Surface-Based Retrieval Reduces Perplexity of Retrieval-Augmented Language Models

Citation for the original published paper (version of record):

Doostmohammadi, E., Norlund, T., Kuhlmann, M. et al (2023). Surface-Based Retrieval Reduces Perplexity of Retrieval-Augmented Language Models. Association for Computational Linguistics . Annual Meeting Conference Proceedings, 2: 521-529.
http://dx.doi.org/10.18653/v1/2023.acl-short.45

Surface-Based Retrieval Reduces Perplexity of Retrieval-Augmented Language Models

Ehsan Doostmohammadi[1*] **Tobias Norlund**[2,4] **Marco Kuhlmann**[1] **Richard Johansson**[2,3]
[1] Linköping University [2] Chalmers University of Technology
[3] University of Gothenburg [4] Recorded Future

Abstract

Augmenting language models with a retrieval mechanism has been shown to significantly improve their performance while keeping the number of parameters low. Retrieval-augmented models commonly rely on a semantic retrieval mechanism based on the similarity between dense representations of the query chunk and potential neighbors. In this paper, we study the state-of-the-art RETRO model and observe that its performance gain is better explained by surface-level similarities, such as token overlap. Inspired by this, we replace the semantic retrieval in RETRO with a surface-level method based on BM25, obtaining a significant reduction in perplexity. As full BM25 retrieval can be computationally costly for large datasets, we also apply it in a re-ranking scenario, gaining part of the perplexity reduction with minimal computational overhead.

1 Introduction

The introduction of the Transformer architecture (Vaswani et al., 2017) has led to a performance boost in language modeling (see, e.g., Brown et al. 2020), but also to a steep increase of computational cost, as the number of parameters and data points is constantly growing. In reaction to this development, there has recently been a surge in work on retrieval-augmented language models (Izacard and Grave, 2021a; Li et al., 2022), which shows that enabling models to retrieve context from large corpora results in lower perplexity and better accuracy in downstream tasks such as question answering, while at the same time using considerably fewer parameters. In this paper, we specifically focus on the Retrieval-Enhanced Transformer architecture (RETRO; Borgeaud et al., 2022).

By augmenting a language model with a retrieval mechanism, RETRO, like similar architectures, tries to decouple *memorization* of the training data from the additional *generalization* that comes with increasing the number of parameters. In RETRO, when a chunk of text (a sequence of tokens) has been generated, a dense representation of this chunk is used to retrieve the most similar neighboring chunks from a large retrieval set, based on their L2 distance. Having the previously generated chunks and their nearest neighbors in the retrieval set, the auto-regressive language model has now access to an extended context when predicting the next chunk. The informativeness of this context depends on the effectiveness of the retrieval method.

Borgeaud et al. (2022) note that part of RETRO's performance can be attributed to the token overlap between the generated chunks and the retrieval set. Our starting point in this paper is the observation that the performance gain is actually *better* explained by such surface-level similarities than by the L2 distance between the dense representations that RETRO uses for retrieval. This is in line with recent work by Norlund et al. (2023), who show that the reduction in loss observed in RETRO "almost exclusively" stems from such overlap rather than more sophisticated generalization. Based on these findings, we replace the semantic retrieval method in RETRO with one based on BM25 (Robertson et al., 1995), a surface-level measure. Our results show that retrieving nearest neighbors using BM25 during inference leads to a 13.6% lower perplexity, compared to dense retrieval based on sentence transformers (ST) (Reimers and Gurevych, 2019), a model trained to represent the semantic similarity between sentences.[1]

Finding the exact neighbors with BM25 is costly on large retrieval sets and might not meet the speed requirements of all applications of retrieval-augmented language models. We therefore explore a hybrid approach where we first retrieve approximate neighbors using ST representations and then

*Correspondence to ehsan.doostmohammadi@liu.se.

[1]The code and the data for this study can be accessed at github.com/edoost/retro_bm25.

Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics
Volume 2: Short Papers, pages 521–529
July 9-14, 2023 ©2023 Association for Computational Linguistics

re-rank them using BM25. We show that this approach yields 24.7% of the perplexity reduction we get with BM25-based retrieval, with only minimal computational overhead.

