Master's Thesis of Data Science

Effects of Duplicated Data in Language Modeling

– Effects of Data Duplication in Pretraining –

데이터 중복이 언어 모델에 미치는 영향

February 2023

Graduate School of Data Science
Seoul National University
Data Science Major

Dayeon Kang

Effects of Duplicated Data in Language Modeling

– Effects of Data Duplication in Pretraining –

Jaejin Lee

Submitting a master's thesis of
Data Science

December 2022

Graduate School of Data Science
Seoul National University
Data Science Major

Dayeon Kang

Confirming the master's thesis written by
Dayeon Kang
January 2023

Chair	이승근
Vice Chair	이재진
Examiner	김태섭

Abstract

This paper studies the effect of deduplication in training data on language models, such as BERT (the encoder−based model) and GPT−2 (the decoder−based model). Previous studies focus on memorizing duplicates in the training dataset whereas we perform several experiments with data deduplication. The pretraining data is first clustered by MinhashLSH, a stochastic method for finding near−duplicate documents in large corpus data, and then deduplicated by Jaccard similarity with various threshold values. Then, the models are finetuned with different downstream tasks. The experimental result indicates that GPT−2 works better with the deduplication, whereas BERT works differently depending on the tasks. It is due to the difference in self−supervised learning methods between BERT and GPT−2. The duplicated data may work on BERT as data augmentation through random masking in its data preprocessing stage. Data duplication may introduce biases and lead to overfitting, but the effect depends on the amount of duplicated data. To improve performance, data deduplication with proper granularity is essential in language model training.

Keyword : Duplication, Language Modeling, MinhashLSH, Self−supervised learning, Pretraining
Student Number : 2021−24330

Table of Contents

Chapter 1. Introduction

1.1. Study Background

Transformer−based pretrained language models (Vaswani et al. (2017)) are getting more attentions and have achieved remarkable performance in natural language processing (NLP) tasks (Xu et al., 2021). The evolution of these models began with GPT (Radford and Narasimhan, 2018) and BERT (Devlin et al., 2019) based on decoder and encoder architecture of the transformer, respectively. These models are basically built on top of self−supervised learning, which essentially learns universal language representations from large volumes of text data in a self−learning manner. These models also become as pretrained model, and by avoiding the training of downstream tasks from scratch, these pretrained model acts as background knowledge. For example, Rogers et al. (2021) reviewed how model BERT works, how it is represented inside the model, and what kind of information it acquires in three aspects: syntactic, semantic, and world knowledge.

Dataset used in self−supervised learning are generally large. The performance of pretraining models can be improved by using a larger dataset. The GPT−2 (Radford et al., 2019) is trained on WebText, a dataset of web documents made of all outbound links from highly ranked on Reddit (this dataset was not made available for public). They utilize 40 GB of text containing slightly over 8 million documents after deduplication and some heuristic based cleaning. On the other side, BERT is pretrained using text from Wikipedia and BookCorpus (Zhu et al., 2015), which amounts to 16GB. However, following to Bandy and Vincent (2021), BookCorpus not only has copyright problem but contains substantial amount of duplication, and exhibits significant skews in genre representation. Thus, our research excluded the BookCorpus from pretraining. Further studies such as XLnet (Yang et al., 2019) and Roberta (Liu et al., 2019) show that the performance can be increased by using a larger pretraining dataset.

Another common dataset used in self-supervised learning is Common Crawl, a dataset that extracts and collects about 20TB of text data from public web pages each month. Raffel et al. (2020) preprocessed and released around 750GB of a cleaned version of Common Crawl called *Colossal Clean Crawled Corpus* (or C4 for short). Finally, T5 based on both encoder and decoder transformer architecture showed state-of-the-art performance in overall NLP downstream tasks by utilizing C4. As our goal focuses on revealing effectiveness and granularity of duplication, the experiment is restricted to BERT and GPT-2 model, and to Wikipedia and RealNews (a subset of the Common Crawl consisting of articles from news domains (Zellers et al., 2019)) pretraining datasets.

As a large corpus plays an important role in the pretraining phase, data preprocessing techniques have recently come into the limelight in NLP, regardless of whether they are in English. It is natural to think duplicate data in the training dataset can make training a large language model more time-consuming and resource intensive.

Preprocessing comes first to obtain high quality data when training a model. C4 goes through heuristics preprocessing such as language detection, JavaScript removal, Bad Words filter, and discarding short lines. Deduplication is one of preprocessing steps. For instance, exact sentence deduplication methods are applied to the C4 dataset (Smith et al. (2013); Grave et al. (2018)). Although C4 is the result of elaborate text data processing, Lee et al. (2022) found the fraction of samples identified as near-duplicates are 3.04% for C4. Moreover, 13.63% of RealNews are near duplicates. As RealNews is derived from news sites, it is presented in slightly different formats on different news sites, resulting in many duplications (Lee et al. (2022)). Similarly, we crawled Korean data from public websites like Common Crawl and used the near duplicate method to find duplicates. *Modu* Korean language news datasets have about 1% near-duplicates even though they were cleansed before it is released, and in the raw crawled dataset showed about 10% overlap especially in the news domains.

1.2. Purpose of Research

Duplication in data means having multiple copies of the same information and could be useful for backing up important information or for sharing the same information with multiple people. However, duplication at the data collection stage can be unintentional or caused by data entry errors. Duplication typically means an exact copy of the original, but broadly it refers to highly similar ones. Thus, in this paper, we take advantage of the broader meaning of duplication and apply document-level near deduplication to the pretraining dataset. The inclusion of a broader concept could be proven by the results of downstream tasks.

