Training Data is More Valuable than You Think: A Simple and Effective Method by Retrieving from Training Data

Shuohang Wang, Yichong Xu, Yuwei Fang, Yang Liu, Siqi Sun, Ruochen Xu, Chenguang Zhu, Michael Zeng
Microsoft Azure Cognitive Services Research
{shuowa, yicxu, yuwfan, yaliu10, siqisun, ruox, chezhu, nzeng}@microsoft.com

Abstract

Retrieval-based methods have been shown to be effective in NLP tasks via introducing external knowledge. However, the indexing and retrieving of large-scale corpora bring considerable computational cost. Surprisingly, we found that **RE**trieving from the tra**IN**ing dat**A** (**REINA**) only can lead to significant gains on multiple NLG and NLU tasks. We retrieve the labeled training instances most similar to the input text and then concatenate them with the input to feed into the model to generate the output. Experimental results show that this simple method can achieve significantly better performance on a variety of NLU and NLG tasks, including summarization, machine translation, language modeling, and question answering tasks. For instance, our proposed method achieved state-of-the-art results on XSum, BigPatent, and CommonsenseQA. Our code is released.[1]

1 Introduction

In natural language processing, retrieval-based methods work by fetching textual information related to the input from large corpora. The model then takes both the input and retrieved results as input to generate results. This can often improve the performance as the model is exposed to related knowledge not present in the input. As a result, retrieval-based methods have been successfully applied in many tasks such as open-domain question answering (Chen et al., 2017), language modeling (Guu et al., 2018; Khandelwal et al., 2020) and machine translation (Khandelwal et al., 2021). However, these methods require building an index of large-scale corpus, and the retrieval leads to a significant computational burden. For example, the kNN-MT model for machine translation has a generation speed two orders of magnitude slower than traditional MT models (Khandelwal et al., 2021).

On the other hand, in the supervised learning setting, the text most similar in distribution to the data in inference is the training data. Thus, we explore whether retrieving from the training data, which is usually much smaller than a large-scale corpus, can help improve the performance. Specifically, we first index a task's labeled training data as input-label pairs. Then, during both training and testing, we retrieve the input-label pairs most similar to the current input[2]. Finally, we concatenate the retrieved training pairs with the input and feed it into the model. An overview of our method is shown in Figure 1.

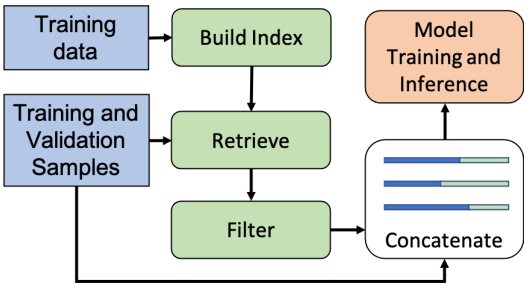

Figure 1: REINA pipeline of model training/inference with retrieval from training data. Filter only happens at training, as the same training sample will be retrieved from the index. For each instance, we concatenate the input with the retrieved content, i.e., data and/or labels, for model training and inference.

We note that our method is similar to recent works in prompt learning (Brown et al., 2020; Liu et al., 2021), where a set of labeled data is carefully chosen based on the input and then included in the prompt for few-shot learning. Our method also bears a resemblance to non-parametric instance-based learning (Gu et al., 2018). However, a critical difference is that we focus on the supervised learning setting, where the model parameters are fine-tuned to learn from given examples to achieve much higher performance than few-shot learning or non-parametric methods.

In the experiments, we evaluate our method

[1] https://github.com/microsoft/REINA

[2] During training, we exclude the training instance itself from the retrieval results to avoid data leakage.

Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics
Volume 1: Long Papers, pages 3170 - 3179

on four popular types of NLP tasks: summarization, language modeling, machine translation, and question answering. We find that i) after integrating REINA, we can achieve significantly better performance on these tasks, 11 datasets in total, than models with different pre-trained models; ii) REINA leads to SOTA performance on the datasets of XSum, CommonsenseQA (Leaderboard No.1), and BigPatent; iii) REINA can scale up more easily by leveraging more labeled data from other datasets via retrieval, outperforming baselines which is trained on the same set of data. iv) the results on 3 summarization tasks show that BART-base with REINA rivals BART-large, which contains twice more parameters now.

The effectiveness of our approach on summarization tasks provides insights into the core of supervised learning. Even with hundreds of millions of parameters, a model cannot memorize all the patterns in the training data. Thus, recapturing related training data as a side-by-side reminder can explicitly provide needed information to enhance the model's performance at inference. It also points out that instead of building models of ever increasing sizes, we can make a decent-size model output high-quality results by leveraging those training data that resemble the instance at hand. This can significantly reduce the computational cost while achieving a similar or better performance of a mega-sized model.

2 Related Work

Retrieval-based Methods Even a pre-trained model as large as GPT-3 (Brown et al., 2020) cannot remember everything, and it is important to leverage information retrieval to collect external knowledge to solve different NLP tasks. There are two types of representations for retriever: bag-of-word (BOW) based sparse representation (Chen et al., 2017) and dense representation from neural networks (Karpukhin et al., 2020).

