An Efficient Memory-Augmented Transformer for Knowledge-Intensive NLP Tasks

Yuxiang Wu † **Yu Zhao** ‡ **Baotian Hu** ‡* **Pasquale Minervini** §† **Pontus Stenetorp** † **Sebastian Riedel** †

† University College London, London, UK ‡ Harbin Institute of Technology, Shenzhen, PRC

§ University of Edinburgh, Edinburgh, UK

{yuxiang.wu, p.stenetorp, s.riedel}@cs.ucl.ac.uk p.minervini@ed.ac.uk

20s151163@stu.hit.edu.cn hubaotian@hit.edu.cn

Abstract

Access to external knowledge is essential for many natural language processing tasks, such as question answering and dialogue. Existing methods often rely on a *parametric model* that stores knowledge in its parameters, or use a *retrieval-augmented model* that has access to an external knowledge source. Parametric and retrieval-augmented models have complementary strengths in terms of computational efficiency and predictive accuracy. To combine the strength of both approaches, we propose the Efficient Memory-Augmented Transformer (EMAT) – it encodes external knowledge into a key-value memory and exploits the fast maximum inner product search for memory querying. We also introduce pre-training tasks that allow EMAT to encode informative key-value representations, and to learn an implicit strategy to integrate multiple memory slots into the transformer. Experiments on various knowledge-intensive tasks such as question answering and dialogue datasets show that, simply augmenting parametric models (T5-base) using our method produces more accurate results (e.g., $25.8 \rightarrow 44.3$ EM on NQ) while retaining a high throughput (e.g., 1000 queries/s on NQ). Compared to retrieval-augmented models, EMAT runs substantially faster across the board and produces more accurate results on WoW and ELI5.[1]

1 Introduction

NLP tasks often require knowledge that is not explicitly provided with the input. For example, Open-Domain Question Answering (ODQA) requires answering an open-domain question without given context passages (Chen et al., 2017), and likewise for open-domain dialogue (Dinan et al., 2019). To handle such tasks, one key challenge is storing and accessing potentially large amounts of knowledge. One approach is a parametric method that trains a sequence-to-sequence generator to represent knowledge within model parameters. Petroni et al. (2019) find that pre-trained Language Models (PLMs) learn a partial knowledge base in their parameters, but its coverage is limited. Increasing model size can improve this issue (Raffel et al., 2020; Roberts et al., 2020; Brown et al., 2020); however, larger language models require significant computational resources.

Retrieval-augmented models (Guu et al., 2020; Lewis et al., 2020b; Izacard and Grave, 2021; Das et al., 2022), on the other hand, retrieve relevant passages from an external knowledge source (e.g., Wikipedia), and use the retrieved passages to inform generation. Despite being more accurate, retrieval-augmented models are often significantly more costly computation-wise than their parametric counterparts, since they require retrieving, encoding, and integrating the external knowledge at inference time.

To combine the strengths of both parametric and retrieval-augmented models, we propose Efficient Memory-Augmented Transformers (EMATs) – an extension to Transformer-based models augmented with an efficient key-value memory module. EMAT first encodes the external knowledge source into key embeddings and value embeddings, to construct the key-value memory (Section 3.1). We choose PAQ (Lewis et al., 2021b), a large collection of question-answering generated from Wikipedia, as our knowledge source; and we encode the questions as keys and answers as values. The transformer model produces dense query vector, retrieves from the key-value memory (Section 3.2), and integrates the returned dense key-value vectors at different encoder layers to enhance generation (Section 3.3). Different from previous approaches (Lample et al., 2019; Fan et al., 2021; Chen et al., 2022), our query representation is com-

*Corresponding author.

[1]Our code and datasets are available at `https://github.com/uclnlp/EMAT`.

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 5184–5196
December 7-11, 2022 ©2022 Association for Computational Linguistics

puted at an early transformer layer, whereas retrieved key and value embeddings are incorporated into the model at a later layer. This design only requires one forward pass through the transformer model, and allows memory retrieval to run concurrently with the transformer forward pass, and hence reduces the computational overhead (see Fig. 1 for our architecture).