Duplication in data can have several effects on the training model. First, duplication makes the data set larger than necessary, so training takes longer even if the results are the same. Additionally, redundant data can negatively impact the performance of model by introducing bias and inaccuracies into a model training. If the pretraining model was trained on duplicated data, errors are more difficult to identify and correct especially in finetuning, which can further reduce the quality of model predictions. Finally, Lee et al. (2022) finds that deduplicating training data makes language models better as models memorize duplicate data therefore biased. Thus, it is important to clean the data before using it for training.

In this paper, we are focusing on duplication in a pretraining stage. Pretraining means a technique of training a model in advance with a large dataset with self-supervised learning methods and then the model finetunes on a specific smaller dataset. Especially in the field of NLP, the transformer model architecture is used to pretrain and then add additional layers at the last part of the transformer model, or overlapping multiple models, if necessary to finetune on the downstream tasks. This approach allows the model to learn general features and avoid overfitting to smaller data which finally helps improve finetuned performance. In other words, pretraining gives a good starting point, which can make the finetuning process more effective. Our research focuses on duplicated data in pretraining phase in both encoder and decoder transformer models comparing results with downstream tasks. This will show how duplicated data in the pretraining stage affects the accuracy and performance of finetuning tasks, and at the same time examining its impact in terms of data quality in pretraining.

1.3. Related Work

Deduplicating Documents Traditionally, finding duplication in documents of a specific domain is a major task. Using model to detect duplication is prevalent trend in the deduplication field. Searle et al. (2021) presents information-theoretic (compression) and language modeling approaches to estimate clinical text redundancy. To improve the resemblance detection in the service cloud storage, Ye et al. (2022) combined BP-Neural network-based backpropagation algorithm with traditional resemblance detection methods. Gyawali et al. (2020) used Locality Sensitive Hashing and word embeddings to deduplicate scholarly documents. In our research, In this paper, MinhashLSH, a traditional Minhash algorithm (Broder, 1997) combined with locality-sensitive hashing (LSH) is used to deduplicate the pretraining dataset.

Documents Similarity Clustering similar documents, specifically used in semantic search or topic modeling, is one of the popular tasks in NLP. Locality Sensitive Hashing can be regarded as a clustering algorithm that aggregates similar datasets. However, to perform similarity tasks in practice, semantic similarity should be considered as well even though lexical and semantic similarities share many commonalities. SimCSE (Gao et al., 2021) is one of the examples considering the semantic similarity and results in high predictive performance on similarity tasks. Since we need to find duplicates in a large dataset, the method for finding duplicates is limited to MinhashLSH which only considers lexical similarity.

Data augmentation and Input perturbations Several surveys explored the data augmentation techniques for NLP (Hedderich et al., 2020; Feng et al., 2021). Data augmentation’s common approach is mitigating the need for labeled data by modifying existing data points through transformations. Therefore, it is mostly used in downstream tasks such as low resource scenarios. Likewise, input perturbations are also used in downstream tasks. Adversarial examples which make perturbation mislead a model to produce a specific wrong prediction. (Jin et al., 2020). Even though these studies are focusing on the quality of dataset as well, our research will explore data quality especially in the pretraining phase.

Memorizing training data Previous studies focus on memorizing duplicates in the training dataset, especially in a transformer-based decoder model, i.e., a generative model. Radford et al. (2019) identifies 8-gram overlaps between GPT-2's training and evaluation datasets and shows that excessive overlap can lead to memorization of the model and distort the generalization performance. Lee et al. (2022) finds that deduplicating training data improves language models. Models trained with duplicated data emit over 1% of memorized data. Since the amount of duplicated data is reduced, model training becomes more efficient than the original, and researchers reveal that deduplication does not hurt perplexity in the validation set. McCoy et al. (2021) define one way to ensure the novelty of model outputs is by deduplicating the training dataset. On the other side, Kandpal et al. (2022) handle privacy issues that come from memorizing a model by deduplicating data in the training dataset.

Chapter 2. Approach

In this section, we describe our approach and experimental methods.

2.1. Pretraining Models

Experiments are performed based on two different language model architectures, BERT (Devlin et al., 2019) and GPT-2 (Radford et al., 2019). They are designed based on the encoder or decoder part of the transformer architecture (Vaswani et al. (2017)). Considering the training speed, both BERT and GPT-2 are pretrained over the open-source Megatron-LM library[①]. Our model configurations, such as the hidden size, the number of layers, the number of attention heads, and the number of parameters, follow those of Megatron-LM (Shoeybi et al., 2019). Specifically, we use 'bert-large-uncased' tokenizer from Google to pretrain BERT, and 'gpt2' tokenizer from OpenAI to pretrain GPT-2. The two models have 12 layers with a hidden size of 768.

BERT and GPT-2 differ not only in the model architecture design but also in the pretraining objectives. BERT, a bi-directional transformer model, originally had masked language modeling (MLM) and next-sentence prediction objectives. Conversely, GPT-2, a left-to-right generative language model, was trained with a causal language modeling (CLM) objective and is powerful at predicting the next token in a sequence. Megatron-LM trains BERT with modified versions of the original objectives. They are whole-word n-gram masking proposed by Joshi et al. (2019) and the sentence order prediction proposed by Lan et al. (2019) replacing the next sentence prediction head. Due to different pretraining outputs and objectives, the two models usually perform better on different downstream tasks. BERT was typically tested with a General

[①] https://github.com/NVIDIA/ Megatron-LM

Language understanding Evaluation (GLUE) score, the SQUAD dataset. GPT-2 was evaluated on the TriviaAQ benchmark (Joshi et al., 2017) or LAMBADA dataset (Paperno et al., 2016), with zero-shot learning. Initially, the main goal of GPT-2 was to develop a general language model that can perform various tasks without finetuning. However, GPT-2 models are finetuned similarly to BERT to see its effects on downstream tasks.