For the sparse representation, as the method is based on BOW and usually rule-based score, such as BM25, is used for ranking, it can be easily adapted to a general large-scale search. This method has also been widely explored to solve open domain question answering (Chen et al., 2017; Wang et al., 2018; Lin et al., 2018) and Machine Translation (Gu et al., 2018).

Dense representation based retrieval (DPR) (Karpukhin et al., 2020) is the most widely explored area in recent years. Dense representations come from encoders, such as Transformer, trained with task-specific data. And these methods can achieve better recall performance than sparse representation on different tasks, such as open domain question answering (Karpukhin et al., 2020; Guu et al., 2020; Yu et al., 2021), knowledge-grounded generation (Zhang et al., 2021), and machine translation (Cai et al., 2021). One drawback of DPR is that it cannot process longer documents, usually less than 128 tokens (Karpukhin et al., 2020). Another drawback is that it needs parallel data for model training on specific tasks.

Considering the generalization and efficiency of sparse representation, in this paper, we use BM25 score (Robertson and Zaragoza, 2009; Schütze et al., 2008) to retrieve from the training data, and our method is more flexible with no requirement of parallel data for model training. Compared to non-parametric systems guided by search engine (Gu et al., 2018; Khandelwal et al., 2020), our proposed method is based on supervised learning and is more general. Lewis et al. (2021) is related to our work by retrieving related questions from pre-built large-scale question-answer pairs. However, our method doesn't need addition data augmentation method, and we have successfully applied REINA to a wide range of downstream tasks, including summarization, question answering, machine translation and language modeling.

Prompt Engineering With the success of large-scale language models (Brown et al., 2020) on few-shot learning, prompt engineering comes to be a popular research direction. The idea is to prepend several labeled instances to the input sequence and then conduct the classification or generation. Liu et al. (2021) proposes to prepend the most related labeled data as prompt to help fewshot inference. Li and Liang (2021) optimizes the prompt in continuous space. Motivated by these works where a good labeled prompt can help fewshot learning, we also prepend/append the most similar labeled training data for all the data in training, validation, and test set. However, different from prompt learning, we focus on supervised learning settings.

3 Model

In this section, we will introduce the details of our proposed method. Briefly, given the input, we first retrieve the most matched instances with labels