With this architecture, it is important that the key-value memory accurately represent the knowledge source, and the transformer learns a strategy to incorporate the retrieved key-value representations into the model. Therefore, we introduce pre-training tasks (Section 4.1), which include auto-encoding objectives to represent the questions and answers, and a question answering task to learn an implicit strategy to incorporate multiple key-value memory slots. Our ablation study (Section 6.1) shows that our pre-training tasks are crucial for the performance, and removing any of them could lead to more than 10pp drop in EM score on ODQA datasets.

Our contribution can be summarised as follows: *i*) we introduce EMAT, an efficient memory access module to augment the transformer architecture; *ii*) we exploit PAQ as our knowledge source, and propose pre-training objectives to encode QA-pairs as key-value memory and to learn integration strategy to incorporate multiple memory slots into transformers; *iii*) experimental results on various knowledge-intensive tasks show that our proposed method significantly outperforms vanilla transformer baselines, while retaining a similar inference speed.

2 Related Work

Retrieve-and-Read Models for ODQA Open-domain question answering is a task that aims to answer a open-domain question without given context passages. Many ODQA systems follow a two-steps *retrieve-and-read* architecture (Chen et al., 2017) where, in the first step, a *retriever* model collects a set of relevant passages, and then a *reader* model processes the retrieved passages and produces the answer (Min et al., 2019; Yang et al., 2019; Wang et al., 2019; Karpukhin et al., 2020; Guu et al., 2020; Lewis et al., 2020b; Izacard and Grave, 2021). Despite their high accuracy, retrieve-and-read systems have a high computational footprint, since they need to process a potentially large number of passages (Wu et al., 2020, 2021).

Efficient OQDA Systems One simple approach to accelerate ODQA is *Closed-Book QA* (CBQA) – a sequence-to-sequence model (Sutskever et al., 2014; Kalchbrenner et al., 2014) such as T5 (Raffel et al., 2020) or BART (Lewis et al., 2020a) is fine-tuned on ODQA data, by training it to produce the answer given the question. CBQA models are substantially faster than retrieve-and-read approaches. However, since they solely rely on their parameters to store factual knowledge, their capacity is limited by the model size, and hence they often produce less accurate results than retrieve-and-read methods (Lewis et al., 2021a; Liu et al., 2021). Another efficient approach is retrieving semantically similar questions from a large collection of QA pair and returning the corresponding answers. Lewis et al. (2021b) propose PAQ, a 65 million QA dataset that is constructed with the objective to cover the most probably-asked questions in Wikipedia. RePAQ (Lewis et al., 2021b), a retrieval-based QA system built on PAQ, won the EfficientQA competition (Min et al., 2020) in 2020, outperforming CBQA models by a large margin. In this work, we choose PAQ as our knowledge source, but different from RePAQ, we develop a generative model. Our results show that EMAT outperforms RePAQ while matching its efficiency.

Memory-Augmented Transformers Geva et al. (2021) show that the Feed-Forward Network (FFN) layers in Transformer-based language models behave similarly to like key-value memories, where keys capture input patterns, and values map to the output vocabulary. Based on this finding, Yao et al. (2022) propose to extend the FFN layers by concatenating a dense representation of the corpus to the layer weights. Fan et al. (2021) introduce a neural module to access a fixed external memory, showing that it can lead to significant improvements on downstream generative dialogue modelling tasks. Concurrently to our work, Chen et al. (2022) propose QAMAT, a method to augment Transformer layers with a key-value memory network encoding question-answer pairs. QAMAT requires two inference steps through the encoder: one to retrieve memory values, and another for concatenating the retrieved values to the input. In contrast, our proposed method only requires a single inference steps, resulting in a significantly smaller computational footprint. Empirically, we show that our method is ≈ 5 times faster than QAMAT, even when using fewer hardware resources.

