetuning

Model Name	Hugging Face model checkpoint name
ALBERT base	albert-base-v2
BERT base	bert-base-uncased
DistilBERT base	distilbert-base-uncased
Facebook BART base	facebook/bart-base
Google Electra base discriminator	google/electra-base-discriminator
Google Electra small discriminator	google/electra-small-discriminator
Google MobileBERT	google/mobilebert-uncased
I-BERT RoBERTa base	kssteven/ibert-roberta-base
Microsoft DeBERTa base	microsoft/deberta-base
RoBERTa base	roberta-base
SqueezeBERT	squeezebert/squeezebert-mnli-headless

Table 2
Configuration of transformer models analyzed in this paper

Model Name	Parameters	Layers	Hidden	Embedding	Vocabulary size
ALBERT base	11M	12	768	128	30k
BERT base	109M	12	768	768	30k
DistilBERT base	67M	6	768	768	30k
Facebook BART base	140M	12	768	768	50k
Google Electra base discriminator	109M	12	768	768	30K
Google Electra small discriminator	13.5M	12	256	128	30K
Google MobileBERT	24M	24	512	128	30K
I-BERT RoBERTa base	124M	12	768	768	50K
Microsoft DeBERTa base	139M	12	768	768	50k
RoBERTa base	124M	12	768	768	50K
SqueezeBERT	51M	12	768	768	30K

As mentioned above, all models were finetuned with these generic params:

- Batch size = 32.
- Fine tuning for 5 epochs.
- Learning rate = 0.00002.
- Weight decay = 0.01.
- AdamW (PyTorch) optimisation strategy.

We can observe that for small models, ALBERT has the best accuracy and F1 score performance. It is fascinating because compared to other small models, it has the smallest size – only 11M params. At the same time, ALBERT is the slowest one, which is expected because the authors of ALBERT tried to optimise the size and training time, not the speed.

- The fastest model is Google Electra small discriminator, compared to ALBERT, which has pretty good results, only two percentage points lower, with more than x10 speed.
- Let us look at regular models' performance.
- We can observe that the Microsoft DeBERTa base has the best accuracy and F1 score for regular-size models. This model is one of the biggest – 139M params. At the same time, the Microsoft DeBERTa base is one of the slowest ones.
- The fastest model is the I-BERT RoBERTa base, but it is fair to say that all regular-size models perform approximately the same in terms of samples per second.

- It is important to note that bigger models with the same architecture give better results – in this case, the Google Electra base model is better than the smaller version by four percentage points. The smaller model has almost eight times fewer parameters than the regular one but is five times faster.

Table 3
Small model's performance on the training and validation parts of the dataset

Model Name	Train Accuracy	Train F1 score	Validation Accuracy	Validation F1 score
ALBERT base	**93.92 ± 3.52**	**95.45 ± 2.67**	**88.53 ± 0.73**	**91.72 ± 0.49**
DistilBERT base	89.81 ± 0.35	92.39 ± 0.25	83.82 ± 0.99	88.66 ± 0.72
Google Electra small discriminator	85.09 ± 0.90	89.07 ± 0.68	83.38 ± 0.75	88.38 ± 0.43
Google MobileBERT	75.56 ± 5.03	82.85 ± 3.49	74.75 ± 5.58	82.62 ± 4.24
SqueezeBERT	93.02 ± 1.11	94.75 ± 0.84	88.19 ± 0.18	91.31 ± 0.15

Table 4
Small model's performance on the test part of the dataset

Model Name	Test Accuracy	Test F1 score	Samples per second
ALBERT base	**84.87 ± 1.30**	**88.64 ± 1.19**	164.88 ± 59.51
DistilBERT base	81.25 ± 0.55	86.19 ± 0.46	438.70 ± 3.01
Google Electra small discriminator	81.47 ± 0.58	86.50 ± 0.38	**1242.70 ± 39.59**
Google MobileBERT	72.44 ± 4.74	80.62 ± 3.71	526.57 ± 28.87
SqueezeBERT	84.58 ± 0.49	88.22 ± 0.40	419.06 ± 6.39