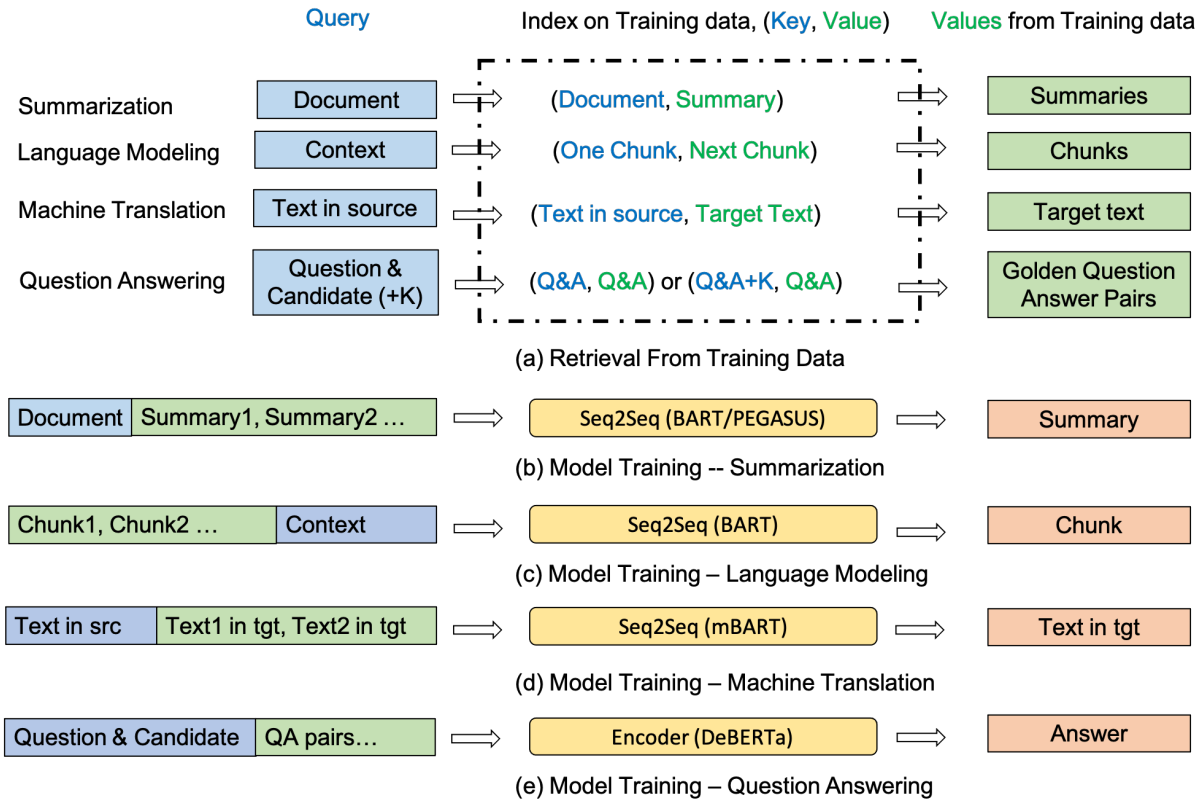

Figure 2: Model training with retrieval from the training data (REINA). (a) Index on the training data and data retrieval for 4 different tasks. Box in blue is the query or the input sequence to encode. Box in green is the retrieved text. (b-e) Leveraging retrieved data for model training with different structures. For language modeling, we prepend the retrieved data to the query data, and append the retrieved data to the query for all the other tasks. After concatenation, we will directly feed them into Transformers, either Seq2Seq or Encoder-only frameworks, for text generation and answering selection. As we focus on the question answering tasks requiring commonsense reasoning, we have another version of index integrating knowledge graph for more precise retrieval. K: external knowledge from ConceptNet and Wiktionary, src: source language, tgt: target language.

from the training data. We then concatenate them with the input sequence to feed into the model for generating the output. An overview of the whole method is shown in Figure 2.

3.1 Retrieval-based Methods

A retrieval-based method collects information most similar to the input from a corpus and then combines it with the input to feed into the NLP model. Suppose we index the corpus into a list of key-value pairs, i.e. $\mathcal{C} = \{(k_i, v_i)\}$. Then, given the input x, the retrieval engine $\mathcal{E}$ matches it with all keys and returns the top K most similar keys to the query together with their values:

$$\{(k_{i_1}, v_{i_1}), ..., (k_{i_K}, v_{i_K})\} = \mathcal{E}(x|\mathcal{C}) \quad (1)$$

In this work, we build the retrieval engine based on the widely used BM25 score (Schütze et al., 2008). We choose BM25 over dense representation mainly for its faster speed.

Then, these retrieved results are combined with the input x to feed into the NLP model $\mathcal{M}$ to generate the output O:

$$O = \mathcal{M}(f(x, \{(k_{i_1}, v_{i_1}), ..., (k_{i_K}, v_{i_K})\}) \quad (2)$$

Here, the combination function f can be concatenation, e.g. $f(x, \{(k_{i_1}, v_{i_1}), ..., (k_{i_K}, v_{i_K})\}) = [x; v_{i_1}; ...; v_{i_K}]$. As data in different tasks is organized in different formats with varying lengths, we will introduce how we define different combination functions f for various tasks in the follows.

3.2 Retrieval from Training Data (REINA)

As retrieval from a large corpus is computationally costly, we propose to retrieve from the labeled training data. In other words, we directly adopt the training data $\mathcal{T} = \{(x_1, y_1), ..., (x_N, y_N)\}$ as the indexed corpus $\mathcal{C}$, where x_i is the input and y_i is the ground-truth label.

Given an input x, the top K retrieved training instances with labels are combined with x as input to the model $\mathcal{M}$, i.e., $\mathcal{M}(f(x, \{(x_{i_1}, y_{i_1}), ..., (x_{i_K}, y_{i_K})\}$. Both training and inference take this retrieve-combine-generate scheme. Note that during training, as the input x is already indexed, we filter it from the retrieval results to avoid data leakage.

Now, we introduce how we define the keys, values, and the combination function for different NLP tasks.

Summarization is to generate a summary for a given document. We first build an index for the document-summary pairs in the training data, where a document is the key and its summary is the value. Given a document x, we search for the most similar documents in the index. As documents are usually quite long, the combination function only keeps the values (summaries), i.e., $f_{summ}(x, \{(x_{i_1}, y_{i_1}), ..., (x_{i_K}, y_{i_K})\}) = [x; y_{i_1}; ...; y_{i_K}]$.

Language Modeling (LM) generates the probability of a given sequence of words. Typically, a Left-to-Right language model (Dong et al., 2020) is trained on chunked sequences with an attention mask. In this paper, we use Seq2Seq based approach, i.e., given a context chunk, we predict the next chunk of text.

In detail, we first chunk all the text in the training data. The IR index is built with one chunk C_i as the key x_i and its next chunk C_{i+1} as the value y_i. Given a chunk x, we look for the most similar keys in the index and prepend their corresponding next chunks to x, i,e., $f_{LM}(x, \{(x_{i_1}, y_{i_1}), ..., (x_{i_K}, y_{i_K})\}) = [y_{i_1}; ...; y_{i_K}; x]$.

Machine Translation is to translate text from the source language $\mathcal{S}$ to the target language $\mathcal{T}$. We define the key to be the sentence in $\mathcal{S}$ and the value to be its translation in $\mathcal{T}$. To keep the sequence short and speed up the training process, we only concatenate the retrieved text in target language: $f_{MT}(x, \{(x_{i_1}, y_{i_1}), ..., (x_{i_K}, y_{i_K})\}) = [x; y_{i_1}; ...; y_{i_K}]$.

