RecombiText: Compositional Data Augmentation for Enhancing LLM Pre-Training Datasets in Low-Resource Scenarios

Alexander Tampier[*], **Lukas Thoma**[*○•], **Loris Schoenegger**[*•], **Benjamin Roth**[*△]
[*]Faculty of Computer Science, University of Vienna, Vienna, Austria
[○]Department of Linguistics, University of Vienna, Vienna, Austria
[•]UniVie Doctoral School Computer Science, Vienna, Austria
[△]Faculty of Philological and Cultural Studies, University of Vienna, Vienna, Austria
Correspondence: alexander.tampier@univie.ac.at

Abstract

We introduce RecombiText Augmentation (RTA), a novel purely statistical NLP method for compositional data augmentation for data-efficient LLM pre-training in low-resource scenarios. RTA identifies lexically and semantically similar sentences within the corpus and generates synthetic sentence pairs from them while preserving underlying patterns from the corpus. We pre-train GPT-2 and RoBERTa language models on a domain-specific, low-resource corpus of 10 million words, with different proportions of augmented data. We compare our RTA-augmented model variants to a baseline model trained on the full original dataset. Zero-shot results show that the language models pre-trained on synthetic data improve in entity tracking, self-paced reading, and morphological generalization benchmarks. In other tasks, the performance is comparable to the baseline model. We demonstrate that it is possible to expand low-resource datasets by two- to four-fold without compromising benchmark performance, solely through statistical processing of the available data.

1 Introduction

Large language models (LLMs), such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), GPT-3 (Brown et al., 2020), and Chinchilla (Hoffmann et al., 2022), are large-scale language models based on the Transformer architecture from Vaswani et al. (2017) that have achieved remarkable performance across various Natural Language Processing (NLP) tasks. However, success is based on extensive training data, often hundreds of billions of words. For example, GPT-3 (Brown et al., 2020) was trained with 570 GB of text after filtering. This typically results in high computational costs, as well as a dependency on vast amounts of available training data in the respective language or domain. However, large amounts of data are not always available in all languages or domains, which limits the applicability of current language model pre-training for low-resource scenarios (Charpentier et al., 2025; Hedderich et al., 2020). In contrast, human language acquisition is far more efficient. For example, children can fully learn a language by the time they reach puberty, even though they are only exposed to 3 to 11 million words per year (Warstadt and Bowman, 2022; Warstadt et al., 2025).

This discrepancy underscores the limitations of current LLM pre-training and emphasizes the need for data-efficient pre-training in low-resource scenarios. Data augmentation (DA) offers a promising solution for efficiently utilizing available training data by generating synthetic examples from existing datasets (Warstadt et al., 2025; Hu et al., 2024). While DA is well-explored in computer vision and downstream NLP tasks (Feng et al., 2021), its application for LLM pre-training in low-resource scenarios is less explored (Warstadt et al., 2025; Hu et al., 2024). Recent efforts show that DA methods can improve model performance. However, many rely on generative models or auxiliary text data beyond the limited domain training set (Theodoropoulos et al., 2024; Haga et al., 2024; Edman et al., 2024; Zhang et al., 2023; Lyman and Hepner, 2024), thereby limiting their suitability for scenarios in which such auxiliary data is not available.

To address this gap, we propose RecombiText Augmentation (RTA). This novel statistical compositional DA method leverages information retrieval techniques and combines lexical and semantic similarity by utilizing a one-point crossover, a concept inspired by genetic algorithms (Goldberg, 1989). Since RTA relies exclusively on the corpus itself, it is independent of models trained on additional text and is therefore ideal for truly low-resource scenarios. RTA generates synthetic sentences in four corpus-dependent phases:

i. Generating corpus-based embeddings
ii. Selecting matching candidates

Proceedings of the First BabyLM Workshop, pages 548–565

iii. Identifying pivot elements with sliding context windows
iv. Applying a one-point crossover to create synthetic sentence pairs.

Experiments on a 10-million-word corpus, acting as a domain-specific low-resource scenario, show that the language models that were trained with RTA-augmented datasets improve the most for performances in zero-shot Entity Tracking (Kim and Schuster, 2023; Charpentier et al., 2025), Self-paced Reading (de Varda et al., 2024), and morphological generalization tasks (WUGs) compared to the baseline.

We demonstrate that purely statistical compositional data augmentation can effectively enhance the language model pre-training dataset in low-resource scenarios without incurring any significant losses in evaluation.

2 Related Work

Good-Enough Compositional Data Augmentation (GECA) (Andreas, 2020) is another compositional data augmentation algorithm for language modeling. GECA identifies and swaps substitutable fragments from sentences that share similar local environments to generate synthetic compositional examples, thereby enabling compositional text recombination without relying on models trained on additional text. In contrast to GECA, our RTA method focuses on lexical and semantical similarities.

Related data augmentation methods for efficient language modeling in low-resource scenarios include using word embeddings from Mikolov et al. (2013) for word substitutions within the training data and external treebanks to ensure grammatical correctness, as proposed by Lyman and Hepner (2024). Haga et al. (2024) generates artificial variation sets that mimic children's speech to produce paraphrased utterances using a pre-trained language model trained on extensive external data. Theodoropoulos et al. (2024) trains a decoder on subsets of the TinyStories dataset (Eldan and Li, 2023) to generate synthetic examples. While these methods enhance performance in low-resource scenarios, many rely on models that were trained on additional text (Lyman and Hepner, 2024; Haga et al., 2024; Edman et al., 2024; Zhang et al., 2023; Theodoropoulos et al., 2024).

3 RecombiText Augmentation

RTA relies exclusively on information from the training corpus, addressing the limitations of methods that depend on resources trained with additional text. Our method generates synthetic sentence pairs based on lexically and semantically similar sentences from the corpus.

3.1 Intuition

RTA is based on the idea of ad-hoc information retrieval (IR), where a user sends a query in natural language and receives relevant results from a collection of documents (Jurafsky and Martin, 2019). Furthermore, it is assumed that sentences that share similar local environments can be swapped to some extent. Based on this, we select a reference sentence that represents the query and perform a hybrid search for lexically and semantically similar sentences within the corpus. We then re-rank them, similar to hybrid search techniques. We borrow the idea from genetic algorithms (Goldberg, 1989) and use a one-point crossover to cut and swap the sentence fragments. To determine the intersection of the reference and candidate sentences, we must assess the pivot elements in each respective sentence. For this purpose, a sliding context window is employed based on semantic similarity and term importance within both sentences. Figure 1 shows the intuition behind the RTA algorithm to create new sentence pairs.

3.2 Algorithmic Formulation

The RTA algorithm operates in four phases using corpus statistics and accessing only the available training data. It combines information retrieval techniques for candidate matching, statistical embeddings for semantic similarity, and a genetic algorithm-inspired crossover for augmentation. The process is divided into four main phases: (i) Word and Sentence Embeddings, (ii) Candidate Selection, (iii) Context Window, and (iv) Crossover Operation. The source code is publicly available [1].

Word and Sentence Embeddings Word embeddings with Global Vectors for Word Representation (GloVe) (Pennington et al., 2014), and sentence embeddings with unsupervised smoothed inverse frequency (uSIF) (Ethayarajh, 2018) are created beforehand. Both methods use the available corpus training data. The sentence embeddings are

[1] https://github.com/luciendgolden/RTA

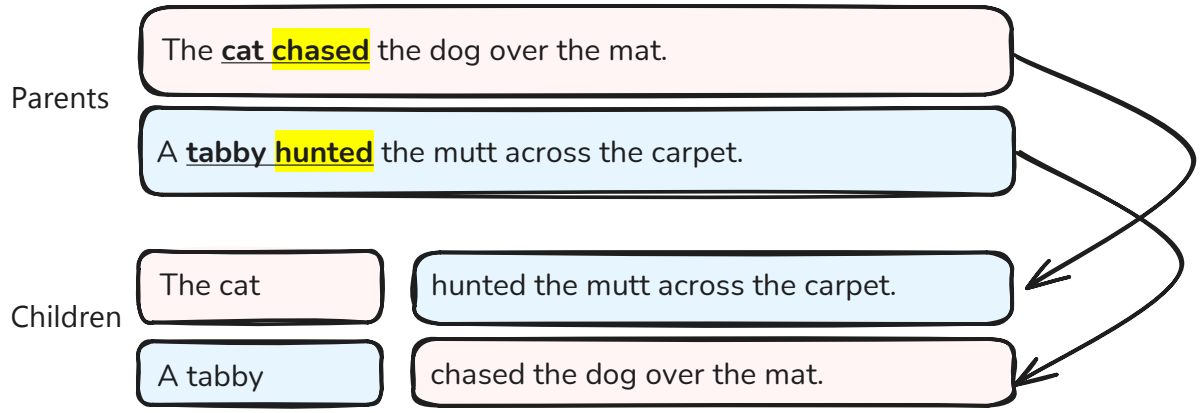

Figure 1: The idea is to create new synthetic sentences based on lexical and semantic similarity from parts of sentences using a one-point crossover. This involves searching for a semantic context window that maximizes the IDF-weighted cosine similarity between a reference and candidate sentence, to determine the pivot elements at which the sentences can be cut and swapped. In this example, the semantic context window size is $W = 2$

stored in the IR system for efficient use of semantic search.