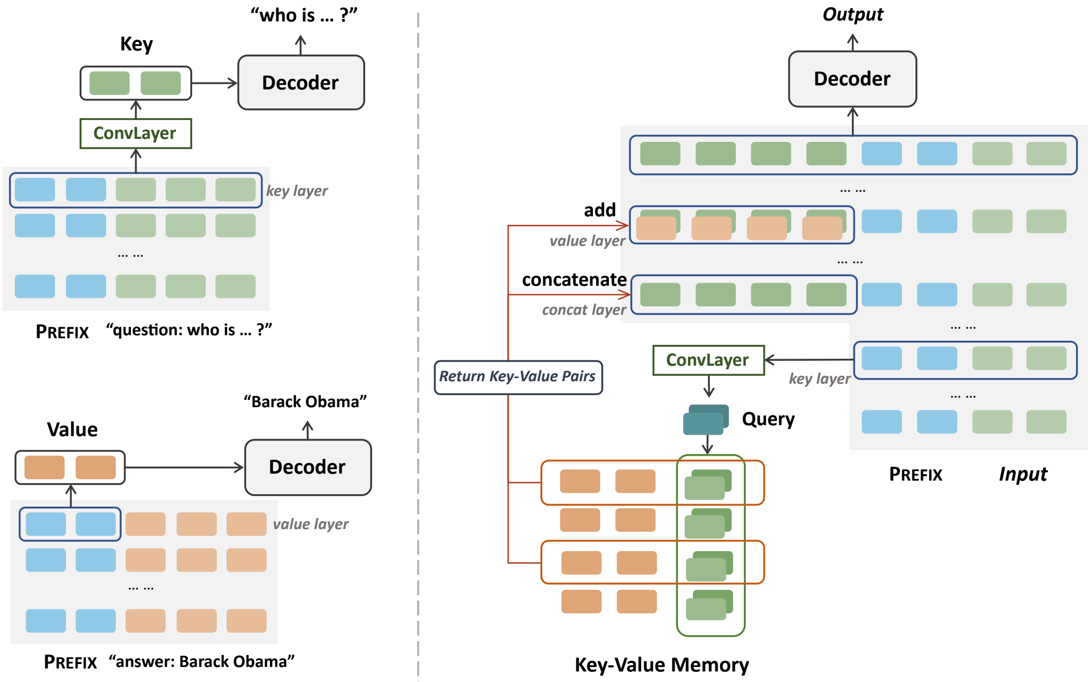


Figure 1: Architecture of the proposed Efficient Key-Value Memory Augmented Transformers (EMAT): factual knowledge is stored in a key-value memory (Section 3.1) where keys and values correspond to questions and answers, respectively; during inference, the model retrieves information from the memory via MIPS (Section 3.2) and uses it to condition the generation process.

3 Efficient Memory-Augmented Transformer

In this work we propose Efficient Memory-Augmented Transformer (EMAT), a model architecture that uses a key-value memory to store millions of dense question-answer representations to inform its predictions (see Fig. 1). Given an input sequence $X = (x_1, \cdots, x_{|X|})$, EMAT's encoder first produces a dense query $\mathbf{q}$ to retrieve from the memory $\mathbf{M}$. The returned key-value representations corresponding to the retrieved k key-value pairs are $Z = (z_1, \cdots, z_k)$. Finally, the decoder generates the target sequence $Y = (y_1, \cdots, y_{|Y|})$ conditioned on the input X and retrieved key-value pairs Z.

3.1 Key-Value Memory

The key-value memory $\mathbf{M} = (\mathbf{K}, \mathbf{V})$ contains representations of keys $\mathbf{K}$ and values $\mathbf{V}$, with each key $\mathbf{k}_i$ mapping to one value $\mathbf{v}_i$. Since we use PAQ (Lewis et al., 2021b) as our knowledge source, each key represents a question, and its value represents the corresponding answer. We use EMAT's encoder to encode the question and the answer separately, and it produces key and value embeddings from l_k-th and l_v-th layer of encoder respectively.

