On the Generalization Ability of Retrieval-Enhanced Transformers

Tobias Norlund[1,4*] **Ehsan Doostmohammadi**[2] **Richard Johansson**[1,3] **Marco Kuhlmann**[2]

[1] Chalmers University of Technology [2] Linköping University
[3] University of Gothenburg [4] Recorded Future

Abstract

Recent work on the Retrieval-Enhanced Transformer (RETRO) model has shown that off-loading memory from trainable weights to a retrieval database can significantly improve language modeling and match the performance of non-retrieval models that are an order of magnitude larger in size. It has been suggested that at least some of this performance gain is due to non-trivial generalization based on both model weights and retrieval. In this paper, we try to better understand the relative contributions of these two components. We find that the performance gains from retrieval largely originate from overlapping tokens between the database and the test data, suggesting less non-trivial generalization than previously assumed. More generally, our results point to the challenges of evaluating the generalization of retrieval-augmented language models such as RETRO, as even limited token overlap may significantly decrease test-time loss. We release our code and model at https://github.com/TobiasNorlund/retro

1 Introduction

Large-scale generative language models have shown promising results toward creating a general-purpose foundation for many natural language applications. While sheer scale-up has resulted in better language modeling performance, the immense costs are an inhibiting factor towards further improvements (Sharir et al., 2020).

Recent work on retrieval-augmented language models, such as the Retrieval-Enhanced Transformer (RETRO; Borgeaud et al., 2022), suggests that *memory* can be effectively off-loaded from the model parameters to an external database. In RETRO, the information retrieved from the database is used to augment the context from which the model predicts new tokens, reducing the need to memorize this information in the model parameters. This opens up for smaller language models with retained performance. Specifically, Borgeaud et al. (2022) report that, with a large enough retrieval database, RETRO can achieve a performance comparable to GPT-3 (Brown et al., 2020) and Jurassic-1 (Lieber et al., 2021) on the Pile (Gao et al., 2020), at only 4% of the parameters. Similarly, RETRO achieves significantly lower bits-per-byte performance compared to a baseline of the same size without retrieval.

Borgeaud et al. (2022) conclude that RETRO has the capacity for non-trivial generalization based on both the model parameters and the retrieval database, even though they find that part of the performance gains can be attributed to lexical overlap between retrieval and test data. In this work, we want to better understand the nature and magnitude of this effect. Our findings indicate that performance gains[1] originate *almost exclusively* from RETRO's ability to copy tokens verbatim from retrieved data, effectively exploiting any (small or large) overlap between training and test data. This suggests that the ability of RETRO to fuse retrieved and in-parameter information may be more limited than previously assumed.

2 Method

To investigate gains from retrieval, we re-implement the RETRO model described by Borgeaud et al. (2022) (with a few deviations; see below). We present the model here in brevity.

2.1 The RETRO Model

RETRO is an autoregressive language model trained with the next-token prediction objective, where the prediction probability is conditioned on additional context retrieved from a database.

Retrieval Retrieval occurs at the granularity of contiguous token chunks with a fixed size m. More specifically, assume that RETRO has already generated a sequence of tokens $x_{1:t}$. Each token x_i

[*] Corresponding author, tobiasno@chalmers.se

[1] Results on RETRO were originally reported in bits-per-byte, while we report results in loss.

Findings of the Association for Computational Linguistics: EACL 2023, pages 1485–1493
May 2-6, 2023 ©2023 Association for Computational Linguistics

belongs to a chunk $C_{c(i)}$, where $c(i) = \lceil i/m \rceil$. The probability of the next token x_{t+1} depends on the previously generated tokens and the context retrieved from the previously seen chunks:

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

Database Retro's database takes the form of a key–value storage $R(N) \mapsto [N, F]$, where N is a chunk from one of the indexed documents, F is the immediately following chunk, and the key $R(N) \in \mathbb{R}^d$ is the embedding of N according to some embedding model R. This database is used to retrieve the k nearest neighbors of a chunk C, based on the embedding $R(C)$:

$$\textsc{Ret}(C) = ([N^1, F^1], \ldots, [N^k, F^k])$$

Architecture Retro is based on the original Transformer architecture (Vaswani et al., 2017). Chunk neighbors are encoded by the encoder and attended to by the decoder. Due to the quadratic complexity in self-attention, each neighbor is encoded separately; all representations are then concatenated and made available to the decoder (Izacard and Grave, 2021). The original decoder is modified such that for the prediction of token x_{t+1}, cross-attention (CA) can only attend to the neighbor representations retrieved based on the previous chunk $C_{c(t+1)-1}$. This is called *chunked cross-attention* (CCA). Furthermore, the encoder is modified to include a restricted form of cross-attention to the decoder. Specifically, the encoder CA attends to the decoder hidden states immediately before the first CCA. We refer to Borgeaud et al. (2022) for more details.

Implementation Details For tokenizing documents, we use the pre-trained T5 tokenizer. The retrieval was performed using approximate nearest neighbor search with the high-performant `faiss` library (Johnson et al., 2019). We implement Retro in PyTorch (Paszke et al., 2019) and use PyTorch Lightning for distributing the training and validation data across GPUs and compute nodes. Our implementation deviates from that of Borgeaud et al. (2022) only in that we

- use learnable relative positional biases as in T5 (Raffel et al., 2020), with a bucket for each unique relative position; and
- instantiate the chunk embedding model R by a pre-trained Sentence-BERT (SB) model (Reimers and Gurevych, 2019) instead of Bert. We deemed SB to be preferable over Bert as it is smaller (i.e. cheaper to compute) and produces embeddings of lower dimensionality (i.e. saves disk space).

2.2 Dataset

Borgeaud et al. (2022) used a multi-lingual version of *MassiveText* (Rae et al., 2021) for both training and retrieval data. To replicate the English portion of this data, we sought open-source alternatives. *MassiveText* comprises text from the categories web text, news, code, books, and Wikipedia. By pooling matching categories from Pile (Gao et al., 2020) and adding the RealNews dataset (Zellers et al., 2019), we obtain a large dataset composed of all five categories, consisting of 36M documents and 52B tokens. We keep the training/validation splits from the Pile categories. For RealNews, we use the provided training set and a subsample of 16,400 documents from the validation set. The full description of our dataset is shown in Table 1.

2.3 Model Training

For our experiments, we train a Retro model that resembles the 425M model[2] in Borgeaud et al. (2022), as shown in Table 2. We train and test on our open-source version of *MassiveText* as described in Section 2.2. During training, we retrieve neighbors from the training set, while at validation time, we retrieve from the union of training and validation sets. We filter out neighbors that originate from the same source document as the query chunk. Each model is trained on sequences of no more than 1,024 tokens; longer sequences are truncated. We use a chunk size of 64 and retrieve two neighbors during both training and validation. We train the model for 140k training steps with a batch size of 16. This means that only 6% of the training documents are actually used during training, excluding retrieved neighbors. We use the Adam optimizer with a fixed learning rate of 1e−4.

3 Experiments

Borgeaud et al. (2022) observed that retrieval increases language modeling performance. To validate this observation, we compare two configurations of our model: Retro[on], where we enable retrieval, and Retro[off], where we remove the CCA layers, thereby reducing Retro to a standard decoder-only language model. As we can see in

[2] The 425M parameters exclude embeddings.

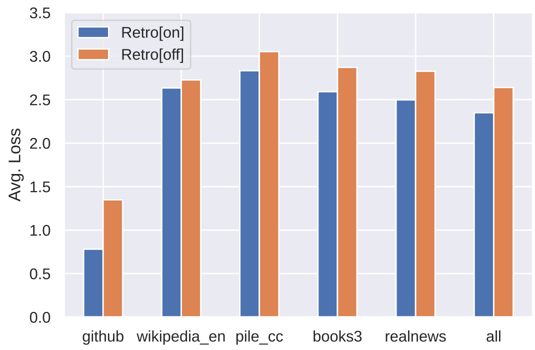

Figure 1: Comparing loss on validation set categories, when using retrieval vs. no retrieval.