Question Answering We mainly consider multiple-choice question answering, where commonsense knowledge is also required to reach the correct answer. For each question x_i, there is a correct choice y_i and several distractive candidate choices. We index the concatenation of the question and the corresponding ground-truth choice. For a new question x, the model is given several choices $c_1, ..., c_M$. We concatenate x with each choice c_i as the query and retrieve related training instances: $\{(x_{i_1}, y_{i_1}), ..., (x_{i_K}, y_{i_K})\} = \mathcal{E}(x; c_i|\mathcal{C})$. The combination function f concatenates both retrieved question and answers with the input: $f_{QA}((x, c_i), \{(x_{i_1}, y_{i_1}), ..., (x_{i_K}, y_{i_K})\}) = [x; c_i; x_{i_1}; y_{i_1}; ...; x_{i_K}; y_{i_K}]$. Then, the model predicts a score representing how likely c_i is the correct choice to x.

As the task requires commonsense knowledge, we build another version of index integrating commonsense knowledge. We follow the strategy from (Xu et al., 2021) and extract the knowledge from ConceptNet (Speer et al., 2017) and Wiktionary[3] for the concepts in the question and choices. For each question x and choice c, we use string match to find corresponding entities in ConceptNet: $E^{(x)} = \{e_1^{(x)}, ..., e_{n_x}^{(x)}\}$ appears in the question, and $E^{(c)} = \{e_1^{(c)}, ..., e_{n_c}^{(c)}\}$ appears in the answer. To find the most relevant concept, we choose the concept with maximum length as the question and answer concept. We find the definition of the chosen concepts from Wiktionary. To find relations in ConceptNet, we find edges that connects question and answer concepts: $R = \{(e_1, r, e_2)|e_1 \in E^{(x)}, e_2 \in E^{(c)}, (e_1, e_2) \in \mathcal{KG}\}$. Here $\mathcal{KG}$ is ConceptNet and r is a relation (e.g., `AtLocation`). We concatenate the Wiktionary definitions and ConceptNet relations R to form the knowledge, $\mathcal{K}$, for a question. The knowledge $\mathcal{K}$ is included both in the query and index. Thus, the retrieval process becomes: $\{(x_{i_1}, c_{i_1}, \mathcal{K}_{i_1}), ..., (x_{i_K}, y_{i_K}, \mathcal{K}_{i_K})\} = \mathcal{E}(x; c_i; \mathcal{K}|\mathcal{C})$. The combination function f concatenates retrieved questions and answers with the input: $f_{QAK}((x, c_i), \mathcal{E}(x; c_i; \mathcal{K}|\mathcal{C})) = [x; c_i; x_{i_1}; y_{i_1}; ...; x_{i_K}; y_{i_K}]$.

3.3 Model Training and Inference

After concatenating the input with the retrieved data from the training corpus, we feed the new sequence into the Seq2Seq framework for generation tasks and the encoder-only framework for question answering tasks. During training, as it will also retrieve the exact golden label, we filter it directly. During inference, we will not filter any retrieved information, as all the retrieve data only come from training set.

Task	Dataset	Train	Dev	Test
Summarization	Multi-News	45k	5.6k	5.6k
	WikiHow	168k	6k	6k
	XSum	204k	11k	11k
	NEWSROOM	993k	108k	108k
	BigPatent	1,207k	67k	67k
Language Modeling	WikiText2	32k	3.3k	3.8k
	WikiText103	801k	1.7k	1.9k
Machine Translation	WMT16 (en-tr)	205k	1k	3k
	WMT16 (en-de)	4,548k	2.2k	3k
Question Answering	CSQA	9.7k	1.2k	1.1k
	PIQA	16k	1.8k	3.4k
	aNLI	170k	1.5k	3.0k

Table 1: Statics of the evaluation datasets. The table shows the number of data in training, dev, and test sets. As we treat the language model as a Seq2Seq problem, the number here is the chunked sequences, each of which contains 64 words for WikiText2 and 128 words for WikiText103.

4 Experiment

In this section, we will introduce more details about experiments and the corresponding analysis.

4.1 Dataset

We evaluate REINA on 4 different tasks with 12 datasets as shown in Table 1.

Summarization We evaluate our method on 5 summarization datasets: 1) **XSum** (Narayan et al., 2018), extreme summarization, is a task of one sentence summarization on one document. The document comes from British Broadcasting Corporation (BBC) online articles. 2) **NEWSROOM** (Grusky et al., 2018) is a summarization dataset on a larger scale and the articles with human-written summaries come from 38 major news publications. 3) **Multi-News** (Fabbri et al., 2019) is a task of multi-document summarization on news articles from the site newser.com. 4) **BigPatent** (Sharma et al., 2019) is constructed on U.S. patent documents along with human written abstracts. The documents cover broader areas in 9 different categories. Another domain, 5) **WikiHow** (Koupaee and Wang, 2018) is to summarize the steps of "How to" solve a problem. The dataset consists of more diverse style articles written by ordinary people. Besides the above datasets, we also introduce CNN/Dailymail (Nallapati et al., 2016) and 160G BART pretraining corpus (Lewis et al., 2020) from BOOKCORPUS, CC-NEWS, OPENWEBTEXT, and STORIES, to scale up the training corpus.