Fig. 1 (left) shows details regarding how key (question) and value (answer) are encoded in EMAT. To encode the key embeddings, we first concatenate a prefix PREFIX of length P with the question q as input, and then obtain the hidden states at the l_k-th layer $\mathbf{h}^{l_k} = [\mathbf{h}_1^{l_k}, \cdots, \mathbf{h}_n^{l_k}]$, where n is the length of the question q prepended with PREFIX. Then, $\mathbf{h}^{l_k}$ is passed through a convolutional neural network layer to produce $[\mathbf{c}_1, \cdots, \mathbf{c}_n]$, and we use the prefix part as our final key representation $\mathbf{k} = [\mathbf{c}_1, \cdots, \mathbf{c}_\mathrm{P}] \in \mathbb{R}^{\mathrm{P} \times h}$. For value embeddings, we prepend a prefix to the answer, feed $[\text{PREFIX}; a]$ into the model, and use the prefix's representation at the l_v-th layer of encoder $\mathbf{v} = [\mathbf{h}_1^{l_v}, \cdots, \mathbf{h}_\mathrm{P}^{l_v}] \in \mathbb{R}^{\mathrm{P} \times h}$ as our value representation, where h is the size of hidden representations.

3.2 Memory Retrieval

The goal of the retriever is to retrieve relevant entries from the key-value memory $\mathbf{M}$ to inform the downstream generation tasks. EMAT's encoder embeds the question into a query $\mathbf{q}$ using the same procedure as the key embeddings, described in Section 3.1. We conduct an extra step of flattening for both $\mathbf{q}$ and $\mathbf{k}$ by averaging: $\bar{\mathbf{k}} = \text{flatten}(\mathbf{k}) =$

$\frac{1}{\mathrm{P}}\sum_{j=1}^{\mathrm{P}}\mathbf{k}_j$. The key-value encoder shares the parameters with the question encoder, and we define the query-key similarity by the inner product between the flattened query representation and key representation $\mathrm{sim}(\mathbf{q},\mathbf{k}) = \langle\bar{\mathbf{q}},\bar{\mathbf{k}}\rangle$. At inference time, this operation can be efficiently computed using Maximum Inner Product Search (MIPS) to retrieve the top-k key-value pairs $Z = \{(\mathbf{k}_i,\mathbf{v}_i)\}_{i=1}^{k}$ based on the similarity. MIPS implementations such as `faiss` (Johnson et al., 2019) enable searching across millions of vectors in milliseconds on a CPU. The retrieved key-value pairs Z are then integrated in later layers of EMAT's encoder.

3.3 Key-Value Integration

Once we have retrieves the top-k key-value pairs Z, they need to be incorporated into the model. More specifically, in the l_c-th layer, all the key embeddings in Z are ordered by their corresponding similarity scores, and concatenated into a matrix $\mathbf{K}' = [\mathbf{k}_i,\cdots,\mathbf{k}_k] \in \mathbb{R}^{\mathrm{P}k\times h}$. Then it is prepended to the l_c-th layer's hidden states. To distinguish the different keys, we additionally add relative positional encodings to $\mathbf{K}'$. In the l_v-th layer, the value embedding in $\mathbf{Z}$ are concatenated in the same way to produce $\mathbf{V}'$, and it is *added* to the positions where their corresponding key embeddings are prepended to. The updated hidden states continue the forward pass of the remaining transformer encoder layers. Finally, the decoder generates the answer condition on the output of the encoder, which already integrates the retrieved key-value representations.