2 Method

We experiment with RETRO (Borgeaud et al., 2022) as a state-of-the-art retrieval-augmented language model.

2.1 Model

RETRO is very similar to a standard auto-regressive language model such as T5 (Raffel et al., 2020), the main differences being the introduction of the retrieval mechanism and how the retrieved neighbors are used for language modeling.

Nearest Neighbor Retrieval In RETRO, all textual data is stored and used in chunks of 64 tokens. When the model has generated a chunk C_u, it retrieves the k nearest neighbors $N_{1:k}$ to that chunk, together with the chunks $F_{1:k}$ following these neighbor chunks in the retrieval data. It then generates the next chunk C_{u+1} conditioned on the retrieved chunk pairs. Retrieval uses the squared L2 distance on a dense representation (DR) of chunks:

$$d(C_u, N_i) = \|DR(C_u) - DR(N_i)\|_2^2$$

This leaves us with

$$\textsc{Ret}(C_u) = ([N_u^1; F_u^1], \dots, [N_u^k; F_u^k])$$

as the retrieved neighbors that the model receives as additional context when generating the next chunk. The likelihood of the first chunk (C_1) does not depend on any neighbors; the model has access to no external context when generating that chunk. During training and perplexity evaluation, the retrieval process is filtered such that chunks originating from the same source document as the training sequence are never considered as neighbors.

Integration of the Neighbors RETRO improves auto-regressive language modeling by conditioning the next token prediction on the retrieved chunks of text. This means that the probability of generating the next token x_{t+1} depends not only on the previously generated tokens $x_{1:t}$ but also on the retrieved neighbors of the previously generated chunks, as well as their following chunks:

$$P\left(x_{t+1} \,|\, x_{1:t}, \textsc{Ret}(C_1), \dots, \textsc{Ret}(C_{u-1}); \theta\right)$$

When generating the next token, the neighbors as well as the current chunk C_u are passed through a Transformer encoder. In the decoder, cross-attention is over the output of that encoder and the concatenation of the intermediary embeddings of the last few tokens in the previous chunk C_{u-1} and the already generated tokens in C_u, a mechanism called *chunked cross-attention*. For more details, see Borgeaud et al. (2022).

Implementation Details As an official implementation of RETRO is not publicly available, we draw upon the implementation in Norlund et al. (2023), which is based on the description in Borgeaud et al. (2022). Our implementation deviates only in that (1) we use learnable relative positional biases as in T5 (Raffel et al., 2020), with a bucket for each unique relative position; (2) instead of BERT (Devlin et al., 2019), we use the pre-trained sentence transformers (ST) (Reimers and Gurevych, 2019) model to embed the chunks for the offline retrieval. ST is preferable over BERT, as it is trained for the task of similarity search, and produces embeddings of lower dimensionality, which makes it more efficient. We use PyTorch (Paszke et al., 2019) and PyTorch Lightning for distributed training. For the tokenization, we use the pre-trained T5 tokenizer (HuggingFace). For retrieving approximate neighbors, we use faiss (Johnson et al., 2019), which performs efficient similarity search between dense representations with GPU support for faster indexing and retrieval.

2.2 Data

Borgeaud et al. (2022) use the *MassiveText* dataset (Rae et al., 2021) for both training and retrieval. As this dataset is not publicly available, we set out to replicate it using open sources. *MassiveText* consists of multilingual text data in five categories: Wikipedia articles, books, GitHub code, news, and common crawl web data. We use *Pile* (Gao et al., 2021) and *RealNews* (Zellers et al., 2019) to build a large dataset resembling *MassiveText*'s composition. The new dataset (see Norlund et al. (2023) for details) consists of 36M documents containing 52B tokens. For *Pile*, we keep the training and validation splits, while for *RealNews*, we use the full training set but downsample the validation set to 16,400 news articles to match the proportions of the categories in *Pile*. For details on the deduplication process, we refer to Gao et al. (2021) and Zellers et al. (2019).