Language Modeling As our model is initialized by a pre-trained model, we select two language modeling datasets, the corpus of which is not used for model pre-training. The text of both datasets, **WikiText103** (Merity et al., 2017) and **WikiText2** (Merity et al., 2017), are extracted from Wikipedia. As the dataset's text is at a document level, the tasks focus on testing the model's ability to remember longer sequences.

Machine Translation We evaluate our method on the translation of English-German and English-Turkish in both directions from WMT16 (Bojar et al., 2016).

Question Answering We have 3 question answering datasets to evaluate our method: 1) **CommonsenseQA** (CSQA, Talmor et al., 2019) is a dataset for commonsense multi-choice question answering. The questions are generated based on commonsense knowledge base, ConceptNet. 2) **Physical IQA** (PIQA, Bisk et al., 2020) is to answer questions requiring physical commonsense reasoning. 3) **Abductive NLI** (aNLI, Bhagavatula et al., 2020) is a multiple-choice question answering task for choosing the more likely explanation. All these tasks are challenging by requiring commonsense knowledge to reach the correct answer.

4.2 REINA Details

For the task of summarization, instead of directly retrieving the most relevant summary (An et al., 2021), we find the most relevant documents by BM25 score and then leverage the corresponding summaries. Compared to the dense passage retrieval based method, our method can handle the long document retrieval and does not need to train. Moreover, REINA is easier to scale up. We also consider joint training baseline on Summarization tasks. Our setting is to test how other datasets can help improve XSum. For REINA, we build index on summarization datasets from different sources. During model training, we will only train models with the XSum dataset along with retrieved data appended to the documents.

For language modeling task, instead of working on word-level retrieval by KNN (Khandelwal et al., 2020), we chunk all the training data. During

[3] https://www.wiktionary.org/

	BigPatent			XSum			WikiHow			Multi-News			NEWSROOM		
	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L	R-1	R-2	R-L
Earlier SOTA	37.5	10.6	22.7	45.1	22.2	37.2	28.5	9.2	26.5	43.4	14.8	17.4	39.9	28.3	36.8
PEGASUS	53.6	33.2	42.3	47.2	24.6	39.3	43.1	19.7	34.8	**47.5**	**18.7**	**24.9**	**45.2**	**33.5**	**41.3**
PEGASUS	38.4	13.5	26.3	46.6	23.9	38.6	35.9	15.3	30.3	43.1	15.4	22.6	41.7	30.7	37.8
REINA (PG)	44.6	21.5	33.0	**48.2**	**26.0**	**40.2**	36.8	16.7	31.0	45.0	17.1	23.8	41.4	30.5	37.5
BART-base	44.2	16.9	28.4	41.0	18.2	33.3	43.3	18.1	33.9	44.8	16.4	23.3	41.3	29.1	37.5
REINA (B)	59.5	42.6	50.6	43.2	21.0	35.5	44.2	19.4	34.9	45.1	16.9	23.6	41.2	29.0	37.5
BART-large	44.9	17.5	28.9	44.7	21.6	36.5	43.4	19.0	34.9	44.1	16.6	22.7	41.6	29.4	38.0
REINA (L)	**60.7**	**43.3**	**51.3**	46.5	24.1	38.6	**44.2**	**20.4**	**35.8**	46.9	17.7	24.0	42.5	30.2	38.7

Table 2: Summarization results. In the top section, we report the results from PEGASUS (Zhang et al., 2020) paper. In the bottom, we reproduce three strong baselines with PEGASUS and BART (Lewis et al., 2020), and show our REINA initialized by the same pre-trained models for fair comparison. The bolded numbers show the SOTA performance and the underlined numbers show the best performance with BART initialization. PEGASUS: PEGASUS-large, B: BART-base, L: BART-large, R-1: Rouge-1, R-2: Rouge-2, R-L: Rouge-L

	XSum		
	R-1	R-2	R-L
BART (XSum)	44.7	21.6	36.5
BART (XSum+CNN)	44.6	21.6	36.9
REINA (XSum)	46.5	24.1	38.6
REINA (XSum+CNN)	47.5	**25.2**	**39.5**
REINA (XSum+NR)	47.5	24.9	39.4
REINA (XSum+160G)	**47.7**	25.1	**39.5**

Table 3: Evaluation on XSum test set with training data scale up. BART is jointly trained with datasets in bracket. REINA is trained with XSum document-summary pairs, but the index is built on the datasets in bracket. CNN: CNN/Dailymail dataset, NR: NEWSROOM dataset, 160G: BART pre-training corpus.

training, besides the retrieved chunks, we will also include the context of the query chunk to generate next chunk. Compared to KNN-LM (Khandelwal et al., 2020), REINA only needs retrieval once per chunk which is much more efficient.