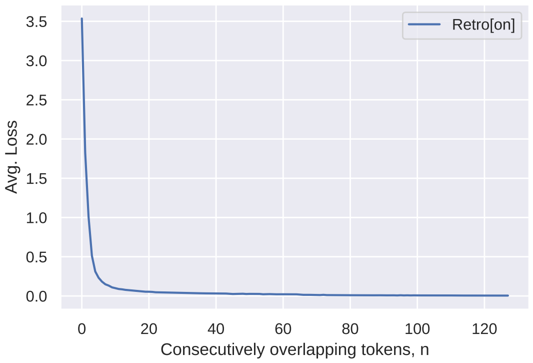

Figure 2: Average loss from RETRO[ON] over tokens in $\Phi(n)$. Note the drastic decrease with increasing overlap.

Figure 1, retrieval reduces the loss across all data categories, and with 11% across the full validation set. GitHub data has the lowest validation loss among all categories and is also where we see the largest reduction in loss, at 42%. Wikipedia sees the smallest reduction in loss, at only 3%. A closer comparison to the results from Borgeaud et al. (2022) is available in Appendix D.

3.1 Loss per Degree of Overlap

As Borgeaud et al. (2022) note, retrieval-based models such as RETRO may more easily exploit evaluation dataset leakage. To quantify how much of the positive effect of retrieval on language modeling performance can be attributed to such leakage, the authors computed bits-per-byte (bpb) for evaluation chunks with different amounts of consecutive token overlap relative to their retrieved neighbors. This analysis showed that, while the positive effect of retrieval decreased with smaller overlaps, it was still significant at overlap levels of at most 8 contiguous tokens, which the authors considered small enough to conclude that while RETRO actually learns to *generalize* from retrieval data, not merely copy-and-paste it. Here we investigate the hypothesis that the bpb reductions observed by Borgeaud et al. (2022) *are localized exclusively in the overlapping tokens*. If this was true, it would challenge the conclusion that RETRO learns non-trivial generalizations based on retrieval data.

To test our hypothesis, we sort the validation set tokens into buckets based on their leftward overlap. Specifically, we put a token x_i into a bucket $\Phi(n)$, where n is the largest number such that x_i and the $n-1$ tokens preceding it consecutively overlap with some neighboring chunk in $\text{RET}(C_{c(i)-1})$. For example, the bucket $\Phi(1)$ contains all tokens x_i for which the unigram x_i appears in some neighbor, but not the bigram $x_{i-1}x_i$; the bucket $\Phi(2)$ contains all x_i for which $x_{i-1}x_i$ overlaps but not $x_{i-2}x_{i-1}x_i$, and so on. As a special case, the bucket $\Phi(0)$ contains all tokens that do not overlap with any of its neighbors. This includes all tokens that occur in a first chunk C_1, which lacks neighbors.

In Figure 2 we plot the average loss per bucket,

$$\frac{1}{|\Phi(n)|} \sum_{x_i \in \Phi(n)} \mathcal{L}_{x_i}^{\text{RETRO[ON]}}, \qquad (1)$$

as a function of n. Here, $\mathcal{L}_{x_i}^{\text{RETRO[ON]}}$ is the loss when predicting token x_i using RETRO[ON][3]. We see that the loss drastically decreases as the consecutive overlap increases. For example, at an overlap of $n = 5$ tokens, the loss is only 6% of the loss for non-overlapping tokens. This suggests that RETRO enters "copy mode" when the previous tokens overlap with those from a neighbor.

3.2 Loss Reductions per Degree of Overlap

For a more detailed analysis of the effect of overlap on predictive performance, we look at the token-specific loss differences between the two configurations RETRO[OFF] and RETRO[ON]:

$$\Delta\mathcal{L}_{x_i} = \mathcal{L}_{x_i}^{\text{RETRO[OFF]}} - \mathcal{L}_{x_i}^{\text{RETRO[ON]}}$$

Note that a loss difference $\Delta\mathcal{L}_{x_i}$ is positive if the access to the retrieved context reduces the token-specific loss for x_i. The overall reduction in loss visible in Figure 1 is the average of the loss differences across all tokens in the validation data. By aggregating loss differences per bucket $\Phi(n)$, we get a fine-grained picture of how the reductions

[3] The sizes of each bucket (accumulated over the validation data) are shown in the appendix, Figure 4.