2.3 Training

We use our dataset to train a RETRO model with approximately 630M parameters. For more details refer to Norlund et al. (2023). During training, we retrieve from the training set; during validation, we retrieve from the union of the training and validation sets. We train the model on sequences truncated to 1,024 tokens. The chunk size is 64, as in Borgeaud et al. (2022), and the number of retrieved neighbors is $k = 2$ for training and validation. We train the model for 140k training steps with a batch size of 16, taking seven days on 16 A100 GPUs. This means that we use 6% of the training data during training, not including the retrieved neighbors. As our optimizer, we use Adam (Kingma and Ba, 2015) with a fixed learning rate of $1\text{e}{-4}$.

3 A Study on Correlations

We experiment with two settings: RETRO[ON], the language model with retrieval enabled, and RETRO[OFF], where there are no chunk cross-attention layers and therefore no retrieval, leaving us with a decoder-only language model. As shown by Borgeaud et al. (2022), the RETRO[ON] model performs better when it can exploit an overlap between the generated text and the retrieved neighbor. This is more apparent in text categories with higher token overlap, such as GitHub. The studies in the RETRO paper also show that allowing more overlap when deduplicating the data results in a lower bits-per-byte (BPB[2]). Norlund et al. (2023) take this further to show even minimal overlap results in significant loss reduction, demonstrating the large extent RETRO relies on surface-level similarities. These findings lead us to hypothesize that having a retrieval method that can find the highest overlapping neighbors will yield lower perplexity (PPL). Because BERT, ST and similar deep representations of sentences do not always capture surface-level similarities, we set out to investigate where performance gains come from.

To this end, we measure how the PPL difference (ΔPPL) between RETRO[ON] and RETRO[OFF] for the current chunk (C_u, $u \geq 2$) correlates with (1) squared L2 distance between the ST embeddings of C_u and $\text{RET}(C_{u-1})$ (ST), and (2) unigram token overlap, based on T5 tokenization, between C_u and $\text{RET}(C_{u-1})$. The results, reported in Table 1, show a considerably stronger correlation between ΔPPL and unigram token overlap (measure 2) than between ΔPPL and L2 distance (measure 1). The trend is similar between Spearman and Pearson correlation coefficients.

[2] BPB $= (L_T/L_B)\mathcal{L}/ln(2)$, where L_T and L_B are the lengths of the validation set in T5 tokens and UTF-8 encoded bytes, respectively, and $\mathcal{L}$ stands for log likelihood loss. L_T/L_B is 0.258415 for our validation set.

X	Y	ρ	r
L2^2 (ST)	ΔPPL	0.328	0.134
token overlap	ΔPPL	0.494	0.415
L2^2 (ST)	token overlap	0.464	0.515

Table 1: Spearman ρ and Pearson r between variables X and Y. L2^2 (ST) is the (negative) squared L2 distance between the ST embeddings of $\text{RET}(C_{u-1})$ and C_u; token overlap is the unigram token overlap between these two chunks; and $\Delta\text{PPL} = \text{PPL}_{\text{RETRO[OFF]}} - \text{PPL}_{\text{RETRO[ON]}}$ for the chunk C_u.

4 Changing the Retrieval Method

As the results from the previous section show a stronger correlation between performance gain and surface-level similarity than ST similarity, we experiment with a retrieval method based on BM25.

4.1 BM25

Okapi BM25, introduced by Robertson et al. (1995), is a bag-of-words retrieval method based on tf–idf scores and some free parameters. These parameters are k_1, which normalizes the term frequency, and b, which controls how much the length of a document would affect the term frequency values. We use Pyserini (Lin et al., 2021), a Python interface to Lucene's BM25 implementation. We build the BM25 index on the training set and leave the free parameters at their default values ($k_1 = 0.9$, $b = 0.4$). These values were also shown to perform the best by Karpukhin et al. (2020a). Using Lucene's Analyzer pipeline[3] results in more than 50M unique words for our corpus. We instead use the T5 tokenizer from Hugging Face Transformers (Wolf et al., 2020) and limit our vocabulary to 32k words for the reranking experiments.

4.2 Retrieving with BM25

We use the model described in Section 2.3 and change the retrieval method only at inference time to retrieve better neighbors. The results can be found in Table 2. The perplexity is 14.00 for

[3] Lucene Analyzers (Lucene) are used to extract index terms from text, which includes tokenization and preprocessing.