Candidate Selection To generate the synthetic data, a query sentence q is randomly selected from the specified corpus. For a query sentence q, lexically and semantically similar candidates are retrieved. Lexical matches are determined via Best Matching 25 (BM25) (Robertson and Zaragoza, 2009) and give ranking $\mathcal{R}_{BM25}$, which is an extended TF-IDF variant where TF-IDF is the product of two terms, namely the term frequency (TF) and the inverse document frequency (IDF). Semantic matches are determined via k-Nearest Neighbors (kNN) with cosine similarity on sentence embeddings and give ranking $\mathcal{R}_{kNN}$, which is used for the semantic classification of the candidates. These rankings are fused using Reciprocal Rank Fusion (RRF) (Cormack et al., 2009) with constant k_{RRF} to produce $\mathcal{R}_{RRF}$, forming the result set $\mathcal{R}_{RRF} = \{d \mid d \in \mathcal{R}_{BM25}\} \cup \{d \mid d \in \mathcal{R}_{kNN}\}$ where d represents the document sentence. A candidate m is then selected from the top k_{top} via top-k sampling with softmax probabilities modulated by temperature T. From the amount of synthetic text generated $\mathcal{D}_{aug}$ relative to the proportional baseline dataset $\mathcal{D}_{base}$ we get an augmentation ratio r, which defines the maximum frequency of use for q and m. Therefore, in the dataset variant that combines 1 million words from the baseline dataset with 9 million words from synthetic text, $\mathcal{D}_{base1M} + \mathcal{D}_{aug9M}$, each q and m sentence can be selected up to 9 times, in the $\mathcal{D}_{base2.5M} + \mathcal{D}_{aug7.5M}$ variant, up to 3 times, and in all other variants only once.

Context Window To ensure linguistic consistency of the generated sentences after the crossover operation, pivot elements are identified where the query sentence and the selected candidate exhibit high semantic similarity in terms of context. This is achieved using a sliding context window for q, m where sequences of the two sentences are compared based on the importance and semantic equivalence. To avoid placing too much importance on words that are often insignificant, the respective words within the context window are weighted by their IDF values. The context window is defined by a fixed size W where W determines the number of words within the window. Within the respective windows W_q, W_m, the words with the highest similarity and importance are searched for.

For each possible starting position in both sentences, a window of size W is defined, which starts at positions i in Q and j in M where Q and M are tokenized versions of the query sentence q and candidate sentence m. Therefore, we define a context similarity $S_{\text{ctx}}(i, j)$ between the corresponding windows W_q, W_m. Equation 1 shows the context similarity calculation.

$$S_{\text{ctx}}(i, j) = \frac{\sum(\cos_k \times \text{idf}_k)}{\sum \text{idf}_k} \tag{1}$$

where k is each token pair within the windows W_q and W_m and $\cos_k$ is the respective cosine similarity between the word embeddings in W_q and W_m. A context window is only accepted if it is above the threshold τ_{window}. Within the best window, find the pair with the maximum similarity $\cos_k \times \text{idf}_k$, which acts as pivot elements for the

following crossover operation.

Crossover Operation One-point crossover is applied at pivot elements to produce new synthetic sentence pairs $\tilde{q}, \tilde{m}$ where the splitting takes place at pivot elements. The algorithm also defines and considers various edge cases, such as when identical sentences are produced. If such edge cases occur, additional retries are executed. If the algorithm fails several times, a new attempt with a randomly new query sentence q is executed.

4 Experimental Setup

The experiments are carried out in four stages. Firstly, the generation of the different training data variants. Secondly, evaluating the quality of augmented training data. Thirdly, the pre-training of the language model, and fourthly, the evaluation of the language models using zero-shot and fine-tuning tasks.

4.1 Data Generation

The experiments use the 10-million-word corpus from the official `strict-small` 2025 BabyLM Challenge by Charpentier et al. (2025). We denote the $\approx 10M$-word internal baseline dataset from Table 4 as $\mathcal{D}_{base}$, with $\mathcal{D}_{baseX}$ representing the sampled proportion of X million words; $\mathcal{D}_{augY}$ as the Y million words generated via RTA; and the combined dataset as $\mathcal{D}_{baseX} + \mathcal{D}_{augY}$. To create the different proportions from the baseline dataset $\mathcal{D}_{base}$, the sample chunks and split script from Warstadt et al. (2025) is used. The custom Python script from Timiryasov and Tastet (2023) is used for preprocessing (see Appendix A).

GloVe (Pennington et al., 2014) is employed to generate word embeddings and trained on the respective baseline proportion for each variant. Sentence embeddings are created corpus-wide with uSIF (Ethayarajh, 2018) when running the algorithm. RTA was applied to the baseline dataset $\mathcal{D}_{\text{base}}$ to generate synthetic sentences in $\mathcal{D}_{\text{aug}}$. The augmented sentences are inserted adjacent to the reference sentences, based on the findings from Haga et al. (2024).

4.2 Data Quality

Perplexity (PPL) is used to evaluate the quality of the generated data $\mathcal{D}_{\text{aug}}$. The evaluation of data quality is only a diagnostic metric and independent of the data generation process. The perplexity is compared for all augmented sentences in $\mathcal{D}_{\text{aug}}$ with the perplexity values for all reference sentences from $\mathcal{D}_{\text{base}}$ to determine whether the augmented examples preserve the underlying linguistic fluency. For this, the official pre-trained openai-community/gpt2 (Radford et al., 2019) from Huggingface is utilized, as it has been trained on large amounts of data, enabling us to determine how surprised the model is by the augmented sentences. The RTA-augmented datasets used in our experiments exhibit an approximately three times higher average PPL (53.09 ± 4.11) compared to the proportional baseline datasets (17.83 ± 0.21). Self-BLEU scores are used to measure the diversity within the corpus and show a modest increase for augmented datasets ($14.57\% \pm 4.45\%$) compared to the proportional baseline datasets ($8.41\% \pm 0.31\%$).

4.3 Language Model Pre-training

For language model pre-training, we utilize a decoder-based language model, GPT-2 Radford et al. (2019), which was trained using next-token prediction. Additionally, we employ an encoder-based language model, RoBERTa (Liu et al., 2019), which was trained using masked language modeling with a custom checkpoint strategy. Training is quantified by the total number of whitespace-separated input words, which must not exceed the threshold of 100 million input words in total for all models trained on text (Charpentier et al., 2025) (for details see Appendix A).

4.4 Language Model Evaluation

For the performance evaluation of the pre-trained language model, the official 2025 BabyLM Challenge evaluation pipeline from Charpentier et al. (2025) is utilized, which encompasses domain-specific zero-shot, fast zero-shot, and fine-tuning tasks.

For zero-shot evaluation, the tasks are the Benchmark of Linguistic Minimal Pairs (BLiMP) and the BLiMP Supplement (Warstadt et al., 2020), which assess grammatical ability. Elements of World Knowledge (EWoK) (Ivanova et al., 2024) evaluates the ability of world modeling for language models. Eye Tracking (Reading ET) and Self-paced Reading (Reading SPR) (de Varda et al., 2024) are behavioral paradigms used to measure word-by-word language processing. Entity Tracking (ENT) (Kim and Schuster, 2023; Charpentier et al., 2025) tests how well language models can follow changes to objects such as a book or an apple within a story or conversation. WUGs (Weissweiler

et al., 2023; Hofmann et al., 2025) tests the linguistic generalization of language models. COMPS (Misra et al., 2022) assesses commonsense property inheritance, where properties of broader categories apply to more specific subcategories. Age of Acquisition (AoA) (Chang and Bergen, 2022) tracks word surprisal over training checkpoints to derive learning curves correlated with human acquisition ages. The fast zero-shot tasks utilize a smaller set of evaluation examples from the zero-shot tasks and evaluate the performance of the language models at specific checkpoints. The fine-tuning tasks utilize selected tasks from the GLUE (Wang et al., 2018) and SuperGLUE (Wang et al., 2019) datasets, which reflect language understanding tasks.