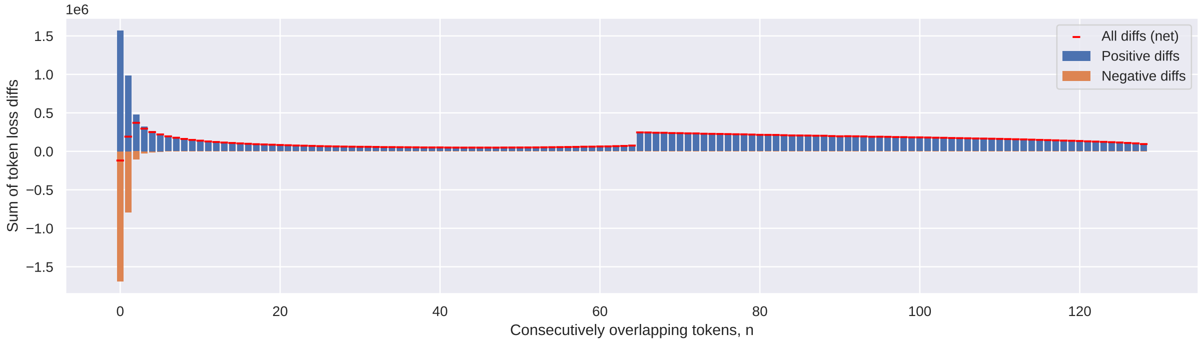

Figure 3: Token-specific loss differences, as distributed over different degrees of overlap. *Positive diffs* shows the sum of all positive loss differences, $\sum_{x_i \in \Phi(n)} \max(0, \Delta\mathcal{L}_{x_i})$, and *Negative diffs* shows the sum of negative loss differences, $\sum_{x_i \in \Phi(n)} \min(0, \Delta\mathcal{L}_{x_i})$. *All diffs* shows the total sum. We see that the vast majority of loss reductions comes from overlapping tokens, e.g. $n > 0$.

are distributed with respect to different degrees of consecutive overlap. This is illustrated in Figure 3.

For non-overlapping tokens ($n = 0$), we can see that there are both positive and negative differences, with a small negative net. For all overlapping tokens ($n > 0$), the net differences are positive, and for buckets with 3 or more overlapping tokens, there are almost no negative differences at all.[4] This shows that the largest share of all loss reductions originates from tokens that are consecutively overlapping in neighbors. Interestingly, the net differences are positive even for very small degrees of overlap. Borgeaud et al. (2022) considered reductions in bits-per-byte from chunks with up to 8 consecutively overlapping tokens as evidence of a non-trivial generalization capacity. However, our results suggest that even a small number of overlapping tokens may cause a large reduction in loss, which we take as an argument against this conclusion.

4 Related Work

Equipping language models with a retrievable external memory has been extensively studied (Guu et al., 2020; Karpukhin et al., 2020; Lewis et al., 2020; Izacard and Grave, 2021; Li et al., 2022). Explicitly leveraging the training data through retrieval to reduce perplexity is proposed in kNN-LM (Khandelwal et al., 2020). kNN-LM matches the leftward context with the leftward context of all training data tokens, and explicitly interpolates between generating and copying the next token. A recent study analyzes kNN-LM to better understand the causes of performance gains (Xu et al., 2023). Similar to our findings in RETRO, lexical overlap has also been found to play a significant role in explaining retrieval performance gains in kNN-LM as well (Drozdov et al., 2022). The idea of kNN-LM is extended in SPALM (Yogatama et al., 2021) to instead learn a gating function that facilitates more dynamic interpolation.

In both kNN-LM and SPALM, retrieval is incorporated at the top of the network. This might induce a bias towards surface-level rather than semantic augmentation. In contrast, retrieval in RETRO is incorporated in lower layers of the network, which opens up for more sophisticated integration of the retrieved information. Our results suggest, however, that retrieval in RETRO also contributes at the surface rather than at the semantic level, similar to the previous works.