4 Training Pipeline of EMAT

4.1 Pre-Training

Auto-encoding Tasks We use T5-base's pre-trained parameters to initialise EMAT, but the prefix embeddings and key encoder's convolutional layer are trained from scratch. To obtain better representation of key and value, we pre-train EMAT with auto-encoding training objectives. We use PAQ-L1, a simplified version of PAQ that consists of 14M QA pairs, as the pre-training corpus. The model is trained to recover the input question x given the key embeddings $\mathbf{k}$, and the answer y given the value embeddings $\mathbf{v}$, as shown in Fig. 1 (left). The two tasks key auto-encoding (KAE) and value auto-encoding (VAE) can be formalised as:

$$\mathcal{L}_{\text{KAE}} = -\sum_{i=1}^{|X|}\log P(x_i \mid \mathbf{k}, x_{<i}),$$

$$\mathcal{L}_{\text{VAE}} = -\sum_{i=1}^{|Y|}\log P(y_i \mid \mathbf{v}, y_{<i}).$$

Generation Task Besides the problem of representing questions and answers in key-value memory $\mathbf{M}$, we also need the model to make use of $\mathbf{M}$ for downstream tasks. Thus, it is also critical to pre-train the model to learn the key-value integration module defined in Section 3.3. Since PAQ provides a large number of QA pairs, we consider a generation task built on PAQ to pre-train the model. More concretely, for each QA pair (x, y) in PAQ, we use the RePAQ model (Lewis et al., 2021b) to retrieve 10 other relevant QA pairs from PAQ, and retrieve their corresponding keys $\mathbf{K}'_x = [\mathbf{k}_1,\cdots,\mathbf{k}_{10}]$ and values $\mathbf{V}'_x = [\mathbf{v}_1,\cdots,\mathbf{v}_{10}]$ from the memory $\mathbf{M}$. Then, the model is trained to generate the answer y given the question x and the key-value embeddings corresponding to the retrieved QA pairs. The objective can be defined as follows:

$$\mathcal{L}_{\text{Gen}} = -\sum_{i=1}^{|Y|}\log P(y_i \mid x, \mathbf{K}'_x, \mathbf{V}'_x, y_{<i}).$$

We adopt a multi-task pre-training objective to minimise $\mathcal{L}_{\text{KAE}} + \mathcal{L}_{\text{VAE}} + \mathcal{L}_{\text{Gen}}$.

4.2 Fine-Tuning on Downstream Tasks

After pre-training, we fine-tune both the memory retrieval module and the generation of EMAT on the downstream tasks.

Retrieval Objective Learning to retrieve relevant key-value pairs that provide useful evidence to solve a given task can be challenging due to the lack of labelled data. To solve this problem, we propose a weakly-supervised method to optimise the retriever. Specifically, we first rank all retrieved key-value pairs retrieved from the memory $\mathbf{M}$ by their inner product scores. We consider the top retrieved key-value pairs: for each retrieved key-value pair, if its corresponding answer is lexically matched with the target output, then the key-value pair is selected as positive sample to optimise the retriever. For short output generation tasks such as ODQA, we match the answer corresponding to the retrieved value with the target answer. For long sequence generation tasks such as open-domain dialogue and long-form QA, we normalise the target sequence (i.e., lower-casing and removing stop words), and check whether if the retrieved value (answer) is contained in the normalised target sequence. Since these key-value pairs are more likely to lead to the correct answer, they can be used to provide a weakly-supervised training signal to the retrieval component of EMAT.

We denote the selected positive key-value pairs as $Z^+ = (z_1^+, \cdots, z_r^+)$, where each pair $z_i^+ = (\mathbf{k}_i^+, \mathbf{v}_i^+)$ is composed by a key component k_i^+ and a value component v_i^+. We sample a key-value pair z_i^+ from Z^+ based on the similarity between the corresponding key $\mathbf{k}_i^+$ and the query $\mathbf{q}$:

$$P_\eta(z_i^+ \mid q) = \frac{\exp(\mathrm{sim}(\mathbf{q}, \mathbf{k}_i^+))}{\sum_{j=1}^{r} \exp(\mathrm{sim}(\mathbf{q}, \mathbf{k}_j^+))},$$
$$z^+ \sim P_\eta(\cdot \mid q, Z^+).$$