2.2. Pretraining Dataset

In General, data crawled from websites is collected by respective URLs, and a text sequence consisting of several paragraphs or sentences remains after cleaning up. To apply our near deduplication method, we use a text sequence in a document, and the deduplication unit is a document.

Our experiment is based on a Wikipedia and RealNews dataset. A title with a text sequence is a single document in the Wikipedia dataset. We prepare the dataset by downloading the XML Wikipedia dumps dated October 2021 and extracting JSON files with an open-source tool, wikiextractor[②] . After preprocessing the raw dataset, the Wikipedia dataset contains 6.3 million documents with an average token length of 541. Its total size is 15GB. On the other side, one news article is a single document in RealNews. It is under the C4 officially released version called RealNewsLike and could be downloaded from the Tensorflow datasets[③]. After preprocessing, it contains 13.7M articles with average token length 574, and size 35GB.

2.3. Near Deduplication

[②] https://github.com/attardi/wikiextractor

[③] https://www.tensorflow.org/datasets/catalog/c4

Minhash (Broder, 1997) is an efficient algorithm widely used in a large−scale duplication search. It is used to represent a document by character n−grams called shingles and extract the given number of these shingles by the permutation of hash functions. The extracted shingles are called the signatures of the document. After being encoded by some hash functions, these signatures are divided into buckets by a static number of consecutive rows. This locality−sensitive hashing (LSH) technique relies on a probabilistic guarantee to produce hash collisions for similar contents[④].

Such a MinhashLSH implementation allows for faster deduplication of a large document corpus without comparing all pairs. If two documents are clustered in the same bucket, it is likely to have a certain similarity threshold. This method ensures very few false positives and false negatives. Proven by Juan, et al (2021), despite the fact that there are superior combinations of values in the training results, none of the combinations produced an AUC below 0.8. Finally, the members of the same bucket are compared using the Jaccard similarity to check if they actually exceed the similarity threshold. The comparison time is reduced significantly compared to comparing all pairs.

We use 5−gram Jaccard similarity in this paper. We also use 10 rows and 10 bands for the LSH hyperparameters. The threshold is the lowest bound for deduplication, and if the clustered documents' similarity is above the threshold, it leaves one and deletes the rest in the cluster.

When we apply MinhashLSH to deduplicate dataset, RealNews (13,777,199) only deletes about 0.65% (90,172) with the threshold 0.7. Therefore, our experiment is designed with the Wikipedia. With the threshold value of 0.7, it deletes 11.45\% (730,882) of the total number of documents (6,384,050) in Wikipedia. We prepare six pretraining datasets for our experiment: the original dataset and the deduplicated dataset with various Jaccard similarity threshold values of 0.5, 0.6, 0.7, 0.8, and 0.9.

[④] https://github.com/mattilyra/LSH

Figure 1 shows samples of near−duplicates with different Jaccard similarity threshold values. Underlines are the parts that are different between the documents. More underlines occur when the threshold is 0.5 than 0.7.

Trying to figure out which of these cars to buy? Compare the Maruti Suzuki Dzire Vs Maruti Suzuki Dzire on CarAndBike to make an informed buying decision as to which car to buy in 2019. The ex−showroom, New Delhi price of the Maruti Suzuki Dzire Petrol starts at ₹ 6.09 Lakh and goes up to ₹ 9.52 Lakh for the fully−loaded Petrol model.	Trying to figure out which of these cars to buy? Compare the Ford Figo Aspire Vs Maruti Suzuki Dzire on CarAndBike to make an informed buying decision as to which car to buy in 2019. The ex−showroom, New Delhi price of the Ford Figo Aspire Petrol starts at ₹ 6.21 Lakh and goes up to ₹ 9.76 Lakh for the fully−loaded Petrol model.

Figure 1−1. An example of near−duplicated documents Jaccard similarity approximately 0.7 in RealNews

Football tournament season The All−Ireland Senior B Hurling Championship of 1994 was the 21st staging of Ireland's secondary hurling knock−out competition. Roscommon won the championship, beating London 1−10 to 1−9 in the final at the Emerald GAA Grounds, Ruislip.	Football tournament season The All−Ireland Senior B Hurling Championship of 1993 was the 20th staging of Ireland's secondary hurling knock−out competition. Meath won the championship, beating London 2−16 to 1−16 in the final at the Emerald GAA Grounds, Ruislip.	Football tournament season The All−Ireland Senior B Hurling Championship of 1986 was the 13th staging of Ireland's secondary hurling knock−out competition. Kerry won the championship, beating London 3−11 to 1−10 in the final at the Emerald GAA Grounds, Ruislip.