5 Conclusions and Future Work

The capacity of language models for generalization is often measured intrinsically using perplexity, loss or bits-per-byte on held-out validation data. Low perplexity language models perform well as few-shot learners on many downstream tasks due to their capacity to both memorize and non-trivially combine textual information from many sources (Brown et al., 2020; Rae et al., 2021; Lieber et al., 2021; Chowdhery et al., 2022). The hope is that we can externalize memory to reduce the footprints of language models without reducing generalization and downstream task performance.

Our results show that the low loss in RETRO almost exclusively originates from tokens overlapping between retrieval and validation data, rather than from more sophisticated generalization. To better

[4] We note a sudden increase in accumulated loss difference for $n > 64$ which is expected considering the way in which we construct the buckets; see Appendix C for more details.

understand this effect, it would be interesting to modify the retrieval component and deliver semantically similar but lexically different context during training. If the retrieved context is uninformative, the model will learn to ignore it, but if the context is too specific (e.g. literal overlap) the model will learn to copy. By better balancing between these two modes, models may become better at utilizing retrieved information at a deeper and more generalizable level.

Limitations

We have made our best effort in trying to reproduce the model and results of Borgeaud et al. (2022). Nonetheless, our experiments were performed on one of the smaller model sizes and with a dataset that is only ~2.5% of their size (52 billion vs. 2 trillion tokens). This was due to computational constraints and lack of larger open datasets. However, as was also shown by Borgeaud et al. (2022), the performance gain of retrieval is constant with respect to model size. We speculate that larger RETRO models mostly improve with respect to loss on tokens that are not overlapping, which would not change our conclusions here.

One noteworthy limitation of our work is the fact that we compare to a non-retrieval baseline (RETRO[OFF]) that was trained with access to retrieved context. We were not able to train a separate non-retrieval baseline due to computational constraints, but note that the bits-per-byte results of RETRO[OFF] and the baseline in Borgeaud et al. (2022) were close to identical.

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 Centre.

References

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.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Ben Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier Garcia, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Diaz, Orhan Firat, Michele Catasta, Jason Wei, Kathy Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. PaLM: Scaling language modeling with pathways.

Andrew Drozdov, Shufan Wang, Razieh Rahimi, Andrew McCallum, Hamed Zamani, and Mohit Iyyer. 2022. You can't pick your neighbors, or can you? when and how to rely on retrieval in the k nn-lm. *arXiv preprint arXiv:2210.15859*.

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. 2020. The Pile: An 800gb dataset of diverse text for language modeling. *arXiv preprint arXiv:2101.00027*.

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.

Gautier Izacard and Edouard Grave. 2021. 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.

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. 2020. 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, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2020. Generalization through Memorization: Nearest Neighbor Language Models. In *International Conference on Learning Representations (ICLR)*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

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

Opher Lieber, Or Sharir, Barak Lenz, and Yoav Shoham. 2021. Jurassic-1: Technical details and evaluation. Technical report, AI21 Labs.

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, 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 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 Johnson, Blake Hechtman, Laura Weidinger, Iason Gabriel, William Isaac, Ed 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.

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.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2022. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(1).

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 and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992, Hong Kong, China. Association for Computational Linguistics.

Or Sharir, Barak Peleg, and Yoav Shoham. 2020. The cost of training NLP models: A concise overview. arXiv preprint 2004.08900.

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

Frank F. Xu, Uri Alon, and Graham Neubig. 2023. Why do nearest neighbor language models work?

Dani Yogatama, Cyprien de Masson d'Autume, and Lingpeng Kong. 2021. Adaptive semiparametric language models. *Transactions of the Association for Computational Linguistics*, 9:362–373.

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.