Model	PPL	BPB
RETRO[OFF]	14.00	0.984
RETRO[ON]-ST	10.87	0.889
RETRO[ON]-ST + BM25 reranking	10.46	0.875
RETRO[ON]-BM25	8.95	0.817

Table 2: PPL and BPB for various retrieval settings on the validation set. The basic RETRO model is the same for all rows.

RETRO[OFF] and 10.87 for RETRO[ON] with ST retrieval (RETRO[ON]-ST), corresponding to a 22.3% reduction in PPL. Replacing the retrieval method with BM25 (RETRO[ON]-BM25) gives an additional 13.7% reduction, which is 61.3% of the initial drop. For comparability with Borgeaud et al. (2022), we also report BPB. The results show that using neighbors with more surface-level similarity to the generated chunk is a solid method for leveraging the retrieval mechanism to reduce the perplexity. If the retrieval augmentation is meant to act as an external memory, or to offload memorization from the model (Borgeaud et al., 2022), then BM25 is a more suitable method to achieve this goal.

4.3 Reranking

While the performance gain is significant, finding the *exact* neighbors using BM25 could be costly, depending on the size of the datasets. On the other hand, `faiss` provides an efficient similarity search for dense vectors to find the *approximate* neighbors. Therefore, if enough of the BM25-retrieved neighbors could be found among top-k `faiss`-retrieved ones, with an efficient reranking, we could expect at least part of the performance gain with minimal computational overhead, as long as k is not significantly large. To find an optimal k, we first need to know how many of BM25 neighbors could be found in top-k `faiss`-retrieved chunks.

Looking at the `faiss`-retrieved neighbors, we see that of top-4 BM25-retrieved neighbors, 17.6% appear in top-100 `faiss`-retrieved chunks, while the overlap is 22.1% for top-1000. We decide to continue our experiment with top-1000 neighbors, but it is obvious that one could get an even higher overlap with a higher k, with diminishing returns. The results in Table 2 show that with the proposed reranking, RETRO[ON]-ST could achieve 21.3% of the PPL reduction of RETRO[ON]-BM25 compared to RETRO[ON]-ST. The reranking results are interesting not only due to their practical implications but also as an analysis revealing the limited number of high-quality neighbors that can be retrieved using semantic retrieval, even in situations where a large k is feasible.

5 Related Work

Augmenting language models with mechanisms that help them incorporate larger contexts has been approached extensively in different forms, such as Guu et al. (2018)'s retrieve-and-edit approach to reduce the PPL in language generation, and Asai et al. (2020) that make use of lexical overlap to improve the performance in question answering. While retrieval-augmentation has been used with different objectives in mind, such as language modeling (Khandelwal et al., 2020; Wu et al., 2022) and machine translation (Khandelwal et al., 2021), question answering has been the application to attract the most interest (Guu et al., 2020; Karpukhin et al., 2020b; Izacard and Grave, 2021b).

An extensive study was performed by Izacard et al. (2022), showing that while we get performance gains using retrieval augmentation, training the retrieval part of the model would yield even more benefits. RETRO (Borgeaud et al., 2022), on the other hand, aims at scaling such language models and therefore opts for keeping the retriever frozen, showing substantial PPL reduction with increasing either the number of language model parameters or the size of retrieval set.

Among the more recent work, Xu et al. (2023) found that training using approximate neighbors resulted in a 2.6% decrease in perplexity. This suggests that non-exact neighbors may have a regularization effect, leading to improved generalization ability. Additionally, Ram et al. (2023) report a drop in perplexity using BM25 over BERT retrieval using in-context retrieval-augmented language models.

6 Conclusions and Future Work

In this paper, we study the source of performance gains in RETRO, which could be generalized to similar retrieval-augmented language models. After observing that the PPL drop correlates more strongly with surface-level overlap between the query and the retrieved text, we replace the retrieval method with BM25, and observe a significant drop in PPL, which confirms us in the findings of the correlation study. This is also an interesting insight as to how these models work, which could be leveraged for performance gain in tasks like question answering where model relies on retrieving facts. In the end, we also conduct an analysis to find out how much `BM25` neighbors overlap with those retrieved using ST. The results show that while `faiss` is able to find some of the neighbors with high token overlap, the majority of them remain unretrieved. This is however, enough to gain part of the loss reduction achieved with a pure `BM25` retrieval system.