Table 5
Regular models performance on the train and validation parts of dataset

Model Name	Train Accuracy	Train F1 score	Validation Accuracy	Validation F1 score
BERT base	92.19 ± 1.34	94.19 ±0.97	82.65 ± 1.43	87.96 ± 0.83
Facebook BART base	93.32 ± 1.25	95.10 ± 0.89	86.76 ± 0.78	90.58 ± 0.68
Google Electra base discriminator	94.38 ± 0.66	**95.84 ± 0.51**	88.24 ± 0.64	91.62 ± 0.42
I-BERT RoBERTa base	92.55 ± 2.58	94.39 ± 2.00	87.79 ± 1.08	91.07 ±0.85
Microsoft DeBERTa base	**93.65 ± 1.70**	95.25 ±1.26	**88.33 ± 1.17**	**91.63 ±0.87**
RoBERTa base	92.67 ± 1.14	94.51 ± 0.88	88.19 ± 0.39	91.40 ± 0.25

Table 6
Regular models performance on the test part of dataset

Model Name	Test Accuracy	Test F1 score	Samples per second
BERT base	80.12 ± 2.05	85.50 ± 1.37	213.91± 3.09
Facebook BART base	85.92 ± 0.21	**89.74 ± 0.18**	156.89 ± 5.03
Google Electra base discriminator	85.31 ± 0.47	89.24 ± 0.38	207.08 ± 8.96
I-BERT RoBERTa base	**86.23 ± 0.50**	**89.63 ± 0.52**	**220.21 ± 0.91**
Microsoft DeBERTa base	**86.40 ± 0.48**	**89.86 ± 0.34**	169.76 ± 67.31

6. Small Experiment with Large Language Models

Large language models are powerful deep learning algorithms for various natural language processing tasks. They are based on transformer models and trained on massive datasets, enabling them to understand, translate, predict, or generate text and perform tasks beyond language understanding. These models have numerous parameters that serve as their knowledge base.

While those language models, in most cases, were created for text generation purposes, it is possible to use them to classify sentences. We wanted to explore whether LLM models can be used for paraphrase detection and how complicated it is to finetune them. We selected the Llama 2 [24] model released in July 2023 as an excellent example of LLM that produces state-of-the-art results on most benchmarks. It is important to note that this model is optimised for dialogue and text generation use cases. Llama 2 is available as a range of large language models (LLMs) with parameters ranging from 7 billion to 70 billion. For this analysis, we selected the model with 7 billion parameters.

Training and finetuning LLMs are technically and computationally challenging because of their colossal size. To overcome this issue, a few approaches were developed.

We used the Parameter-Efficient Finetuning (PEFT) [25] library for Llama 2 finetuning. We finetune only a few (extra) model parameters using this library, significantly decreasing computational and storage costs. This library supports a few methods of finetuning. We selected LoRa [26] because of its LLM application. The main advantage of LoRa is that it does not try to finetune the original model - it keeps pre-trained weights frozen. Instead, it creates two small matrices containing weight updates using low-rank decomposition. The finetuning task becomes a task to build those two small matrices.

Llama 2 finetuning details:

- Linear learning rate of $2 \times 10{-}5$
- Weight decay of 0.1
- Batch size of 16

LoRa Config:

- Task type – SEQ_CLS
- r = 16 (the rank of the update matrices, expressed in int. Lower rank results in smaller update matrices with fewer trainable parameters.)
- Lora alpha = 16
- Lora dropout = 0.1
- Bias = "all"

To finetune the model on the MRPC dataset, we used a single NVIDIA A100 (40GB - Google Collab).

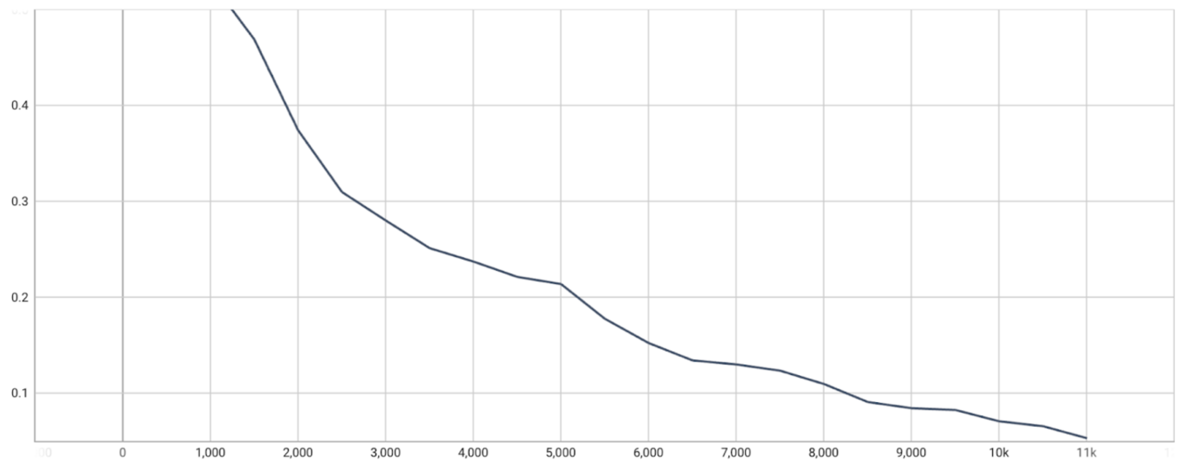

Figure 1: Graph of train loss during finetuning

We finetuned the model for 40 epochs, but from the validation accuracy graph, it is clear that the increase in accuracy during the last 20 is very minor.