5 Results and Discussion

The quantitative results are presented briefly below, followed by a brief insight into the qualitative results.

Decoder For our GPT-2 model (Table 1), the dataset variant that uses only 25% of the original data (25/75) achieves the highest performance for entity tracking (Kim and Schuster, 2023; Charpentier et al., 2025). The variant with 50% of the original data (50/50) achieves the highest performance for BLiMP Supplement (Warstadt et al., 2020), EWoK (Ivanova et al., 2024), WUGs Past, Reading SPR and ET (de Varda et al., 2024), and GLUE (Wang et al., 2018, 2019) over the baseline. The 75% original data variant (75/25) achieves the best results for BLiMP (Warstadt et al., 2020). The baseline showed a negligible correlation with human language acquisition, while the augmented datasets led to moderate improvements. The variant (75/25) achieved the highest correlation. However, no variant achieved statistical significance of $p < 0.05$ (see Appendix A).

Encoder For our RoBERTa model (Table 2), the dataset variant that uses only 25% of the original data (25/75) achieves the highest average performance for entity tracking (Kim and Schuster, 2023; Charpentier et al., 2025), WUGs Past (Weissweiler et al., 2023; Hofmann et al., 2025), and COMPS (Misra et al., 2022) benchmarks over the baseline. The detailed fine-tuning results show that the mixed variants achieve improvements in reading comprehension (Khashabi et al., 2018), recognizing text entailments (Dagan et al., 2005; Giampiccolo et al., 2007; Bentivogli et al., 2009), and identifying the meaning of an ambiguous word (Levesque et al., 2012). Regarding the correlation with human language acquisition, the variants consistently showed near-zero correlations (see Appendix A for details).

We want to highlight that all our model variants pre-trained on augmented data saw fewer words in total (see Appendix A.3) but achieved similar or better results in the zero-shot and fine-tuning tasks.

Qualitative Insights Table 3 presents qualitative examples generated with the RTA method and possibly explains the pronounced gains in entity tracking and grammatical understanding of the LMs. For example, the sentence `"Two months passed, and spring deepened into summer"` and the augmented version `"Days deepened into summer"` could improve entity tracking (Kim and Schuster, 2023; Charpentier et al., 2025) by highlighting time progressions. Similarly, swapping phrases such as `"My Missionary life has, on the whole, been a very happy one..."` with `"My Missionary life has, on the whole, was very happy"` leads to syntactic robustness, which may have resulted in stronger BLiMP (Warstadt et al., 2020) results in the augmented datasets. Mixing weather descriptions, as in the examples `"It was very early on a hot December morning."` and `"On nice sunny days when it was not very cold she took them out in the carriage,"` could, for example, expand world knowledge and thus have led to improvements in EWoK (Ivanova et al., 2024) and reading tasks (de Varda et al., 2024).

6 Conclusion

RecombiText Augmentation is a statistical compositional data augmentation approach that generates synthetic sentences via lexical-semantic similarities and a one-point crossover. Our experimental results show that the proposed augmentation method can expand low-resource datasets by two- to fourfold without degrading language model benchmark performance, and on tasks such as entity tracking, self-paced reading, and morphological generalization, it even outperforms models trained on the full original dataset. This highlights substantial data efficiency gains over training solely on the full original dataset.

		BLiMP				WUGs			Reading		
Variant	**Prop.**	**BLiMP**	**Suppl.**	**EWoK**	**ENT**	**Adj.**	**Past**	**COMPS**	**SPR**	**ET**	**GLUE**
Baseline	0%	61.62	58.17	50.01	12.95	0.69	-0.06	**51.48**	4.01	11.83	64.02
$\mathcal{D}_{base1M} + \mathcal{D}_{aug9M}$	10%	58.30	57.42	49.81	16.24	0.47	-0.17	50.38	3.92	10.07	64.06
$\mathcal{D}_{base2.5M} + \mathcal{D}_{aug7.5M}$	25%	60.18	60.48	50.41	**19.65**	0.54	**0.24**	50.49	3.97	11.65	64.50
$\mathcal{D}_{base5M} + \mathcal{D}_{aug5M}$	50%	61.87	**61.08**	**50.80**	16.51	0.56	**0.24**	50.83	**4.74**	**12.05**	**64.76**
$\mathcal{D}_{base7.5M} + \mathcal{D}_{aug2.5M}$	75%	**62.91**	59.02	49.74	17.50	0.56	0.03	51.42	4.13	11.64	63.39
$\mathcal{D}_{base9M} + \mathcal{D}_{aug1M}$	90%	61.27	59.80	50.24	16.53	**0.71**	-0.05	50.84	4.39	11.80	62.70
$\mathcal{D}_{base10M} + \mathcal{D}_{aug10M}$	100%	61.99	59.11	50.20	16.19	0.54	0.07	51.22	4.32	12.02	64.66

Table 1: GPT-2 evaluation results with downsampled size from baseline dataset (prop.), Blimp, Bimp Supplement (Suppl.), EWoK, Entity Tracking (ENT), COMPS with accuracy (in %), WUGs with Spearman's rank correlation coefficient ρ, Self-paced Reading (SPR), Eye Tracking (ET) with change in R^2, and (Super)GLUE subset with macroaverage accuracy (in %).

		BLiMP				WUGs			Reading		
Variant	**Prop.**	**BLiMP**	**Suppl.**	**EWoK**	**ENT**	**Adj.**	**Past**	**COMPS**	**SPR**	**ET**	**GLUE**
Baseline	0%	54.44	50.98	**51.01**	32.27	0.58	-0.09	50.34	3.08	10.06	61.27
$\mathcal{D}_{base1M} + \mathcal{D}_{aug9M}$	10%	**57.06**	52.33	50.17	28.41	0.52	-0.03	50.66	**3.53**	10.17	61.51
$\mathcal{D}_{base2.5M} + \mathcal{D}_{aug7.5M}$	25%	55.15	51.77	49.38	**38.89**	0.48	**0.23**	**51.25**	3.52	11.01	61.19
$\mathcal{D}_{base5M} + \mathcal{D}_{aug5M}$	50%	55.56	51.55	49.89	33.65	0.67	-0.03	50.93	3.50	**11.02**	**62.29**
$\mathcal{D}_{base7.5M} + \mathcal{D}_{aug2.5M}$	75%	55.76	52.72	49.92	29.02	0.68	-0.20	50.90	3.42	**11.03**	61.65
$\mathcal{D}_{base9M} + \mathcal{D}_{aug1M}$	90%	54.30	51.67	49.62	33.83	0.57	-0.15	50.85	2.93	10.25	62.11
$\mathcal{D}_{base10M} + \mathcal{D}_{aug10M}$	100%	54.39	**53.10**	49.46	29.70	**0.72**	-0.03	50.49	2.81	10.48	61.98

Table 2: RoBERTa evaluation results with downsampled size from baseline dataset (prop.), Blimp, Bimp Supplement (Suppl.), EWoK, Entity Tracking (ENT), COMPS with accuracy (in %), WUGs with Spearman's rank correlation coefficient ρ, Self-paced Reading (SPR), Eye Tracking (ET) with change in R^2 and (Super)GLUE subset with macroaverage accuracy (in %).

Reference Sentences	**Augmented Sentences**
Two months passed, and spring deepened into summer.	Two months passed, and spring ran into weeks, weeks into months, but the expected agent of deliverance was not forthcoming.
Days ran into weeks, weeks into months, but the expected agent of deliverance was not forthcoming.	Days deepened into summer.
Life on the whole was very happy. My Missionary life has, on the whole, been a very happy one....'	Life on the whole, been a very happy one....' My Missionary life has, on the whole was very happy.
I've already started something. Cos I've been spectating	I've been spectating Cos I've already started something.
It was very early on a hot December morning.	it was not very cold she took them out in the carriage.
On nice sunny days when it was not very cold she took them out in the carriage.	On nice sunny days when It was very early on a hot December morning.