The proposed methods could also be used during training. By retrieving more overlapping neighbors during training, the process of guiding the model to use retrieved neighbors for language modeling could be done more efficiently. This is particularly relevant when augmenting an already trained language model with a retrieval mechanism. As reported by Borgeaud et al. (2022), retrieval augmentation results in a larger drop in BPB as the number of model parameters and the size of retrieval data grow. This calls for more efficient methods based on surface-level similarities if we wish to exploit this potential. Furthermore, the retrieval system in RETRO is based on semantic retrieval, the model seems to rely more on surface-level similarities. This could affect the generalizability capabilities of such models, which necessitates further investigations. Lastly, we only evaluate our modified RETRO model on language modeling. It would be interesting to know the impacts of `BM25` retrieval on downstream tasks where retrieval is of use.

Limitations

We only experiment with one type of retrieval-augmented language models, i.e., RETRO. However, the ways the other models retrieve neighbors and integrate them are not so much different to affect the results in this paper. The experiments in this paper are done with a small size RETRO model and data compared to the sizes considered by Borgeaud et al. (2022), due to computational limitations. According to the same authors, however, the gains should be constant with the increase of the model and retrieval set size. The larger models are mainly different in their behavior when there is no overlap. However, this should not affect the *copying* tendency of these models tremendously, as it is still the easiest way to generate the next token. It is also worth noting that RETRO[OFF], while not using retrieval at test time, is still *trained* using retrieval – so it is not a complete retrieval-free model. The results presented by Borgeaud et al. (2022) however, show that RETRO[OFF] is on a par with their retrieval-free baseline in terms of BPB. Finally, we note that our evaluations have only considered the perplexity under teacher forcing, and we have not investigated the behavior of the model in free-form generation or with any kind of fine-tuning.

Acknowledgements

This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. The computations were enabled by resources provided by the National Academic Infrastructure for Supercomputing in Sweden (NAISS) at Alvis partially funded by the Swedish Research Council through grant agreement no. 2022-06725, and by the Berzelius resources provided by the Knut and Alice Wallenberg Foundation at the National Supercomputer Center.

References

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *International Conference on Learning Representations*.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, Diego De Las Casas, Aurelia Guy, Jacob Menick, Roman Ring, Tom Hennigan, Saffron Huang, Loren Maggiore, Chris Jones, Albin Cassirer, Andy Brock, Michela Paganini, Geoffrey Irving, Oriol Vinyals, Simon Osindero, Karen Simonyan, Jack Rae, Erich Elsen, and Laurent Sifre. 2022. Improving language models by retrieving from trillions of tokens. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 2206–2240. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2021. The pile: An 800gb dataset of diverse text for language modeling. *CoRR*, abs/2101.00027.

Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. 2018. Generating sentences by editing prototypes. *Transactions of the Association for Computational Linguistics*, 6:437–450.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. 2020. Realm: Retrieval-augmented language model pre-training. In *International Conference on Machine Learning*, pages 3929–3938. PMLR.

HuggingFace. Huggingface T5.

Gautier Izacard and Edouard Grave. 2021a. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Gautier Izacard and Edouard Grave. 2021b. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022. Few-shot learning with retrieval augmented language models. *arXiv preprint arXiv:2208.03299*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020a. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020b. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Urvashi Khandelwal, Angela Fan, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2021. Nearest neighbor machine translation. In *International Conference on Learning Representations*.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through memorization: Nearest neighbor language models. In *International Conference on Learning Representations (ICLR)*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR (Poster)*.

Huayang Li, Yixuan Su, Deng Cai, Yan Wang, and Lemao Liu. 2022. A survey on retrieval-augmented text generation. arXiv preprint 2202.01110.