Figure 1−2. An example of near−duplicated documents Jaccard similarity approximately 0.7 in Wikipedia

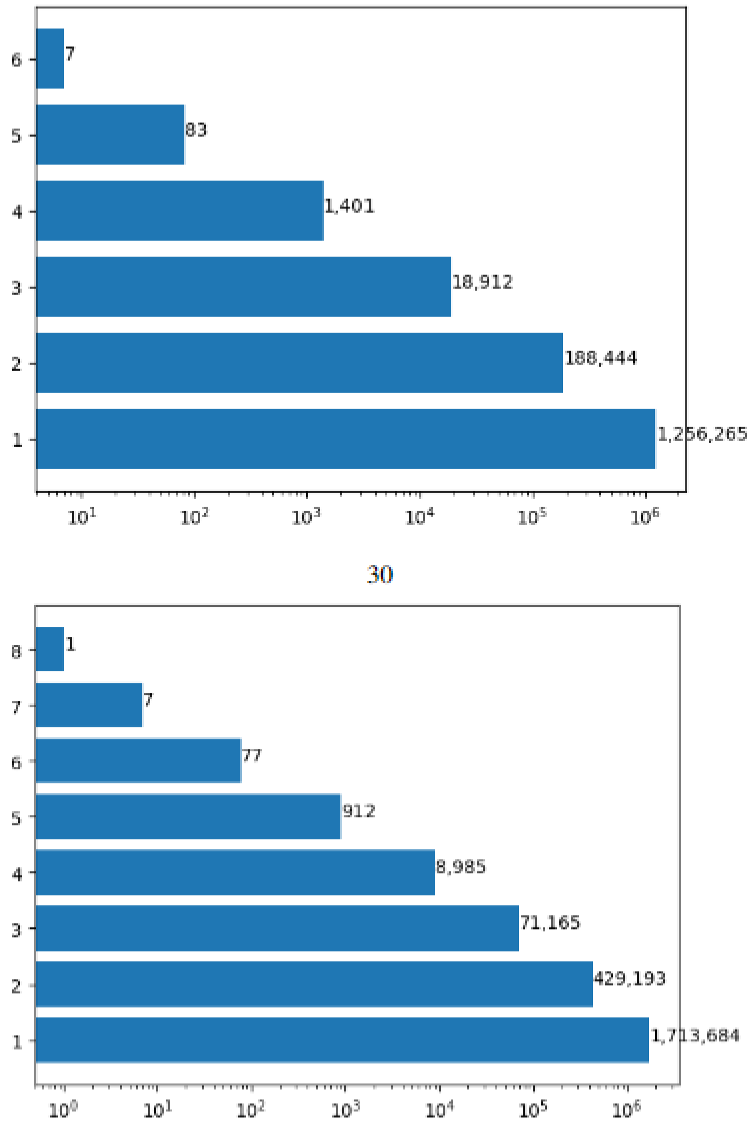
Species of butterfly Aloeides macmasteri, the	Species of butterfly Aloeides braueri, the Brauer's

McMaster's copper, is a butterfly of the family Lycaenidae. It is found in South Africa, where it is widespread but localised and known from the Western Cape, then across the Great Karoo to Namaqualand. It is also found from Coega to Grassland in the Eastern Cape. The wingspan is 28–32 mm for males and 30–35 mm females. Adults are on wing from September to November and from February to April. There are two generations per year.	copper, is a butterfly of the family Lycaenidae. It is found in South Africa, where it is known from highland hillsides covered in sour grassveld in the Eastern Cape. The wingspan is 26–28 mm for males and 28–32 mm females. Adults are on wing from October to November and from January to February. There are two generations per year.

Figure 1-3. An example of near-duplicated documents Jaccard similarity approximately 0.5 in Wikipedia

2.4. Injection of Exact Document Duplication

Random selection of documents for injection is conducted in the way of sampling without replacement. Figure2 shows the log scaled numbers of duplicated documents for 30% and 50% duplication, respectively. For instance, three on the y-axis means a document is injected (duplicated) three times into the original dataset.



30

50

Figure 2. The log scaled distribution of numbers sampled with sampling without replacement in duplication injection.

Chapter 3. Experiments

The General Language Understanding Evaluation (GLUE) benchmark (Wang et al., 2018a) is a collection of diverse natural language understanding tasks. For our downstream tasks, we select STS-B, CoLA, MRPC, and RTE in GLUE. The details of finetuning are in Appendix B. For GPT-2 models, we additionally perform a zero-shot test using the LAMBADA dataset (Paperno et al., 2016).

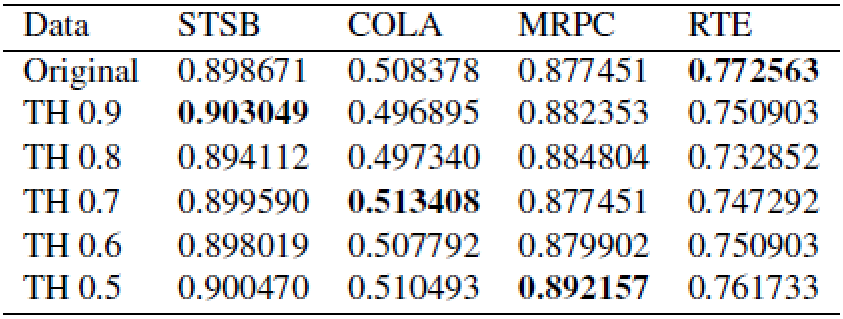
The performance of each pretrained model is measured over several epochs, and the best finetuning result of each model is selected to see the deduplication effects. See Appendix A for more details and the selected epochs of each model.

3.1. Near Deduplication Results

3.1.1. BERT

Table 1 shows the GLUE Test results of the BERT models. It is difficult to choose the best one from the results shown in Table 1. The performance of models pretrained with different deduplicated datasets is inconsistent and varies depending on the types of tasks.

The inconsistency of results between original and deduplicated datasets could be interpreted based on the self-supervised learning method of BERT. BERT puts random masks into the input samples when training. There could exist cases where the same sentence is masked differently. Since BERT learns by predicting the masks, different masks for the same sentence correspond to different prediction results. That is, duplicate sentences play the role of different sentences and improve the generalization capability of BERT. Thus, data duplication only affects BERT a little, and it has a similar effect to increasing the amount of training data.