Table 3: Examples of RTA-generated augmented sentences based on a picked reference sentence and matching candidates.

Learnings

The experiments have demonstrated that the proposed method improves the morphological generalization, entity tracking, and reading comprehension capabilities of language models over the baseline. However, the strategy used, the frequency with which query sentences are combined with candidate sentences, and how augmentation is performed

play a significant role. Based on the experiments and results, it can be assumed that language models develop a better understanding of language when they frequently see how the same sentence can be combined in different ways. This can be observed, for example, in variants with 25% original data (25/75), which yield better results in entity tracking and morphological generalization than more balanced variants.

Limitations

The success of semantic search for candidates and the context window depends on word embeddings. As a result, ineffective or low-quality embeddings can lead to candidates that are less semantically relevant and therefore influence the relevance for the resulting augmented sentences. Furthermore, as the embedding quality decreases, the algorithm relies more heavily on lexical matches within the context windows for the respective intersections. It is plausible to assume that such factors could influence the performance when evaluating the language models.

Focusing on the evaluation of the quality of word and sentence embeddings when using the RTA method could ensure robust semantic alignment. Furthermore, a separate assessment of the effects of lexical versus semantic augmentations can lead to a better understanding of the respective improvements. The selection of the matching candidates and the presentation of the resulting data to the language model could provide deeper insights into the individual contributions. This separate evaluation would further contribute to the optimization of the RTA method by showing which type of augmentation leads to improvements. Investigating the effects of the used RTA hyperparameters could also enable more targeted data augmentation strategies for low-resource scenarios.

References

Jacob Andreas. 2020. Good-Enough Compositional Data Augmentation.

Luisa Bentivogli, Peter Clark, Ido Dagan, and Danilo Giampiccolo. 2009. The fifth pascal recognizing textual entailment challenge. *TAC*, 7(8):1.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Tyler A. Chang and Benjamin K. Bergen. 2022. Word acquisition in neural language models. *Transactions of the Association for Computational Linguistics*, 10:1–16.

Lucas Charpentier, Leshem Choshen, Ryan Cotterell, Mustafa Omer Gul, Michael Hu, Jaap Jumelet, Tal Linzen, Jing Liu, Aaron Mueller, Candace Ross, and 1 others. 2025. Babylm turns 3: Call for papers for the 2025 babylm workshop. *arXiv preprint arXiv:2502.10645*.

Lucas Georges Gabriel Charpentier and David Samuel. 2024. Gpt or bert: why not both? *arXiv preprint arXiv:2410.24159*.

BNC Consortium and 1 others. 2007. British national corpus. *Oxford Text Archive Core Collection*.

Gordon V Cormack, Charles LA Clarke, and Stefan Buettcher. 2009. Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 758–759.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine learning challenges workshop*, pages 177–190. Springer.

Andrea Gregor de Varda, Marco Marelli, and Simona Amenta. 2024. Cloze probability, predictability ratings, and computational estimates for 205 english sentences, aligned with existing eeg and reading time data. *Behavior Research Methods*, 56(5):5190–5213.

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.

Lukas Edman, Lisa Bylinina, Faeze Ghorbanpour, and Alexander Fraser. 2024. Are BabyLMs Second Language Learners? *arXiv preprint*. ArXiv:2410.21254 [cs].

Ronen Eldan and Yuanzhi Li. 2023. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*.

Kawin Ethayarajh. 2018. Unsupervised random walk sentence embeddings: A strong but simple baseline. In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 91–100.

Steven Y Feng, Varun Gangal, Jason Wei, Sarath Chandar, Soroush Vosoughi, Teruko Mitamura, and Eduard Hovy. 2021. A survey of data augmentation approaches for nlp. *arXiv preprint arXiv:2105.03075*.

Philip Gage. 1994. A new algorithm for data compression. *The C Users Journal*, 12(2):23–38.

Martin Gerlach and Francesc Font-Clos. 2020. A standardized project gutenberg corpus for statistical analysis of natural language and quantitative linguistics. *Entropy*, 22(1):126.

Danilo Giampiccolo, Bernardo Magnini, Ido Dagan, and William B Dolan. 2007. The third pascal recognizing textual entailment challenge. In *Proceedings of the ACL-PASCAL workshop on textual entailment and paraphrasing*, pages 1–9.

John J Godfrey, Edward C Holliman, and Jane McDaniel. 1992. Switchboard: Telephone speech corpus for research and development. In *Acoustics, speech, and signal processing, ieee international conference on*, volume 1, pages 517–520. IEEE Computer Society.

David E. Goldberg. 1989. *Genetic Algorithms in Search, Optimization and Machine Learning*, 1st edition. Addison-Wesley Longman Publishing Co., Inc., USA.

Akari Haga, Akiyo Fukatsu, Miyu Oba, Arianna Bisazza, and Yohei Oseki. 2024. BabyLM Challenge: Exploring the Effect of Variation Sets on Language Model Training Efficiency. *arXiv preprint*. ArXiv:2411.09587 [cs].

Michael A Hedderich, Lukas Lange, Heike Adel, Jannik Strötgen, and Dietrich Klakow. 2020. A survey on recent approaches for natural language processing in low-resource scenarios. *arXiv preprint arXiv:2010.12309*.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, and 1 others. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Valentin Hofmann, Leonie Weissweiler, David R Mortensen, Hinrich Schütze, and Janet B Pierrehumbert. 2025. Derivational morphology reveals analogical generalization in large language models. *Proceedings of the National Academy of Sciences*, 122(19):e2423232122.

Michael Y Hu, Aaron Mueller, Candace Ross, Adina Williams, Tal Linzen, Chengxu Zhuang, Ryan Cotterell, Leshem Choshen, Alex Warstadt, and Ethan Gotlieb Wilcox. 2024. Findings of the second babylm challenge: Sample-efficient pretraining on developmentally plausible corpora. *arXiv preprint arXiv:2412.05149*.

Anna A Ivanova, Aalok Sathe, Benjamin Lipkin, Unnathi Kumar, Setayesh Radkani, Thomas H Clark, Carina Kauf, Jennifer Hu, RT Pramod, Gabriel Grand, and 1 others. 2024. Elements of world knowledge (ewok): A cognition-inspired framework for evaluating basic world knowledge in language models. *arXiv preprint arXiv:2405.09605*.

Daniel Jurafsky and James H Martin. 2019. Speech and language processing 3rd edition draft. *October 2019*.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. 2018. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 252–262.

Najoung Kim and Sebastian Schuster. 2023. Entity tracking in language models. *arXiv preprint arXiv:2305.02363*.

Hector J Levesque, Ernest Davis, and Leora Morgenstern. 2012. The winograd schema challenge. *KR*, 2012(13th):3.

Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 923–929.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Alex Lyman and Bryce Hepner. 2024. Whatif: Leveraging word vectors for small-scale data augmentation. In *The 2nd BabyLM Challenge at the 28th Conference on Computational Natural Language Learning*, pages 229–236.

Brian MacWhinney. 2000. *The CHILDES project: The database*, volume 2. Psychology Press.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*.

Kanishka Misra, Julia Taylor Rayz, and Allyson Ettinger. 2022. Comps: Conceptual minimal pair sentences for testing robust property knowledge and its inheritance in pre-trained language models. *arXiv preprint arXiv:2210.01963*.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Stephen Robertson and Hugo Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389. Publisher: Now Publishers.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Andreas Stolcke, Klaus Ries, Noah Coccaro, Elizabeth Shriberg, Rebecca Bates, Daniel Jurafsky, Paul Taylor, Rachel Martin, Carol Van Ess-Dykema, and Marie Meteer. 2000. Dialogue act modeling for automatic tagging and recognition of conversational speech. *Computational linguistics*, 26(3):339–373.

Ole Tange. 2025. Gnu parallel 20250522 ('leif tange'). GNU Parallel is a general parallelizer to run multiple serial command line programs in parallel without changing them.

Nikitas Theodoropoulos, Giorgos Filandrianos, Vassilis Lyberatos, Maria Lymperaiou, and Giorgos Stamou. 2024. BERTtime Stories: Investigating the Role of Synthetic Story Data in Language pre-training. *arXiv preprint*. ArXiv:2410.15365 [cs].

Inar Timiryasov and Jean-Loup Tastet. 2023. Baby llama: knowledge distillation from an ensemble of teachers trained on a small dataset with no performance penalty. *arXiv preprint arXiv:2308.02019*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *Advances in neural information processing systems*, 32.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*.