For multi-choice question answering, we build two types of indexes with or without external knowledge from ConceptNet and Wiktionary. For the query, the concatenation of question and one candidate answer, we also have two versions, with or without knowledge. After adding knowledge, there would be more word overlaps when key concept words between questions are matched. The retrieved information will be treated as either a prompt or additional knowledge to encode together and then predicts the answer probability of each candidate.

4.3 Optimization Details

Our information retrieval is based on Lucene Index [4]. Our model training is based on Transformers library [5]. All our experiments are based on 8-GPU machines.

For summarization tasks, we initialized the model with three types of pre-trained models, PEGASUS-large (Zhang et al., 2020), BART-base, and BART-large (Lewis et al., 2020). Optimization is based on AdamW (Loshchilov and Hutter, 2019). We tune the hyper-parameters from learning rate {2e-05, 5e-05, 7e-05}, and set dropout 0.1, batch size 32. For both baseline and our method, we set the maximal length of the input sequence to be 1024. We use the original document to generate summary in baselines. For REINA, we set the maximal length of the original document 600 and then append the top-5 retrieved summaries from training data.

For language modeling tasks, we initialized the model with BART-base and BART-large. We set the number of words in each chunk to 128 for WikiText103 and 64 for WikiText2. For each chunk generation, we set the context length of baseline methods 1024. For our method, we set the context 512 and prepend the retrieved text. The maximal length of the concatenated sequence is 1024. We use optimizer Adam (Kingma and Ba, 2015) with learning rate 5e-05, dropout 0.1, batch size 32.

For machine translation tasks, we initialized the model with mBART-large (Liu et al., 2020). We

[4]https://lucene.apache.org/pylucene/

[5]https://github.com/huggingface/transformers

	CSQA	aNLI	PIQA
Dev Set results			
DeBERTa	84.0	88.8	85.6
REINA (w/o K)	**88.8**	88.6	85.5
REINA (w/ K)	86.8	**89.6**	**86.9**
Test Set results			
CALM	71.8	82.4	76.9
UNICORN	79.3	87.3	**90.1**
DEKCOR	83.3	-	-
DeBERTa	-	86.8	85.1
REINA	**84.6**	**88.0**	85.4

Table 4: Question answering results. CALM (Zhou et al., 2021) is continue-pretrained from RoBERTa-large model. UNICORN (Lourie et al., 2021) and DEKCOR (Xu et al., 2021) use the T5-11B model. Our DeBERTa baseline is close to DEKCOR but with different pretrained initializations. REINA is also based on DeBERTa. We first evaluate REINA on dev set to verify whether integrating external knowledge in REINA can lead to better performance. And then submit the best one for hidden test set evaluation. We achieve leaderboard No.1 on CommonsenseQA. K: external knowledge from ConceptNet and Wiktionary.

	WikiText103	WikiText2
Transformer-XL	18.30	-
kNN-LM	15.79	-
GPT-2	17.48	18.34
BART-Base	15.88	20.41
REINA (B)	14.76	20.78
BART-Large	12.10	**15.11**
REINA (L)	**11.36**	15.62

Table 5: Language modeling results. The evaluation metric is perplexity (PPL). The top part of the table comes from the original papers, Transformer-XL (Dai et al., 2019), kNN-LM (Khandelwal et al., 2020), GPT-2 (Radford et al., 2019). The bottom part is our implementation with fair comparison. B: BART-base, L: BART-large

	WMT16			
	en2tr	tr2en	en2de	de2en
XLM	-	-	26.4	34.3
mBART	18.4	23.1	32.6	37.0
REINA	**18.8**	**23.6**	**32.9**	37.0

Table 6: Machine translation on WMT16. We compare with baselines XLM (Lample and Conneau, 2019) and mBART (Liu et al., 2020). REINA is initialized by mBART for fair comparison. The evaluation metric is based on SacreBLEU. Source and target languages are concatenated by "2". tr: Turkish, de: German, en: English.

follow the hyper-parameter setting from the original paper with Adam optimizer, dropout 0.3, label smoothing 0.2, warm-up steps 2500, maximum learning rate 3e-05, and training updates 40K in total.

For question answering datasets, our method is based on DeBERTa (He et al., 2021) with 1.5B parameters. We use optimizer AdamW (Loshchilov and Hutter, 2019) with learning rate 3e-06, batch size 8. As the datasets requiring commonsense reasoning, we also leverage knowledge bases, ConceptNet and Wiktionary, in REINA .

4.4 Experiment Results

Our experiment results on the summarization tasks are shown in Table 2. Our evaluation metric is based on Rouge-1/2/L scores, same as PEGASUS (Zhang et al., 2020). We have a broad experiment on 5 datasets, ranging from single document summarization (XSum) to multi-document summarization (Multi-News), from news domain to wiki knowledge (WikiHow) and patent (BigPatent) domains. We re-run all of our baseline methods. Based on the experiment results, we find that REINA can significantly boost the baselines initialized with different pre-trained models, such as PEGASUS, BART-base, and BART-large, on all 5 datasets. Besides, our method with BART-large can achieve state-of-the-art performance on XSum and BigPatent datasets. Moreover, we find REINA can help base models beat larger models. For example, REINA (BART-base) is better than both PEGASUS-LARGE and BART-large on BigPatent and WikiHow datasets.