We then select m negative pairs $\{z_j^-\}_{j=1}^m$ that do not match the target sequence. Finally, the positive pairs z^+ and the negative pairs z^- are used to train the retriever, by optimising the following objective:

$$\mathcal{L}_{\mathrm{Ret}} = - \log \frac{\exp(\mathrm{sim}(\mathbf{q}, \mathbf{k}_i^+))}{\exp(\mathrm{sim}(\mathbf{q}, \mathbf{k}_i^+)) + \sum_{j=1}^{m} \exp(\mathrm{sim}(\mathbf{q}, \mathbf{k}_j^-))}.$$

Memory Caching for More Efficient Training As described above, EMAT uses MIPS for retrieving the key-value pairs that are the most relevant to solve the current task. However, updating the memory $\mathbf{M}$ after each training update may not be feasible when the number of entries in $\mathbf{M}$ is very large. To alleviate this problem, we design a *memory caching* mechanism. At the beginning of each training epoch, we freeze the memory $\mathbf{M}$ and, for each training example, we retrieve the top-n key-value pairs. The memory $\mathbf{M}$ is updated only at the end of the epoch by re-encoding all entries in the knowledge source.

Overall Fine-Tuning Objective The generator is optimised to generate the target y given the input x and the top-n retrieved key-value pairs Z:

$$\mathcal{L}_{\mathrm{Gen}} = -\sum_{i=1}^{|Y|} \log P(y_i \mid x, Z, y_{<i}),$$

so the overall fine-tuning objective is $\mathcal{L}_{\mathrm{Ret}} + \mathcal{L}_{\mathrm{Gen}}$.

4.3 Inference

During inference, we use a fast Hierarchical Navigable Small World (HNSW, Malkov and Yashunin, 2020) graph index, generated by `faiss`, to search and retrieve from the key-value memory $\mathbf{M}$. If the $l_k < l_c$, the search process can run in parallel with the evaluation of the layers $l_k + 1, \cdots, l_c - 1$ in EMAT. Since the search process can be efficiently executed on CPU, it does not increase the GPU memory requirements of the model.

5 Experiments

5.1 Experimental Setup

Datasets We consider several knowledge-intensive NLP tasks, including Open-Domain Question Answering (ODQA), Open-Domain Dialogue (ODD), and Long-Form Question Answering (LFQA). In ODQA, the aim is to answer factual questions using a large collection of documents of diversified topics. We choose three commonly used datasets – NaturalQuestions (NQ, Kwiatkowski et al., 2019), TriviaQA (TQA, Joshi et al., 2017), and WebQuestions (Berant et al., 2013). In addition, we consider two generation tasks from the Knowledge Intensive Language Tasks (KILT, Petroni et al., 2021) benchmark to test whether our method generalises to tasks beyond ODQA. Specifically, we consider Wizard-of-Wikipedia (WoW, Dinan et al., 2019) for ODD. This task requires modelling long dialogue history and acquire relevant Wikipedia knowledge to produce a response utterance. Furthermore, we consider the Explain Like I'm Five (ELI5, Fan et al., 2019) dataset for LFQA. In ELI5, answers are often more diverse and open-ended compared to ODQA, and they tend to be significantly longer – they can be composed by several sentences.

Knowledge Source We use PAQ (Lewis et al., 2021b) as our knowledge source, and encode question-answer pairs in the model's key-value memory. Since the generative model used to generate the QA pairs in PAQ was trained on NaturalQuestions and TriviaQA, PAQ has a high coverage for these two ODQA datasets. In this work, we also evaluate on tasks beyond ODQA, where it is not clear how PAQ can be used. Therefore, our evaluation on ODD and LFQA aims to demonstrate that EMAT generalises to different knowledge-intensive generation tasks using PAQ as the underlying knowledge source.