Alex Warstadt and Samuel R Bowman. 2022. What artificial neural networks can tell us about human language acquisition. In *Algebraic structures in natural language*, pages 17–60. CRC Press.

Alex Warstadt, Aaron Mueller, Leshem Choshen, Ethan Wilcox, Chengxu Zhuang, Juan Ciro, Rafael Mosquera, Bhargavi Paranjabe, Adina Williams, Tal Linzen, and 1 others. 2025. Findings of the babylm challenge: Sample-efficient pretraining on developmentally plausible corpora. *arXiv preprint arXiv:2504.08165*.

Alex Warstadt, Alicia Parrish, Haokun Liu, Anhad Mohananey, Wei Peng, Sheng-Fu Wang, and Samuel R. Bowman. 2020. BLiMP: The Benchmark of Linguistic Minimal Pairs for English. *Transactions of the Association for Computational Linguistics*, 8:377–392.

Leonie Weissweiler, Valentin Hofmann, Anjali Kantharuban, Anna Cai, Ritam Dutt, Amey Hengle, Anubha Kabra, Atharva Kulkarni, Abhishek Vijayakumar, Haofei Yu, and 1 others. 2023. Counting the bugs in chatgpt's wugs: A multilingual investigation into the morphological capabilities of a large language model. *arXiv preprint arXiv:2310.15113*.

Zheyu Zhang, Han Yang, Bolei Ma, David Rügamer, and Ercong Nie. 2023. Baby's CoThought: Leveraging Large Language Models for Enhanced Reasoning in Compact Models. In *Proceedings of the BabyLM Challenge at the 27th Conference on Computational Natural Language Learning*, pages 158–170, Singapore. Association for Computational Linguistics.

A Appendix A

A.1 Setup Details

The RTA algorithm was implemented locally with Python 3.13. GNU Parallel (Tange, 2025) was used to execute the jobs for generating the augmented datasets, enabling multiple instances to be run simultaneously. The language models were trained on 1x NVIDIA H100 Tensor Core GPU with Python 3.12.9, Transformers 4.50.3, and PyTorch 2.7.1+cu126. For reproducibility, random seeds are used.

A.2 Data Preprocessing

The datasets were preprocessed to remove specific metadata. The custom Python script from Timiryasov and Tastet (2023) was used. We apply corpus-specific cleanup functions to normalize the text for model training. The common steps for all functions include removing extra spaces, tabs, and unnecessary line breaks. Additional customized processes vary depending on the corpus. For example, in OpenSubtitles (Lison and Tiedemann, 2016), subtitle credits are removed. In Simple English Wikipedia, leading line breaks at the end are removed.

A.3 Language Model Pre-training Details

GloVe (Pennington et al., 2014) is trained exclusively on the baseline proportion $|\mathcal{D}_{baseX}|$, which is deducted from the total 100 million word budget to determine the remaining allocation for LM training. The training corpus was tokenized using Byte Pair Encoding (BPE) (Gage, 1994; Sennrich et al., 2015) with the script from Charpentier and Samuel (2024). This yields the average number of subword tokens per whitespace-separated word (Avg. Splits/Word), which varies slightly across variants due to differences in data composition. All LMs use a batch size of 16 and a sequence length of 512, resulting in $16 \times 512 = 8192$ tokens per training step. The approximate number of words processed per step is then calculated as $\text{Words/Step} = \frac{\text{Tokens/Step}}{\text{Avg. Splits/Word}}$. To utilize the allocated number of words for the LMs without exceeding the budget, the maximum training steps are determined by the formula $\text{Max Steps} = \frac{\text{\# Words LM}}{\text{Words/Step}}$. The number of times the LMs iterate over the dataset is called an epoch and is, in our case, calculated as $\text{Epoch} = \frac{\text{\# Words LM}}{\text{\# Words Dataset}}$. Checkpoints are created every 1 million words for the first 10 million words and every 10 million words thereafter, up to a total of 100 million words. To determine when the language models have encountered a specific number of words, we perform specific calculations. Table 6 shows the detailed experimental language model setup for each variant.

Dataset	Description	Citation	# Words
British National Corpus (BNC)	Dialogue	(Consortium et al., 2007)	0.93M
CHILDES	Child-directed speech	(MacWhinney, 2000)	2.84M
Project Gutenberg (children's stories)	Written English	(Gerlach and Font-Clos, 2020)	2.54M
OpenSubtitles	Movie subtitles	(Lison and Tiedemann, 2016)	2.04M
Simple English Wikipedia	Written Simple English	-	1.45M
Switchboard Dialog Act Corpus	Dialogue	(Godfrey et al., 1992; Stolcke et al., 2000)	0.15M
Total			9.95M

Table 4: Datasets for the experiments simulating the low-resource scenario by Charpentier et al. (2025) after preprocessing.

Variant	Proportion	# Total Words	# Base Words	# Aug Words	Ratio r
$\mathcal{D}_{base10M}$	0%	9.95M	9.95M	0.00M	0.00
$\mathcal{D}_{base1M} + \mathcal{D}_{aug9M}$	10%	10.55M	1.06M	9.49M	9.00
$\mathcal{D}_{base2.5M} + \mathcal{D}_{aug7.5M}$	25%	9.68M	2.43M	7.25M	3.00
$\mathcal{D}_{base5M} + \mathcal{D}_{aug5M}$	50%	9.72M	4.96M	4.76M	1.00
$\mathcal{D}_{base7.5M} + \mathcal{D}_{aug2.5M}$	75%	9.99M	7.48M	2.52M	0.33
$\mathcal{D}_{base9M} + \mathcal{D}_{aug1M}$	90%	9.96M	8.95M	1.01M	0.11
$\mathcal{D}_{base10M} + \mathcal{D}_{aug10M}$	100%	19.49M	9.95M	9.54M	1.00

Table 5: Sampled proportions from the original `strict-small` low-resource training data by Charpentier et al. (2025), total words (in millions) in the resulting dataset, actual number of base words sampled from the baseline dataset $\mathcal{D}_{base}$, augmented words generated as $\mathcal{D}_{aug}$ with RTA and the approximate nominal augmentation ratio $r = \frac{|\mathcal{D}_{aug}|}{|\mathcal{D}_{base}|}$. The actual ratios may vary slightly due to data preprocessing.

Variant	# Words Dataset	Avg. Splits/Word	# Words GloVe	# Words LM	Tokens/Step	Words/Step	Max Steps	Epochs
$\mathcal{D}_{base10M}$	9.95M	1.608	0.00M	100.00M	8,192	5,095	19,627	10.05
$\mathcal{D}_{base1M} + \mathcal{D}_{aug9M}$	10.55M	1.463	1.06M	98.94M	8,192	5,599	17,672	9.38
$\mathcal{D}_{base2.5M} + \mathcal{D}_{aug7.5M}$	9.68M	1.518	2.43M	97.57M	8,192	5,397	18,079	10.08
$\mathcal{D}_{base5M} + \mathcal{D}_{aug5M}$	9.72M	1.529	4.96M	95.04M	8,192	5,358	17,738	9.77
$\mathcal{D}_{base7.5M} + \mathcal{D}_{aug2.5M}$	9.99M	1.564	7.48M	92.52M	8,192	5,238	17,664	9.26
$\mathcal{D}_{base9M} + \mathcal{D}_{aug1M}$	9.96M	1.59	8.95M	91.05M	8,192	5,152	17,672	9.14
$\mathcal{D}_{base10M} + \mathcal{D}_{aug10M}$	19.49M	1.527	9.95M	90.05M	8,192	5,365	16,785	4.62

Table 6: Language Model training setup for each data augmentation variant, where # Words represents the total whitespace-separated words in the preprocessed training dataset, Avg. Splits/Word indicates the average subword tokens per word from BPE tokenizer, # Words GloVe is the size of the dataset with which the GloVe model was trained, # Words LM reflects the budgeted words for the language model, Tokens/Step is the number of tokens per training step the LM sees, Words/Step approximates the number of words per training step the LM sees, Max Steps gives the total training steps for the LM, Epochs is the number the LM iterates over the dataset.