We also evaluate the ability of REINA on learning from more related datasets. Our experiment results are shown in Table 3. The evaluation is conducted on XSum test set and we use three related data sources from CNN/Dailymail, NEWSROOM, and a 160G raw-text corpus[6]. Based on the experiments, we can see that simply training the model on merged dataset (XSum + other sources) doesn't lead to any gains. However, after adding one additional data source to build index and applying

[6]For the 160G data, we treat the first sentence as summary and the rest as document.

Document	No international side has toured Bangladesh since 20 people were killed in a siege at a cafe in Dhaka in July.The England and Wales Cricket Board said in August that tour would go ahead following a security review ...
Summary	England one-day captain Eoin Morgan and opening batsman Alex Hales have opted out of October's tour of Bangladesh because of security concerns.
REINA 1	England one-day captain Eoin Morgan says he will never again go on a tour where security concerns may affect his game.
REINA 2	Eoin Morgan and Alex Hales remain "very much part of the group" despite not touring Bangladesh, says stand-in England one-day captain Jos Buttler.
Question	Brawn opened the curtains so that the sun could do what?
Answer	REINA chooses: **warm room**, Baseline chooses: *shine brightly*
REINA 1	What effect did the sun have on the residents inside? warm house.
REINA 2	James installed his new curtains to keep the light from shinning on his television. Where is James probably hanging his curtains? house.

Table 7: Examples from dev sets and the corresponding labeled data retrieved from training set. The top case comes from a summarization task, XSum. The bottom case comes from a question answering task, CommonsenseQA. For summarization tasks, we will only append the document with the retrieved summaries. For CommonsenseQA, we will append the golden QA pairs to the question. The golden answer is "warm room". REINA 1/2 refers to different retrieved data.

REINA, there's 1% improvement in Rouge scores[7]. Overall, our REINA can effectively leverage the most relevant data from additional datasets while being trained only on the target task.

For question answering tasks, our results are shown in Table 4. We test REINA on three datasets, where commonsense knowledge is usually required to answer the question. Thus we first verify whether we need external knowledge during the retrieval. According to the experiments, we find that directly retrieving the labeled data without knowledge works best for CommonsenseQA dataset, but involving knowledge can help on aNLI and PIQA datasets. And REINA can significantly improve our baselines with DeBERTa on all the datasets. Moreover, after submitting our best results to the corresponding leaderboards, REINA achieves state of the art on CommonsenseQA dataset (Leaderboard No.1) and beat strong baselines on aNLI and PIQA datasets.

Our evaluation of language modeling is shown in Table 5. Our method can achieve significant improvement on WikiText103 dataset over both BART-base and BART-large baselines. However, it cannot lead to better performance on WikiText2. One reason may be that WikiText2 is a much smaller dataset, and it's hard for REINA to retrieve the most related text. Besides, we also find Seq2Seq model can be a very strong baseline which means we can leverage more pre-trained models such as PEGASUS, T5 (Raffel et al., 2030), and BART, for language modeling in future work. And Seq2Seq frame would be more flexible to integrate external knowledge to boost performance further.

For machine translation, we make use of the datasets from WMT16. We select one low-resource language, Turkish-English, and one rich-resource, German-English, for REINA evaluation, as shown in Table 6. We re-implement mBART baseline for translation in both directions. To make a fair comparison, REINA is also based on mBART. We can find that REINA can further boost performance under three settings, translating English to Turkish, Turkish to English, and English to German.

4.5 Further Analysis

We show a case study on the data retrieved by REINA . We list two cases from XSum and CommonsenseQA dev sets. From the case on summarization task, we can find that the first retrieved summary from training set, REINA 1, shows the same point of "security concerns" as the golden summary. And the other case on multi-choice question answering, REINA 1 suggests that the sun can warm up a place that shares the same commonsense knowledge to answer the question. After, although we cannot visualize how the neural encoders work by leveraging the retrieved data, we have shown that the data from REINA have very strong correlation with the golden labels.

5 Conclusion

In this paper, we propose a simple and effective method to fully make use training dataset. Our

[7]In our experiments, we follow Xu and Durrett (2021) by ignoring the retrieved data if there are over three 7-gram overlap between retrieved summary and golden summary.

proposed method is general and can be easily integrated into different models on different tasks. We prove that REINA can effectively improve baseline performance on 11 datasets covering summarization, language modeling, machine translation, and question answering tasks.

References

Chenxin An, Ming Zhong, Zhichao Geng, Jianqiang Yang, and Xipeng Qiu. 2021. Retrievalsum: A retrieval enhanced framework for abstractive summarization. *arXiv preprint arXiv:2109.07943*.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Wen-tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. *International Conference on Learning Representations (ICLR)*.