Baselines We compare our method with three types of baselines: *parametric models*, *retrieval-only approaches*, and *retrieval-augmented models*. Parametric models fine-tune sequence-to-sequence PLMs such as T5 (Raffel et al., 2020) or BART (Lewis et al., 2020a) on a datasets, by casting each task as a sequence generation problem conditioned on the input. In our experiments, we consider parametric models of multiple sizes, including T5-base, T5-large, T5-3B, T5-11B (Roberts et al., 2020), and BART-large (Lewis et al., 2020a). Retrieval-only approaches retrieve the most relevant information from the knowledge source (PAQ), and return the top answer as output. In ODQA benchmark we use the RePAQ model proposed by Lewis et al. (2021b); in ODD and LFQA, we use the EMAT key retrieval module described in Section 3.2 as the retriever. Retrieval-augmented models such as RAG (Lewis et al., 2020b) or FiD (Izacard and Grave, 2021) retrieve relevant passages from Wikipedia using a dense retriever such as DPR (Karpukhin et al., 2020), and then use the retrieved passages and the input sequence to condition the generation process.

Pre-Training and Fine-Tuning Configurations We base our EMAT on T5 (Raffel et al., 2020), and initialise our model with the pre-trained parameters from T5-base.[2] To evaluate the speed and accuracy of our proposed method under different computation environments, we pre-train and fine-tune EMAT using two settings. In the former setting, we set $l_k = 3$, $l_c = 3$, $l_v = 7$, which emulates an environment where key embeddings has fast access, but there is delay in acquiring value embeddings; we refer to this setting as *Fast Key, Slow Value* (FKSV). In the latter setting, $l_k = 3$, $l_c = 10$, $l_v = 11$, which corresponds to a scenario where both key querying and value reading can have significant delays. We refer to this setting as *Slow Key, Slow Value* (SKSV). All details on the training hyperparameters the hardware used in our experiments are available in Appendix B.

5.2 Open-Domain Question Answering

Table 1 shows the experimental results on three ODQA datasets: NQ (Kwiatkowski et al., 2019), TQA (Joshi et al., 2017), and WQ (Berant et al., 2013). We report the Exact Match (EM) scores and the average inference speed measured by queries per second (Q/s). Compared with *parametric models*, our proposed method yields substantially higher EM scores across three datasets. EMAT-FKSV outperforms T5-base, which share the same backbone model, by 18.5, 20.0, 10.1 percentage points on NQ, TQA and WQ respectively. These results indicate that our method of augmenting transformer with key-value memory effectively extends model's knowledge capacity. Compared with *retrieval-only models*, our method also demonstrates strong performance. RePAQ baselines also exploit PAQ as knowledge source, and hence is comparable with our method. Our EMAT-FKSV outperforms the best RePAQ retriever (RePAQ-large) by 2.8 and 3.1 percentage points on NQ and TQA respectively. Speed-wise, EMAT can answer 1000-1200 Q/s, which is a high throughput in ODQA and is comparable to some of the fastest parametric models and retrieval-only models.

In ODQA, *retrieval-augmented models* are known to be highly accurate, but are also computationally inefficient (Min et al., 2020). EMAT is significantly faster than these models. For example, FiD-base uses the same backbone T5-base model as EMAT, but retrieves and concatenates 20 to 100 passages from Wikipedia. Despite being less accurate on NQ and TQA, EMAT is two or-