Variant	PPL		Self-BLEU	
	D_{baseX}	D_{augY}	D_{baseX}	D_{augY}
$\mathcal{D}_{base1M} + \mathcal{D}_{aug9M}$	18.23	47.50	8.83	12.65
$\mathcal{D}_{base2.5M} + \mathcal{D}_{aug7.5M}$	17.69	49.84	8.67	12.56
$\mathcal{D}_{base5M} + \mathcal{D}_{aug5M}$	17.83	54.21	8.50	12.33
$\mathcal{D}_{base7.5M} + \mathcal{D}_{aug2.5M}$	17.81	55.05	8.02	18.33
$\mathcal{D}_{base9M} + \mathcal{D}_{aug1M}$	17.66	59.21	8.30	22.67
$\mathcal{D}_{base10M} + \mathcal{D}_{aug10M}$	17.75	53.51	8.19	11.74

Table 7: Data quality evaluation for the base dataset $\mathcal{D}_{base}$ and the augmented dataset $\mathcal{D}_{aug}$ with their respective PPL and Self-BLEU (in %) values.

Variant	AoA	
	GPT-2	RoBERTa
Baseline	-0.001 (p=0.997)	0.00
$\mathcal{D}_{base1M} + \mathcal{D}_{aug9M}$	0.120 (p=0.670)	-0.274 (p=0.656)
$\mathcal{D}_{base2.5M} + \mathcal{D}_{aug7.5M}$	0.233 (p=0.404)	0.00
$\mathcal{D}_{base5M} + \mathcal{D}_{aug5M}$	0.031 (p=0.888)	0.00
$\mathcal{D}_{base7.5M} + \mathcal{D}_{aug2.5M}$	0.265 (p=0.182)	0.00
$\mathcal{D}_{base9M} + \mathcal{D}_{aug1M}$	0.089 (p=0.718)	0.00
$\mathcal{D}_{base10M} + \mathcal{D}_{aug10M}$	0.204 (p=0.388)	0.00

Table 8: AoA results for the different model variants.

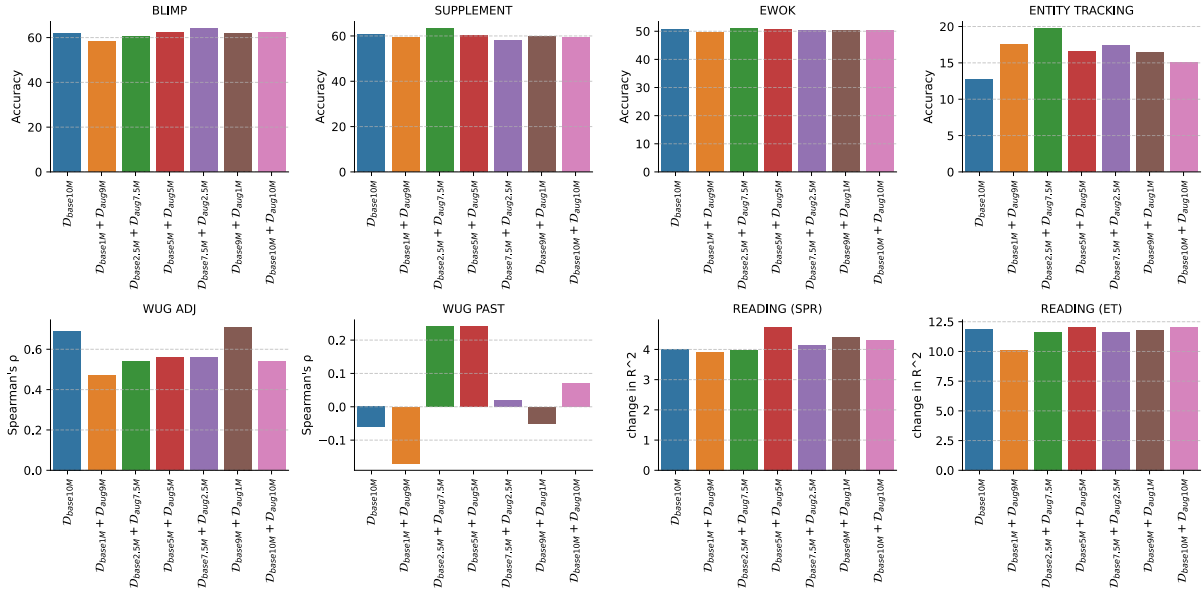

Figure 2: GPT-2 final fast zero-shot checkpoint results.

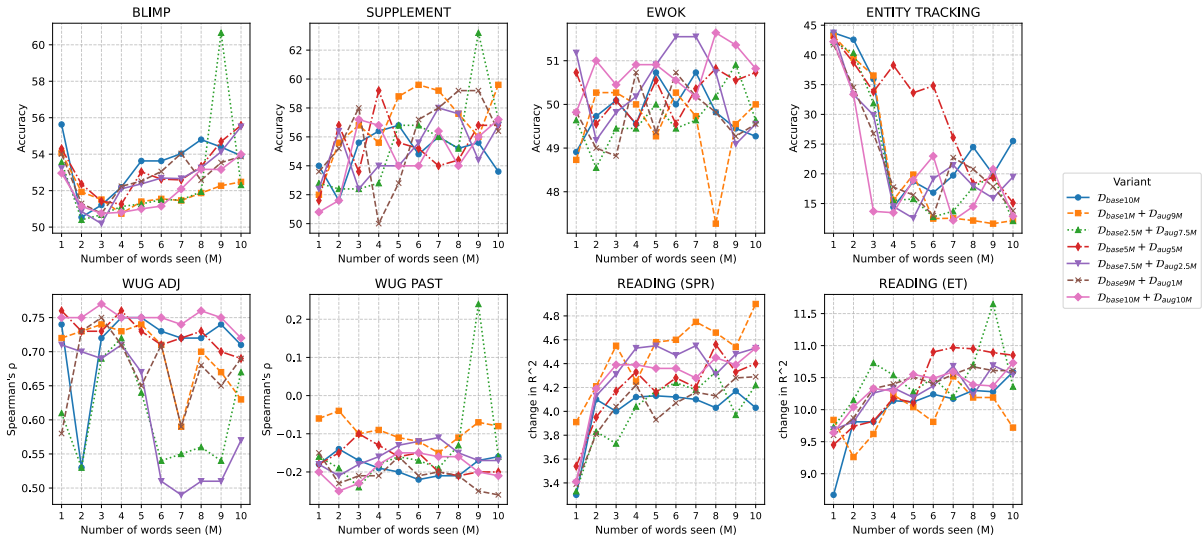

Figure 3: GPT-2 fast zero-shot results across training checkpoints where the language model has seen between 1M–10M words.

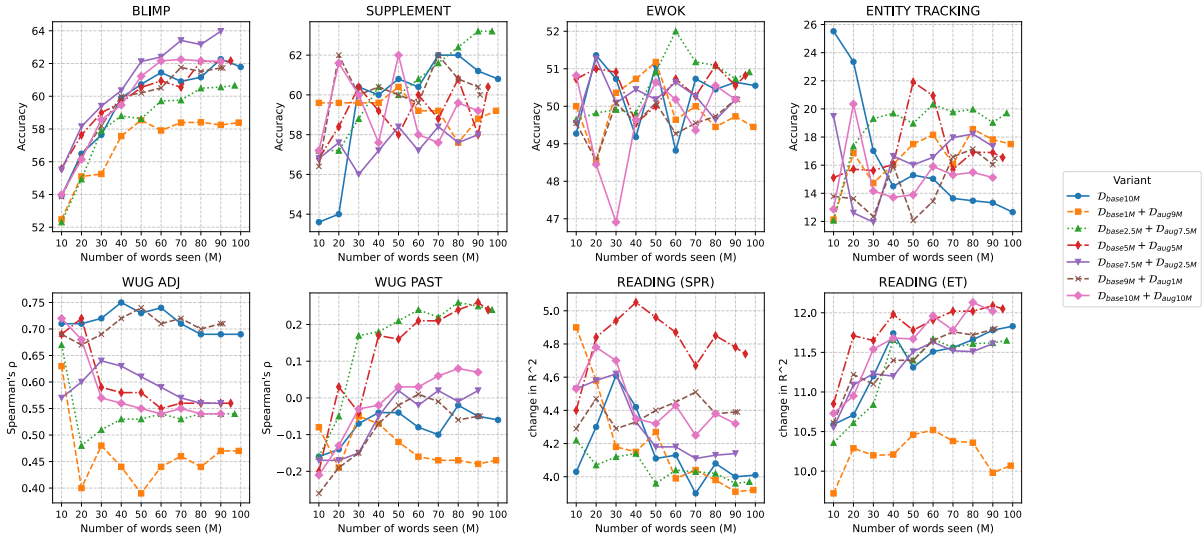

Figure 4: GPT-2 fast zero-shot results across training checkpoints where the language model has seen between 10M–100M words.