To finetune Llama 2, we used a different, more powerful GPU, so we will not include a comparison of samples per second because it would not make sense.

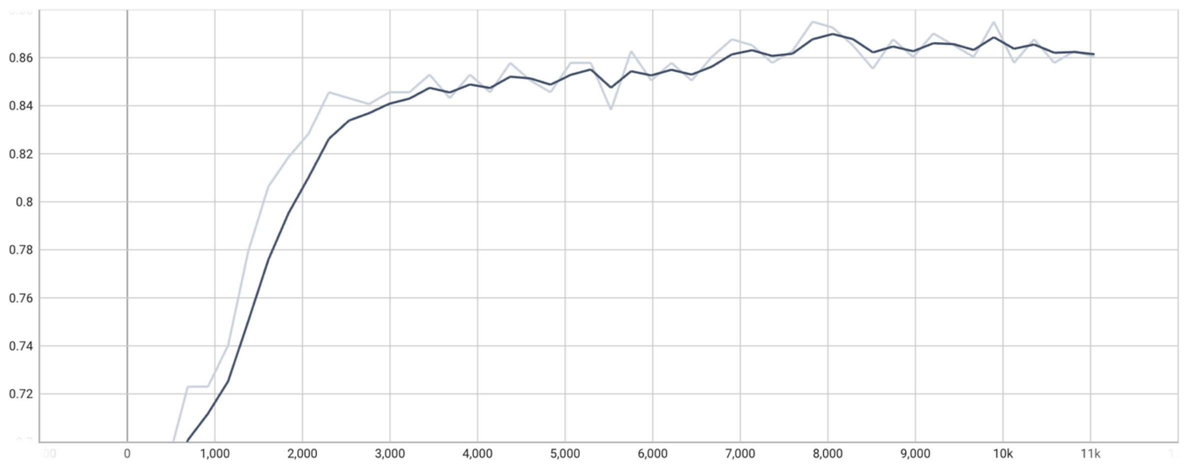

Figure 2: Graph of validation accuracy during finetuning

Table 7
LLM model performance on the test part of dataset after finetuning

Model Name	Test Accuracy	Test F1 score
Llama 2 7b	84.52	88.48

7. Conclusions

In this paper, we tried to analyse the state-of-the-art Transformer models. Almost all of them have the same bidirectional structure, and they all build on the BERT model foundation.

The field of model application is one of the main criteria for model selection. One should investigate smaller Transformer models like ALBERT or SqueezeBERT to perform relatively well on small devices. It should be used if accuracy is more critical - regular or large models will be your choice.

Large Language Models can also be used to detect paraphrases, but the cost and complexity of their finetuning are significantly larger compared with regular BERT-based models. We could not achieve better accuracy with the finetuned Llama 2 model. As a result, further improvements to the model performance can be made by exploring large models with even more parameters (Llama 2 70B model, for example) or including more sentence-aware context into Transformer architecture. A Good example is a DeBERTa model that tries to build a separate representation of a word and its position.

8. References

[1] C. Fellbaum, WordNet: An Electronic Lexical Database / Christiane Fellbaum, The MIT Press, 1998. doi:10.7551/mitpress/7287.001.0001.
[2] P. W. F. Thomas K Landauer and D. Laham, "An introduction to latent semantic analysis," Discourse Processes, vol. 25, no. 2–3, pp. 259–284, 1998, doi:10.1080/01638539809545028.
[3] C. Boonthum, "iSTART: Paraphrase Recognition," in Proceedings of the ACL Student Research Workshop, Jul. 2004, pp. 31–36. URL: https://aclanthology.org/P04-2006.
[4] N. Madnani, J. Tetreault, and M. Chodorow, "Re-examining Machine Translation Metrics for Paraphrase Identification," in Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Jun. 2012, pp. 182–190. [Online]. Available: https://aclanthology.org/N12-1019.
[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2019. URL: https://arxiv.org/abs/1810.04805. doi: 10.48550/arXiv.1810.04805.
[6] B. Dolan, C. Quirk, and C. Brockett, "Unsupervised Construction of Large Paraphrase Corpora: Exploiting Massively Parallel News Sources," in COLING 2004: Proceedings of the 20th