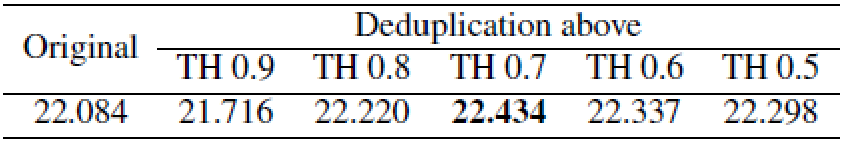
Data	STSB	COLA	MRPC	RTE
Original	0.898671	0.508378	0.877451	**0.772563**
TH 0.9	**0.903049**	0.496895	0.882353	0.750903
TH 0.8	0.894112	0.497340	0.884804	0.732852
TH 0.7	0.899590	**0.513408**	0.877451	0.747292
TH 0.6	0.898019	0.507792	0.879902	0.750903
TH 0.5	0.900470	0.510493	**0.892157**	0.761733

Table 1. GLUE Test results of BERT models.

3.1.2. GPT-2

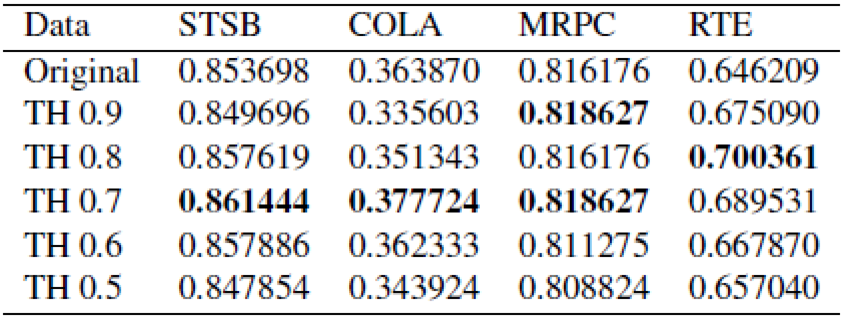
The LAMBADA dataset (Paperno et al., 2016) tests the ability of language models to structure long-range dependencies in text. The task is to predict the final word of sentences because a human requires at least 50 tokens of context to predict successfully. Table 2 shows that the pretrained GPT-2 model with the deduplication threshold of 0.7 results in the best accuracy. This may suggest that deduplicating the data around the threshold of 0.7 least overfits to the pretraining data and better generalizes the model.

In contrast to the results of BERT, Table 3 shows GPT-2's consistent results for the downstream tasks. Overall, the threshold around 0.7 gives the best performance. The results can be interpreted similarly to the previous LAMBADA results of GPT-2. That is, the better generalization of a pretrained model is useful for finetuning a specific task. Deduplicating based on the threshold of around 0.7 seems ideal, but it would be better to decide the value considering the type of the downstream task.



Original	Deduplication above				
	TH 0.9	TH 0.8	TH 0.7	TH 0.6	TH 0.5
22.084	21.716	22.220	**22.434**	22.337	22.298

Table 2. Zero-shot accuracy results of the GPT-2 models with the LAMBADA dataset. Pretrained with a deduplication threshold of 0.7 gives the best score.



Data	STSB	COLA	MRPC	RTE
Original	0.853698	0.363870	0.816176	0.646209
TH 0.9	0.849696	0.335603	**0.818627**	0.675090
TH 0.8	0.857619	0.351343	0.816176	**0.700361**
TH 0.7	**0.861444**	**0.377724**	**0.818627**	0.689531
TH 0.6	0.857886	0.362333	0.811275	0.667870
TH 0.5	0.847854	0.343924	0.808824	0.657040

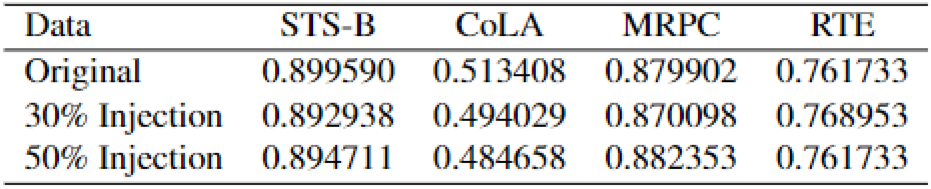
Table 3. GLUE Test results of GPT-2 models.

3.2. Duplication Injection Results

While GPT-2 shows consistent results, BERT does not perform better in models pretrained with deduplicated data. We conduct an additional experiment with BERT to see how duplicated data impacts BERT. We prepare a synthetic dataset based on the deduplicated dataset with the threshold 0.7 (we will call it original in this case), injecting exact document duplicates by random selection without allowing replacements to make it as close to the actual case as possible. To see a clear outcome, we use extreme ratios of data duplication 30% and 50%.

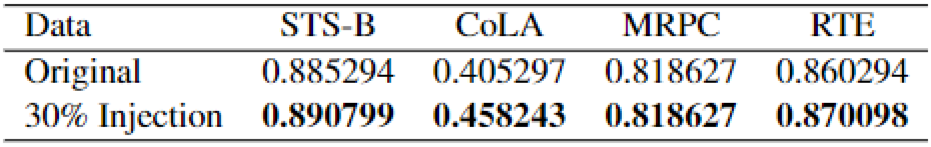
The latest epoch of each pretrained BERT model is selected and compared to obtain the best performance from finetuning tasks. As shown in Table 4, results are similar between the original and duplicates. This corresponds to the results already obtained from the previous near-duplicates BERT experiment. The duplication effect vanishes and works similarly for the different finetuning tasks.