Variant	Prop.	BoolQ	MNLI	MRPC	QQP	MultiRC	RTE	WSC	Macro Avg.
Baseline	0%	67.58	50.81	81.25	62.86	**63.94**	**60.43**	**67.31**	64.02
$\mathcal{D}_{base1M} + \mathcal{D}_{aug9M}$	10%	67.65	54.79	82.46	66.99	63.94	53.96	63.46	64.06
$\mathcal{D}_{base2.5M} + \mathcal{D}_{aug7.5M}$	25%	67.03	54.16	82.01	66.97	63.94	59.71	63.46	64.50
$\mathcal{D}_{base5M} + \mathcal{D}_{aug5M}$	50%	66.73	**55.89**	**83.23**	66.17	60.81	59.71	65.38	**64.76**
$\mathcal{D}_{base7.5M} + \mathcal{D}_{aug2.5M}$	75%	66.91	52.89	81.96	66.73	59.98	56.12	65.38	63.39
$\mathcal{D}_{base9M} + \mathcal{D}_{aug1M}$	90%	67.83	49.29	81.37	61.22	60.35	58.27	63.46	62.70
$\mathcal{D}_{base10M} + \mathcal{D}_{aug10M}$	100%	**68.75**	55.62	82.65	**68.20**	60.60	56.12	65.38	64.66

Table 9: GPT-2 detailed fine-tuning results.

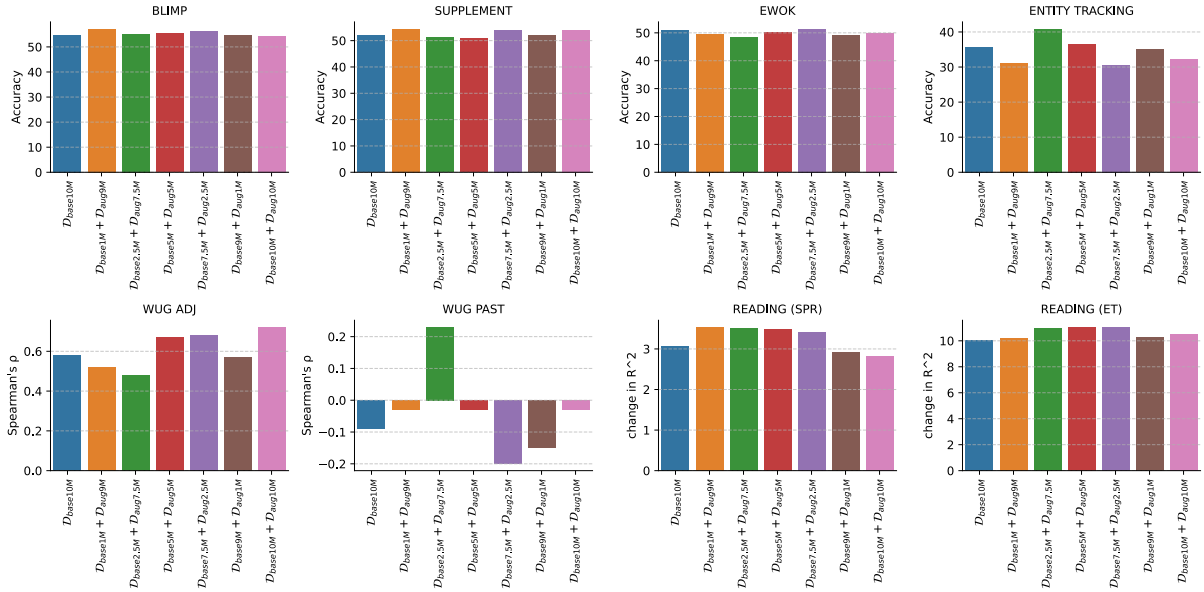

Figure 5: RoBERTa final fast zero-shot checkpoint results.

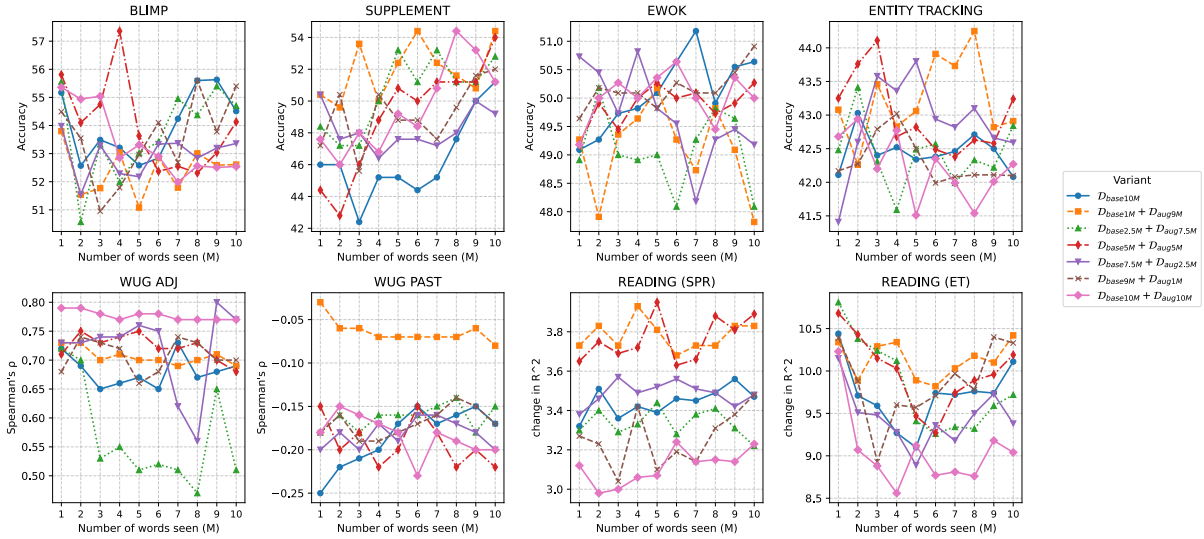

Figure 6: RoBERTa fast zero-shot results across training checkpoints where the language model has seen between 1M–10M words.

Variant	Prop.	BoolQ	MNLI	MRPC	QQP	MultiRC	RTE	WSC	Macro Avg.
Baseline	0%	66.24	40.99	**83.39**	59.46	57.55	54.68	65.38	61.27
$\mathcal{D}_{base1M} + \mathcal{D}_{aug9M}$	10%	66.36	42.32	81.23	**61.18**	57.96	56.12	65.38	61.51
$\mathcal{D}_{base2.5M} + \mathcal{D}_{aug7.5M}$	25%	66.12	41.95	81.55	60.33	57.55	56.83	61.54	61.19
$\mathcal{D}_{base5M} + \mathcal{D}_{aug5M}$	50%	66.54	**43.42**	82.47	61.08	**60.23**	55.40	65.38	**62.29**
$\mathcal{D}_{base7.5M} + \mathcal{D}_{aug2.5M}$	75%	**67.22**	43.13	82.28	60.31	57.96	56.83	65.38	61.65
$\mathcal{D}_{base9M} + \mathcal{D}_{aug1M}$	90%	66.67	42.46	82.13	60.59	57.96	**58.99**	**67.31**	62.11
$\mathcal{D}_{base10M} + \mathcal{D}_{aug10M}$	100%	65.87	42.14	80.75	58.47	59.32	57.55	**67.31**	61.98

Table 10: RoBERTa detailed fine-tuning results.