International Conference on Computational Linguistics, Aug. 2004, pp. 350–356. URL: https://aclanthology.org/C04-1051.
[7] Quora Duplicate Questions | Kaggle, 2023. URL: https://www.kaggle.com/aymenmouelhi/quora-duplicate-questions.
[8] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. 2020. URL: https://arxiv.org/abs/1909.11942. doi: 10.48550/arXiv.1909.11942.
[9] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. 2020. URL: https://arxiv.org/abs/1910.01108. doi: 10.48550/arXiv.1910.01108.
[10] G. Hinton, O. Vinyals, and J. Dean, Distilling the Knowledge in a Neural Network. 2015. URL: https://arxiv.org/abs/1503.02531. doi: 10.48550/arXiv.1503.02531.
[11] M. Lewis et al., BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension. 2019. URL: https://arxiv.org/abs/1910.13461. doi: 10.48550/arXiv.1910.13461.
[12] K. Clark, M.-T. Luong, Q. V. Le, and C. D. Manning, ELECTRA: Pre-training Text Encoders as Discriminators Rather Than Generators. 2020. URL: https://arxiv.org/abs/2003.10555. doi: 10.48550/arXiv.2003.10555.
[13] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman, “GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding,” in Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, Nov. 2018, pp. 353–355. doi: 10.18653/v1/W18-5446.
[14] Z. Sun, H. Yu, X. Song, R. Liu, Y. Yang, and D. Zhou, MobileBERT: a Compact Task-Agnostic BERT for Resource-Limited Devices. 2020. URL: https://arxiv.org/abs/2004.02984. doi: 10.48550/arXiv.2004.02984.
[15] Y. Liu et al., RoBERTa: A Robustly Optimized BERT Pretraining Approach. 2019. URL: https://arxiv.org/abs/1907.11692. doi: 10.48550/arXiv.1907.11692.
[16] S. Kim, A. Gholami, Z. Yao, M. W. Mahoney, and K. Keutzer, I-BERT: Integer-only BERT Quantization. 2021. URL: https://arxiv.org/abs/2101.01321. doi: 10.48550/arXiv.2101.01321.
[17] Dong, Z., Yao, Z., Gholami, A., Mahoney, M. W., and Keutzer, K. HAWQ: Hessian aware quantisation of neural networks with mixed-precision. In Proceedings of the IEEE International Conference on Computer Vision, pp. 293–302, 2019.
[18] P. He, X. Liu, J. Gao, and W. Chen, DeBERTa: Decoding-enhanced BERT with Disentangled Attention. 2021. URL: https://arxiv.org/abs/2006.03654. doi: 10.48550/arXiv.2006.03654.
[19] A. Wang et al., SuperGLUE: A Stickier Benchmark for General-Purpose Language Understanding Systems. 2020. URL: https://arxiv.org/abs/1905.00537. doi: 10.48550/arXiv.1905.00537
[20] F. N. Iandola, A. E. Shaw, R. Krishna, and K. W. Keutzer, SqueezeBERT: What can computer vision teach NLP about efficient neural networks? 2020. URL: https://arxiv.org/abs/2006.11316. doi: 10.48550/arXiv.2006.11316.
[21] F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size. 2016. URL: https://arxiv.org/abs/1602.07360. doi: 10.48550/arXiv.1602.07360.
[22] E. Bisong, "Google colaboratory" in Building Machine Learning and Deep Learning Models on Google Cloud Platform, Berkeley, CA, USA:Apress, pp. 59-64, 2019.
[23] T. Wolf et al., "Transformers: State-of-the-Art Natural Language Processing," in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Oct. 2020, pp. 38–45. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.6.
[24] H. Touvron et al., Llama 2: Open Foundation and Finetuned Chat Models. 2023. URL: https://arxiv.org/pdf/2307.09288.pdf. doi:10.48550/arXiv.2307.09288.
[25] PEFT: State-of-the-art Parameter-Efficient Finetuning, version 0.4, 2023. URL:https://github.com/huggingface/peft.
[26] E. J. Hu et al., LoRA: Low-Rank Adaptation of Large Language Models. 2021. URL: https://arxiv.org/abs/2106.09685. doi: 10.48550/arXiv.2106.09685.