Table 5 shows the results of the first few epochs (3 to 5, depending on the tasks) of pretraining the BERT models of the original and 30% duplication injection. We see that the model performs better with the duplication injection for all downstream tasks. Thus, the BERT model pretrained with duplicated data converges faster. This implies that duplication acts as data augmentation because of random masking in pretraining BERT. Duplication could help self-supervised learning rather than negatively affect the model to overfit.



Data	STS-B	CoLA	MRPC	RTE
Original	0.899590	0.513408	0.879902	0.761733
30% Injection	0.892938	0.494029	0.870098	0.768953
50% Injection	0.894711	0.484658	0.882353	0.761733

Table 4. GLUE Test results of the BERT models with duplication injection.



Data	STS-B	CoLA	MRPC	RTE
Original	0.885294	0.405297	0.818627	0.860294
30% Injection	**0.890799**	**0.458243**	**0.818627**	**0.870098**

Table 5. GLUE Test results of the BERT models with duplication injection after a few epochs.

Chapter 4. Conclusion

Our experimental results indicate that the duplicated data in language modeling work differently depending on the self-supervised learning methods used in pretraining the language model. Because of the randomness of the random masking in the training inputs to BERT, the duplicated data act as augmented data and improves the generalization capability of BERT. The duplicated data also makes BERT converge faster. If there is no randomness in the training method, as we see in GPT-2, deduplication gives better results. In addition, since data duplication makes the dataset larger than necessary, training takes longer, even if the performance of the models might be the same. Furthermore, data duplication negatively affects the GPT-2 model's performance by introducing biases. The model memorizes the severely duplicated data. In this case, applying the near-deduplication method, MinhashLSH, is compelling. The experimental results indicate that the Jaccard similarity threshold value of around 0.7 is an ideal option.

4.1. Discussion and Future work

Using more extensive and various datasets could be a further investigation direction for finding the effects of duplicated data. In addition, other than BERT and GPT-2, for instance, T5 could also be selected as a pretraining model. To compare with our results on BERT, a different self-supervised method, such as dynamic masking used in model Roberta, could be used to see the effect of duplicated data. Lastly, random seeds are used for initializing within a consecutive process (deduplication, pretraining and finetuning), even though it is reproducible, it is intrinsically difficult to completely rule out the influence of randomness. Therefore, testing more than once and finding consistent result are necessary.

As GPT-2 results reveal that a model memorizes duplicates, if a downstream task is related to the pretraining content, not the tasks that are testing the general language understanding, it may show better performance. Duplication, therefore, could be used as a method for a domain specific task. However, in terms of learning a language, prepared dataset is better deduplicated to be used in training generally shown in our analysis.

Bibliography

[1] J. Bandy and N. Vincent. Addressing “documentation debt” in machine learning research: A retrospective datasheet for bookcorpus. *arXiv preprint arXiv:2105.05241*, 2021.

[2] A. Z. Broder. On the resemblance and containment of documents. In Proceedings. Compression and Complexity of SEQUENCES 1997 (Cat. No. 97TB100171), pages 21–29. IEEE, 1997.

[3] C. Chelba, T. Mikolov, M. Schuster, Q. Ge, T. Brants, P. Koehn, and T. Robinson. One billion word benchmark for measuring progress in statistical language modeling. arXiv preprint arXiv:1312.3005, 2013.

[4] J. Ciro, D. Galvez, T. Schlippe, and D. Kanter. Lsh methods for data deduplication in a wikipedia artificial dataset. arXiv preprint arXiv:2112.11478, 2021.

[5] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805, 2018.

[6] S. Y. Feng, V. Gangal, J. Wei, S. Chandar, S. Vosoughi, T. Mitamura, and E. Hovy. A survey of data augmentation approaches for nlp. arXiv preprint arXiv:2105.03075, 2021.

[7] T. Gao, X. Yao, and D. Chen. Simcse: Simple contrastive learning of sentence embeddings. arXiv preprint arXiv:2104.08821, 2021.

[8] E. Grave, P. Bojanowski, P. Gupta, A. Joulin, and T. Mikolov. Learning word vectors for 157 languages. arXiv preprint arXiv:1802.06893, 2018.

[9] B. Gyawali, L. Anastasiou, and P. Knoth. Deduplication of scholarly documents using locality sensitive hashing and word embeddings. 2020.

[10] X. Han, Z. Zhang, N. Ding, Y. Gu, X. Liu, Y. Huo, J. Qiu, Y. Yao, A. Zhang, L. Zhang, et al. Pre-trained models: Past, present and future. AI Open, 2:225–250, 2021.

[11] M. A. Hedderich, L. Lange, H. Adel, J. Str ¨otgen, and D. Klakow. A survey on recent approaches for natural language processing in low-resource scenarios. arXiv preprint arXiv:2010.12309, 2020.

[12] D. Jin, Z. Jin, J. T. Zhou, and P. Szolovits. Is bert really robust? a strong baseline for natural language attack on text classification and entailment. In Proceedings of the AAAI conference on artificial intelligence, volume 34, pages 8018–8025, 2020.

[13] M. Joshi, D. Chen, Y. Liu, D. S. Weld, L. Zettlemoyer, and O. Levy. Spanbert: Improving pre-training by representing and predicting spans. Transactions of the Association for Computational Linguistics, 8:64–77, 2020.

[14] M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. arXiv preprint arXiv:1705.03551, 2017.

[15] N. Kandpal, E. Wallace, and C. Raffel. Deduplicating training data mitigates privacy risks in language models. arXiv preprint arXiv:2202.06539, 2022.