Yonatan Bisk, Rowan Zellers, Jianfeng Gao, Yejin Choi, et al. 2020. Piqa: Reasoning about physical commonsense in natural language. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Varvara Logacheva, Christof Monz, et al. 2016. Findings of the 2016 conference on machine translation. In *First Conference on Machine Translation*.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Neural Information Processing Systems (NeurIPS)*.

Deng Cai, Yan Wang, Huayang Li, Wai Lam, and Lemao Liu. 2021. Neural machine translation with monolingual translation memory. *Association for Computational Linguistics (ACL)*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *Association for Computational Linguistics (ACL)*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *Association for Computational Linguistics (ACL)*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2020. Unified language model pre-training for natural language understanding and generation. *International Conference on Machine Learning (ICML)*.

Alexander R Fabbri, Irene Li, Tianwei She, Suyi Li, and Dragomir R Radev. 2019. Multi-news: A large-scale multi-document summarization dataset and abstractive hierarchical model. *Association for Computational Linguistics (ACL)*.

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor OK Li. 2018. Search engine guided neural machine translation. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Kelvin Guu, Tatsunori B Hashimoto, Yonatan Oren, and Percy Liang. 2018. Generating sentences by editing prototypes. *Transactions of the Association for Computational Linguistics (TACL)*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *International Conference on Machine Learning (ICML)*.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. Deberta: Decoding-enhanced bert with disentangled attention. *International Conference on Learning Representations (ICLR)*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *Empirical Methods in Natural Language Processing (EMNLP)*.

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

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

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. *International Conference on Learning Representations (ICLR)*.

Mahnaz Koupaee and William Yang Wang. 2018. Wikihow: A large scale text summarization dataset. *arXiv preprint arXiv:1810.09305*.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *Neural Information Processing Systems (NeurIPS)*.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2020. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *Association for Computational Linguistics (ACL)*.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021. Paq: 65 million probably-asked questions and what you can do with them. *Transactions of the Association for Computational Linguistics*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. *Association for Computational Linguistics (ACL)*.

Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distantly supervised open-domain question answering. In *Association for Computational Linguistics (ACL)*.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *arXiv preprint arXiv:2101.06804*.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics (TACL)*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled weight decay regularization. *International Conference on Learning Representations (ICLR)*.

Nicholas Lourie, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. 2021. Unicorn on rainbow: A universal commonsense reasoning model on a new multitask benchmark. *AAAI Conference on Artificial Intelligence (AAAI)*.

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2017. Pointer sentinel mixture models. *International Conference on Learning Representations (ICLR)*.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *SIGNLL Conference on Computational Natural Language Learning (CoNLL)*.

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *Empirical Methods in Natural Language Processing (EMNLP)*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2030. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research (JMLR)*.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*. Now Publishers Inc.

Hinrich Schütze, Christopher D Manning, and Prabhakar Raghavan. 2008. *Introduction to information retrieval*. Cambridge University Press Cambridge.

Eva Sharma, Chen Li, and Lu Wang. 2019. Bigpatent: A large-scale dataset for abstractive and coherent summarization. *Association for Computational Linguistics (ACL)*.

Robyn Speer, Joshua Chin, and Catherine Havasi. 2017. Conceptnet 5.5: An open multilingual graph of general knowledge. In *AAAI conference on artificial intelligence (AAAI)*.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. Commonsenseqa: A question answering challenge targeting commonsense knowledge. *North American Chapter of the Association for Computational Linguistics (NAACL)*.

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R 3: Reinforced ranker-reader for open-domain question answering. In *AAAI Conference on Artificial Intelligence (AAAI)*.

Jiacheng Xu and Greg Durrett. 2021. Dissecting generation modes for abstractive summarization models via ablation and attribution. *Association for Computational Linguistics (ACL)*.

Yichong Xu, Chenguang Zhu, Ruochen Xu, Yang Liu, Michael Zeng, and Xuedong Huang. 2021. Fusing context into knowledge graph for commonsense question answering. In *Association for Computational Linguistics (ACL)*.

Donghan Yu, Chenguang Zhu, Yuwei Fang, Wenhao Yu, Shuohang Wang, Yichong Xu, Xiang Ren, Yiming Yang, and Michael Zeng. 2021. Kg-fid: Infusing knowledge graph in fusion-in-decoder for open-domain question answering. *arXiv preprint arXiv:2110.04330*.

Jingqing Zhang, Yao Zhao, Mohammad Saleh, and Peter Liu. 2020. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization. In *International Conference on Machine Learning (ICML)*.

Yizhe Zhang, Siqi Sun, Xiang Gao, Yuwei Fang, Chris Brockett, Michel Galley, Jianfeng Gao, and Bill Dolan. 2021. Joint retrieval and generation training for grounded text generation. *arXiv preprint arXiv:2105.06597*.

Wangchunshu Zhou, Dong-Ho Lee, Ravi Kiran Selvam, Seyeon Lee, and Xiang Ren. 2021. Pre-training text-to-text transformers for concept-centric common sense. In *International Conference on Learning Representations (ICLR)*.