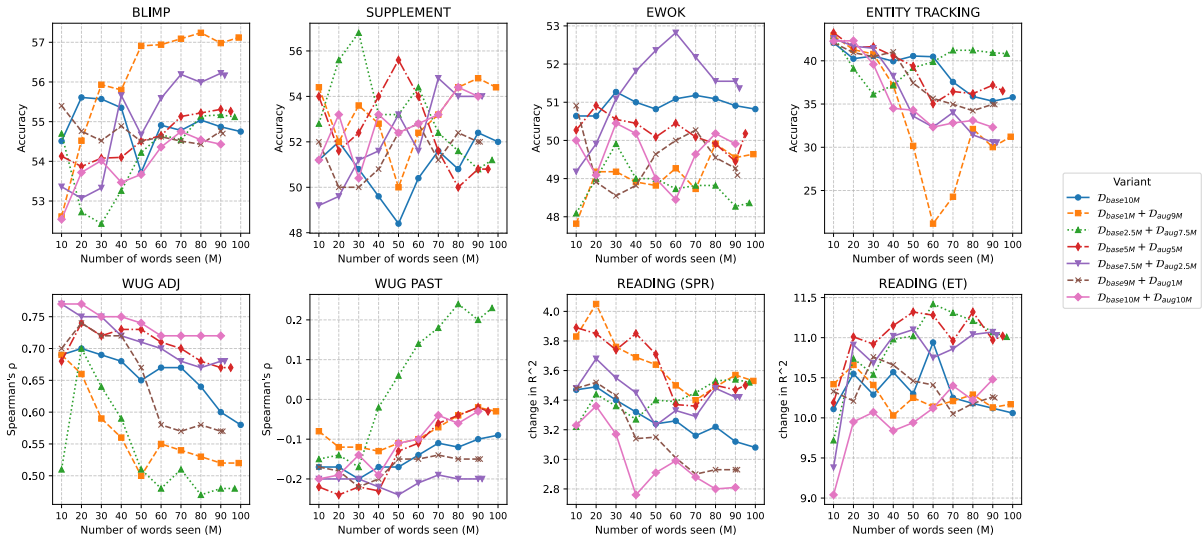

Figure 7: RoBERTa fast zero-shot results across training checkpoints where the language model has seen between 10M–100M words.

Hyperparameter	**Value**
Vector Size	50
Window Size	10
Minimum Vocabulary Count	5
Maximum Co-occurrence Weight (`x_max`)	10
Maximum Iterations	25
Number of Threads	4
Memory	4.0 GB

Table 11: Hyperparameters for the GloVe model used in the experiments.

Variant	**RRF** k	τ_{window}	W	**top-**k	T	**Ratio** r
$\mathcal{D}_{base1M} + \mathcal{D}_{aug9M}$	60	60%	3	50	1.3	9.00
$\mathcal{D}_{base2.5M} + \mathcal{D}_{aug7.5M}$	60	60%	3	30	1.1	3.00
$\mathcal{D}_{base5M} + \mathcal{D}_{aug5M}$	60	60%	3	20	1.0	1.00
$\mathcal{D}_{base7.5M} + \mathcal{D}_{aug2.5M}$	60	60%	3	15	0.9	0.33
$\mathcal{D}_{base9M} + \mathcal{D}_{aug1M}$	60	60%	3	10	0.8	0.11
$\mathcal{D}_{base10M} + \mathcal{D}_{aug10M}$	60	60%	3	20	1.0	1.00

Table 12: Hyperparameters for the RTA algorithm used in the experiments.

Hyperparameter	Value
Model Type	openai-community/gpt2
Tokenizer	BPE
Learning Rate	5×10^{-5}
Maximum Gradient Norm	1.0
Adam β_1	0.9
Adam β_2	0.999
Adam ϵ	1×10^{-8}
Block Size	512
Batch Size	16
Save Strategy	Steps
Save Total Limit	20
Logging Steps	100
Evaluation Strategy	Steps
Seed	42

Table 13: Hyperparameters for the GPT-2 model training used in the experiments.

Hyperparameter	Value
Model Type	FacebookAI/roberta-base
Tokenizer	BPE
Learning Rate	5×10^{-5}
Maximum Gradient Norm	1.0
Adam β_1	0.9
Adam β_2	0.999
Adam ϵ	1×10^{-8}
Sequence Length	512
Batch Size	16
MLM Probability	15%
Save Strategy	Steps
Save Total Limit	20
Logging Steps	100
Evaluation Strategy	Steps
Seed	42

Table 14: Hyperparameters for the RoBERTa model training used in the experiments.

Hyperparameter	MultiNLI, RTE, QQP, MRPC	BoolQ, MultiRC	WSC
Learning Rate	3×10^{-5}	3×10^{-5}	3×10^{-5}
Batch Size	32	16	32
Epochs	10	10	30
Weight Decay	0.01	0.01	0.01
Optimizer	AdamW	AdamW	AdamW
Scheduler	cosine	cosine	cosine
Warmup Percentage	6%	6%	6%
Dropout	0.1	0.1	0.1

Table 15: Hyperparameters for fine-tuning the language models used in the experiments.

Expression	Definition
D	Set of documents
q	Query sentence
$\|q\|, \|m\|$	Sentence lengths
$\tilde{q}, \tilde{m}$	Augmented sentence pairs
$\mathcal{R}_{BM25}$	Lexical search results (BM25)
$\mathcal{R}_{kNN}$	Semantic search results (k-NN)
$\mathcal{R}_{RRF}$	Fused ranked list from Reciprocal Rank Fusion
τ_{window}	Threshold for context window similarity
k_{top}	Number of top-k candidates for sampling
$p(x_i)$	Softmax probability
m	Candidate sentence
W	Fixed size of the context window
$\mathbf{v}_t \in \mathbb{R}^d$	Word embedding of token t in d-dimensional space
$\mathbf{v}_{uSIF} \in \mathbb{R}^d$	Sentence embeddings in d-dimensional space
$\text{idf}(t, D)$	IDF value of token t in corpus D
$S_{\text{ctx}}(i, j)$	Context window similarity
i, j	Positions within the sliding context windows
k^*	Pivot index within the context window maximizing similarity
i^*, j^*	Pivot elements for q, m
$\|\|$	Concatenation operator

Table 16: Mathematical notation for the RecombiText Augmentation (RTA) algorithm.

Algorithm 1: RecombiText Augmentation (RTA) Pseudo Algorithm

Data: Dataset D, $q \in D$, $\mathbf{v}_t \in \mathbb{R}^d$, $\mathbf{v}_{uSIF} \in \mathbb{R}^d$, $\mathrm{idf}(t, D)$
Result: $\tilde{q}, \tilde{m}$
Retrieve $\mathcal{R}_{BM25}$ for q;
Retrieve $\mathcal{R}_{kNN}$ for q on $\mathbf{v}_{uSIF} \in \mathbb{R}^d$;
$\mathcal{R}_{RRF} \leftarrow empty$;
for d *in* $\mathcal{R}_{BM25} \cup \mathcal{R}_{kNN}$ **do**
    Compute $\mathrm{RRFscore}$;
    Add d to $\mathcal{R}_{RRF}$ with its RRF score;
end
Sort $\mathcal{R}_{RRF}$ in descending order of RRF scores;
$retry \leftarrow 1$;
for *retry* $\leq |\mathcal{R}_{RRF}|$ **do**
    Select k_{top} candidates from $\mathcal{R}_{RRF}$;
    Compute $p(x_i)$ on k_{top} candidates according to top-k sampling;
    Sample candidate sentence m according to probabilities p;
    Tokenize $q = \langle q_0, \ldots, q_{|q|-1} \rangle$ and $m = \langle m_0, \ldots, m_{|m|-1} \rangle$;
    Normalize q, m;
    $S_{\max} \leftarrow 0$, $i^* \leftarrow 0$, $j^* = 0$;
    for $i = 0$ **to** $|q| - W_q + 1$ **do**
        for $j = 0$ **to** $|m| - W_m + 1$ **do**
            Compute $S_{\text{ctx}}(i, j)$ according to Equation 1;
            if $S_{ctx}(i, j) > S_{\max}$ **then**
                $S_{\max} \leftarrow S_{\text{ctx}}(i, j)$;
                Compute k^* for W_q, W_m according to $\cos_k \times \mathrm{idf}_k$;
                $i^* \leftarrow i + k^*$;
                $j^* \leftarrow j + k^*$;
            end
        end
    end
    if $S_{\max} > \tau_{window}$ **then**
        $\tilde{q} = \langle q_0, \ldots, q_{i^*-1} \mid\mid m_{j^*}, \ldots, m_{|m|-1} \rangle$;
        $\tilde{m} = \langle m_0, \ldots, m_{j^*-1} \mid\mid q_{i^*}, \ldots, q_{|q|-1} \rangle$;
        return $\tilde{q}, \tilde{m}$;
    else
        $retry \leftarrow retry + 1$;
        Remove candidate m from $\mathcal{R}_{RRF}$;
        continue;
    end
end