Jimmy Lin, Xueguang Ma, Sheng-Chieh Lin, Jheng-Hong Yang, Ronak Pradeep, and Rodrigo Nogueira. 2021. Pyserini: A python toolkit for reproducible information retrieval research with sparse and dense representations. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '21, page 2356–2362, New York, NY, USA. Association for Computing Machinery.

Apache Lucene. Lucene analyzer.

Tobias Norlund, Ehsan Doostmohammadi, Richard Johansson, and Marco Kuhlmann. 2023. On the generalization ability of retrieval-enhanced transformers. In *Findings of the Association for Computational Linguistics: EACL 2023*, pages 1485–1493, Dubrovnik, Croatia. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Jack W. Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, H. Francis Song, John

Aslanides, Sarah Henderson, Roman Ring, Susannah Young, Eliza Rutherford, Tom Hennigan, Jacob Menick, Albin Cassirer, Richard Powell, George van den Driessche, Lisa Anne Hendricks, Maribeth Rauh, Po-Sen Huang, Amelia Glaese, Johannes Welbl, Sumanth Dathathri, Saffron Huang, Jonathan Uesato, John Mellor, Irina Higgins, Antonia Creswell, Nat McAleese, Amy Wu, Erich Elsen, Siddhant M. Jayakumar, Elena Buchatskaya, David Budden, Esme Sutherland, Karen Simonyan, Michela Paganini, Laurent Sifre, Lena Martens, Xiang Lorraine Li, Adhiguna Kuncoro, Aida Nematzadeh, Elena Gribovskaya, Domenic Donato, Angeliki Lazaridou, Arthur Mensch, Jean-Baptiste Lespiau, Maria Tsimpoukelli, Nikolai Grigorev, Doug Fritz, Thibault Sottiaux, Mantas Pajarskas, Toby Pohlen, Zhitao Gong, Daniel Toyama, Cyprien de Masson d'Autume, Yujia Li, Tayfun Terzi, Vladimir Mikulik, Igor Babuschkin, Aidan Clark, Diego de Las Casas, Aurelia Guy, Chris Jones, James Bradbury, Matthew J. Johnson, Blake A. Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Edward Lockhart, Simon Osindero, Laura Rimell, Chris Dyer, Oriol Vinyals, Kareem Ayoub, Jeff Stanway, Lorrayne Bennett, Demis Hassabis, Koray Kavukcuoglu, and Geoffrey Irving. 2021. Scaling language models: Methods, analysis & insights from training gopher. *CoRR*, abs/2112.11446.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Ori Ram, Yoav Levine, Itay Dalmedigos, Dor Muhlgay, Amnon Shashua, Kevin Leyton-Brown, and Yoav Shoham. 2023. In-context retrieval-augmented language models. *arXiv preprint arXiv:2302.00083*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using siamese BERT-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Stephen Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford. 1995. Okapi at TREC-3. In *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 109–126. Gaithersburg, MD: NIST.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Yuhuai Wu, Markus Norman Rabe, DeLesley Hutchins, and Christian Szegedy. 2022. Memorizing transformers. In *International Conference on Learning Representations*.

Frank F Xu, Uri Alon, and Graham Neubig. 2023. Why do nearest neighbor language models work? *arXiv preprint arXiv:2301.02828*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
It's the last section and it is unnumbered.

☑ A2. Did you discuss any potential risks of your work?
We mention that relying on surface-level similarities could affect the generalizability capabilities of such models, which necessitates further investigations.

☑ A3. Do the abstract and introduction summarize the paper's main claims?
They could be found in the abstract, section 1 (introduction), and even the other sections.

☑ A4. Have you used AI writing assistants when working on this paper?
We used Grammarly to a limited extent.

B ☑ Did you use or create scientific artifacts?

It's all over the paper, but mainly section 2.

☑ B1. Did you cite the creators of artifacts you used?
It's all over the paper, but mainly section 2.

☒ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
One should consult to the main papers for that.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. It is not taken care of by us, but by the authors of those datasets.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Left blank.

C ☑ Did you run computational experiments?

Sections 2, 3, and 4.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
Section 2.3.

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
Section 2.

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Some of them are not applicable, but the rest are discussed in Section 2.

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
Yes. We will even publish our code later for absolute transparency.

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.