[16] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut. Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942, 2019.

[17] K. Lee, D. Ippolito, A. Nystrom, C. Zhang, D. Eck, C. Callison-

Burch, and N. Carlini. Deduplicating training data makes language models better. arXiv preprint arXiv:2107.06499, 2021.

[18] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. arXiv preprint arXiv:1907.11692, 2019.

[19] R. T. McCoy, P. Smolensky, T. Linzen, J. Gao, and A. Celikyilmaz. How much do language models copy from their training data? evaluating linguistic novelty in text generation using raven. arXiv preprint arXiv:2111.09509, 2021.

[20] D. Paperno, G. Kruszewski, A. Lazaridou, Q. N. Pham, R. Bernardi, S. Pezzelle, M. Baroni, G. Boleda, and R. Fern ´andez. The lambada dataset: Word prediction requiring a broad discourse context. arXiv preprint arXiv:1606.06031, 2016.

[21] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, et al. Improving language understanding by generative pre-training. 2018.

[22] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. OpenAI blog, 1(8):9, 2019.

[23] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, P. J. Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. J. Mach. Learn. Res., 21(140):1–67, 2020.

[24] A. Rogers, O. Kovaleva, and A. Rumshisky. A primer in bertology: What we know about how bert works. Transactions of the Association for Computational Linguistics, 8:842–866, 2020.

[25] T. Searle, Z. Ibrahim, J. Teo, and R. Dobson. Estimating redundancy in clinical text. Journal of Biomedical Informatics, 124:103938, 2021.

[26] M. Shoeybi, M. Patwary, R. Puri, P. LeGresley, J. Casper, and B. Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism. arXiv preprint arXiv:1909.08053, 2019.

[27] J. R. Smith, H. Saint-Amand, M. Plamada, P. Koehn, C. Callison-Burch, and A. Lopez. Dirt cheap web-scale parallel text from the common crawl. Association for Computational Linguistics, 2013.

[28] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. Advances in neural information processing systems, 30, 2017.

[29] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461, 2018.

[30] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. Advances in neural information processing systems, 32, 2019.

[31] X. Ye, X. Xue, W. Tian, Z. Xu, W. Xiao, and R. Li. Chunk content is not enough: Chunk-context aware resemblance detection for deduplication delta compression. arXiv preprint arXiv:2106.01273, 2021.

[32] R. Zellers, A. Holtzman, H. Rashkin, Y. Bisk, A. Farhadi, F. Roesner, and Y. Choi. Defending against neural fake news. Advances in neural information processing systems, 32, 2019.

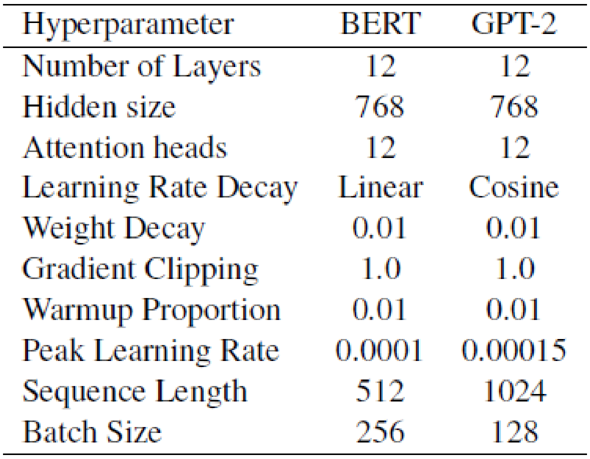
[33] Y. Zhu, R. Kiros, R. Zemel, R. Salakhutdinov, R. Urtasun, A. Torralba, and S. Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In Proceedings of the IEEE international conference on computer

vision, pages 19–27, 2015.