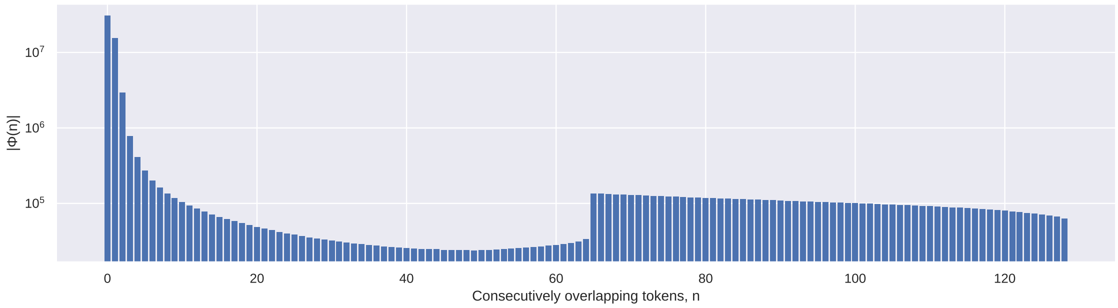

Figure 4: Number of validation set tokens in each bucket $\Phi(n)$. Since the neighbors have a maximal length of 128 tokens, this is also the longest possible overlap n.

		Documents	Chunks	Tokens
Training	Pile-CC	15,728k	269M	16.7B
	Wikipedia En	5,082k	61M	3.8B
	GitHub	5,417k	181M	11.4B
	Books3	83k	191M	12.2B
	RealNews	9,360k	130M	8.0B
	Total	35,670k	833M	52.2B
Validation	Pile-CC	52.8k	900.4k	56.0M
	Wikipedia En	17.4k	215.9k	13.3M
	GitHub	18.3k	598.4k	37.7M
	Books3	0.3k	727.6k	46.5M
	RealNews	16.4k	234.5k	14.5M
	Total	105.3k	2,676.8k	168.0M

Table 1: Statistics for our MassiveOpenText dataset. We use the web text, Wikipedia, GitHub and Books3 corpora from the Pile, and news text from RealNews.

A MassiveOpenText statistics

Statistics on the number of documents, chunks and tokens for each split and text category are shown in Table 1.

B Retro model details

We show hyperparameters of our Retro model in Table 2.

	Param	
Encoder	Num layers	2
	Num heads	14
	Hidden size	896
	FFN	3584
	CA layers	[2]
Decoder	Num layers	12
	Num heads	12
	Hidden size	1536
	FFN	6144
	CCA layers	[6,9,12]

Table 2: Hyperparameters of our trained Retro model.

C Consecutively overlapping tokens

As explained in Section 3.1, we sort validation set tokens into buckets denoted $\Phi(n)$ depending on the longest overlapping leftward context.

In Figure 4 we show the number of tokens in each bucket. We note a big "jump" from $n = 64$ to $n = 65$, which can be explained by the following rationale. A neighbor $[N, F]$ to a chunk C_i is retrieved based on the similarity between C_i and N. In the case where both $C_i = N$ and $C_{i+1} = F$, tokens in C_{i+1} will be put into $\Phi(n)$ with $n = 65, \ldots, 128$. The jump in Figure 4 indicates such duplicates are common in our data.

D Model validation

As we aim to reproduce the 425M model trained in Borgeaud et al. (2022), it is important to validate that the implementations are equivalent and that their evaluation results are comparable. However, evaluations of the 425M model in Borgeaud et al. (2022) on the Pile are not available, making it hard to make direct comparisons. Borgeaud et al.

(2022) report evaluation results on the C4 (Raffel et al., 2022) dataset, with various sizes of retrieval datasets. For their setup with 36B retrieval tokens, which is the most similar to our own retrieval size, they report that bits-per-byte is reduced by ~ 2% (from 0.92 to 0.90) when using retrieval. That could be compared to our results on Pile-CC, as both datasets originate from Common Crawl. In our experiments, loss is reduced by 7% (from 3.05 to 2.83) on Pile-CC.

Evaluations on the Pile in Borgeaud et al. (2022) are only reported for their largest model (7B params) and largest retrieval set (2T tokens). For example, on Pile–GitHub their reduction is ~53% whereas our reduction is 42%.

While these numbers are not directly comparable, we believe they indicate that our reimplementation of the Retro model is working as expected.