Appendix

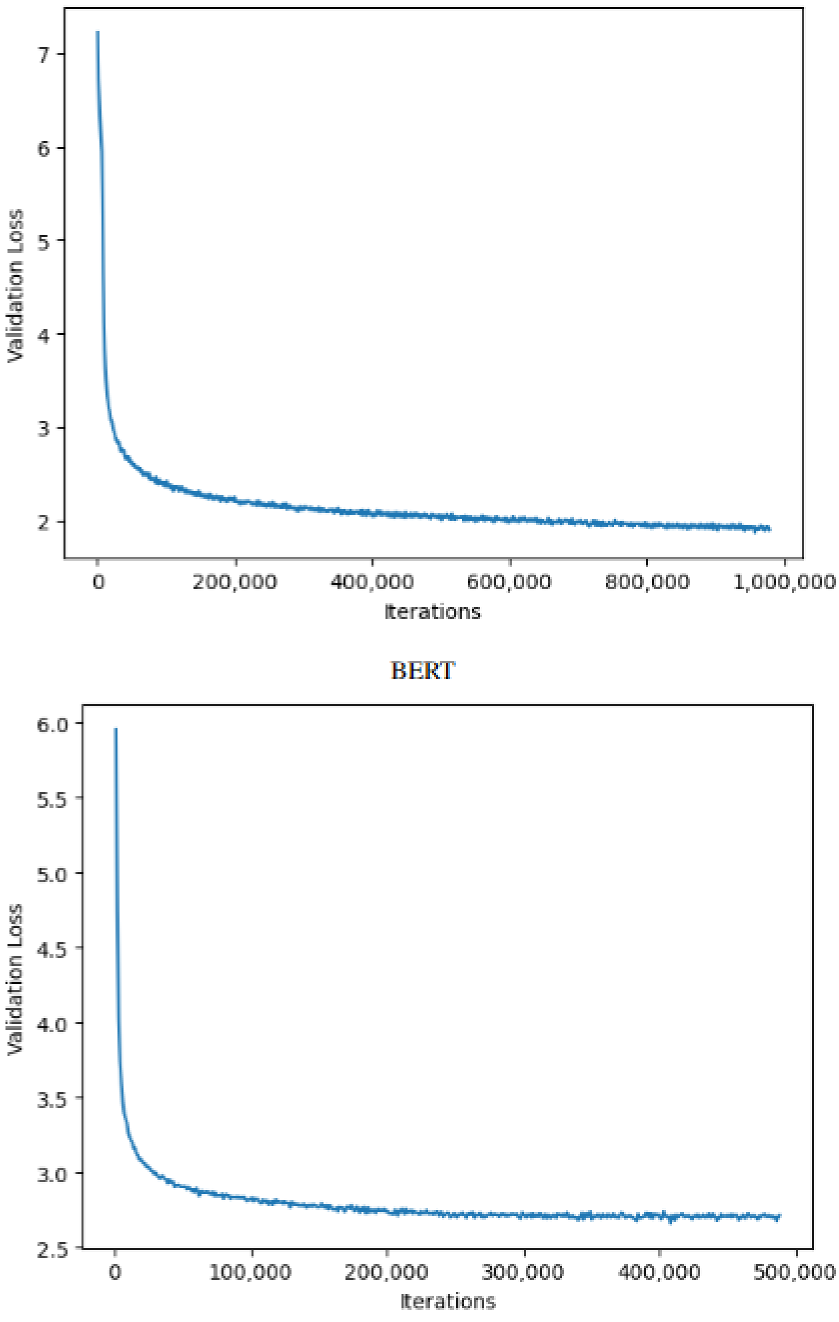
A. Pretraining Details

Table 6 presents the set of hyperparameters used in pretraining BERT and GPT−2 models. GPT−2 models are trained for 500,000 iterations and converge within 500,000 iterations. Depending on the models, 15 to 18 epochs are within 500,000 iterations. BERT models are trained for 1,000,000 iterations, converge within 1,000,000 iterations and correspond to 21 to 24 epochs. Figure 3 shows each model's validation loss along iterations. The validation dataset is a split of the deduplicated dataset with a threshold of 0.5.



Hyperparameter	BERT	GPT-2
Number of Layers	12	12
Hidden size	768	768
Attention heads	12	12
Learning Rate Decay	Linear	Cosine
Weight Decay	0.01	0.01
Gradient Clipping	1.0	1.0
Warmup Proportion	0.01	0.01
Peak Learning Rate	0.0001	0.00015
Sequence Length	512	1024
Batch Size	256	128

Table 6. Hyperparameters used for pretraining models.



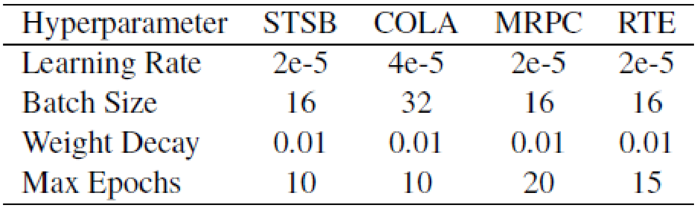
BERT

GPT-2

Figure 3. The validation losses of BERT and GPT−2.

B. Finetuning Details

Table 7 presents a set of hyperparameters for finetuning the models on the downstream tasks. We ran each model twice with two different seeds to get the best performance.



Hyperparameter	STSB	COLA	MRPC	RTE
Learning Rate	2e-5	4e-5	2e-5	2e-5
Batch Size	16	32	16	16
Weight Decay	0.01	0.01	0.01	0.01
Max Epochs	10	10	20	15

Table 7. Hyperparameters used for finetuning models on downstream tasks.

Abstract

이 연구는 BERT(인코더 기반 모델) 및 GPT-2(디코더 기반 모델)와 같은 언어 모델에 대한 훈련 데이터의 중복 제거 효과를 제시하는 데 목적이 있다. 기존 연구에서는 생성 모델에 한하여 중복 제거의 이점을 밝혔으며, 모델이 암기된 텍스트를 덜 생성하고 모델의 훈련 단계가 더 적게 필요하다는 것을 발견하였다. 이에 덧붙여 현 연구에서는 데이터 중복 제거에 대해 몇 가지 추가적인 실험을 수행한다. 사전 학습 데이터는 우선 MinhashLSH(대규모 말뭉치 데이터에서 유사한 문서를 찾기 위한 확률론적 방법)로 클러스터링 한 다음, 다양한 임계값의 Jaccard 유사성으로 중복 document를 제거하는 전처리 과정을 거친다. 구성된 데이터셋을 기반으로 사전 학습을 진행하고, 이후 다양한 downstream 작업에 finetuning한다. GPT-2는 중복 제거된 모델에서 더 높은 성능을 내는 반면, BERT는 downstream 작업에 따라 다른 성능을 보인다. 이는 BERT와 GPT-2의 self-supervised learning 방식의 차이 때문이다. BERT에서는 데이터 전처리 단계에서 랜덤 마스킹 방식을 통해 중복된 데이터가 오히려 데이터 augmentation으로 작용할 수 있다. 그렇지만 결과적으로 데이터 중복은 편향을 도입하고 과적합으로 이어질 수 있으며, 그 효과는 중복 데이터의 양에 따라 다를 수 있다. 따라서 성능을 향상시키기 위해선 언어 모델 훈련에서 적절한 임계값의 데이터 중복 제거가 필수적이다.