Model	NQ		TQA	WQ
	EM	Q/s	EM	EM
Parametric models				
T5-base (Roberts et al., 2020)	25.8	1600	24.4	26.6
T5-large (Roberts et al., 2020)	27.6	570	29.5	27.7
T5-3B (Roberts et al., 2020)	30.4	55	35.1	33.6
T5-11B (Roberts et al., 2020)	32.6	-	42.3	37.2
BART-large (Lewis et al., 2020a)	26.5	570	26.7	27.4
Retrieval-only models				
Dense Retriever (Lewis et al., 2021a)	26.7	-	28.9	-
DensePhrases (Lee et al., 2021)	40.9	18	50.7	-
RePAQ-base (Lewis et al., 2021b)	40.9	1400	39.7	29.4
RePAQ-large (Lewis et al., 2021b)	41.2	1100	-	-
RePAQ-xlarge (Lewis et al., 2021b)	41.5	800	41.3	-
Retrieval-augmented models				
REALM (Guu et al., 2020)	40.4	-	55.8	40.7
DPR (Karpukhin et al., 2020)	41.5	2.7	57.9	42.4
QAMAT (Chen et al., 2022)	44.7	240*	48.0	39.4
RePAQ rerank (Lewis et al., 2021b)	45.7	55	48.9	37.6
RAG (Lewis et al., 2020b)	44.5	9.6	56.8	45.2
FiD-base (Izacard and Grave, 2021)	48.2	3.7	65.0	32.4
FiD-large (Izacard and Grave, 2021)	51.4	1.4	67.6	-
Ours				
EMAT-FKSV	44.3	1000	44.4	36.7
EMAT-SKSV	43.3	1200	43.7	33.2

Table 1: Exact Match (EM) results for EMAT in comparison to recent state-of-the-art systems. $*$ QAMAT runs on 32 TPU-v3 with 1024GB TPU memory, whereas ours run on A100 GPU with 40GB GPU memory.

[2]We use the original version of T5 without SSM to ensure that our results are comparable with the baselines.

Model	F1	R-L	U/s
Parametric models			
Trans MemNet (Dinan et al., 2019)	11.85	10.11	-
BART-large (Lewis et al., 2020a)	12.86	11.77	55
T5-base (Raffel et al., 2020)	13.53	12.40	160
Retrieval-augmented models			
BART + DPR (Petroni et al., 2021)	15.19	13.23	0.7
RAG (Lewis et al., 2020b)	13.11	11.57	3.4
Retrieval-only models			
RePAQ w/ EMAT key encoder	1.84	1.48	-
Ours			
EMAT-FKSV	**15.78**	**14.73**	141
EMAT-SKSV	15.35	14.68	**150**

Table 2: Results on the Wizard-of-Wikipedia dataset from the KILT benchmark.

Model	F1	R-L	Q/s
Parametric models			
BART-large (Lewis et al., 2020a)	19.23	20.55	30
T5-base (Raffel et al., 2020)	16.01	19.08	76
Retrieval-augmented models			
BART + DPR (Petroni et al., 2021)	17.88	17.41	0.2
RAG (Lewis et al., 2020b)	14.51	14.05	0.4
Retrieval-only models			
RePAQ w/ EMAT key encoder	1.40	1.65	-
Ours			
EMAT-FKSV	18.42	20.61	67
EMAT-SKSV	**19.03**	**20.91**	**71**

Table 3: Results on the ELI5 dataset from the KILT benchmark.

ders of magnitude faster than FiD-base and more accurate on WQ. On NaturalQuestions in terms of EM score, our method outperforms REALM and DPR, and is comparable with QAMAT and RAG. QAMAT (Chen et al., 2022) is a concurrent work to ours and is the fastest among the retrieval-augmented models. But QAMAT runs on 32 TPU-v3 (Jouppi et al., 2017), which have roughly 1024GB TPU memory, and the MIPS search is conducted on TPU. In contrast, EMAT runs on a single A100 GPU with 40GB GPU memory, and the MIPS search is executed on CPU. Despite using substantially fewer resources, EMAT-FKSV is roughly 4.2 times faster than QAMAT, and EMAT-SKSV is 5 times faster.

5.3 Generalisation to Open-Domain Dialogue and Long-Form QA

Open-Domain Dialogue Open-domain dialogue is a dialogue task that requires accessing knowledge from Wikipedia to produce dialogue response. Table 2 shows the results on the open-domain dialogue dataset Wizard-of-Wikipedia (Dinan et al., 2019) from the KILT (Petroni et al., 2021) benchmark. The utterances from dialogue history are concatenated into a input sequence, and the output sequence is the corresponding response utterance. We follow Petroni et al. (2021) and evaluate the models with F1 and ROUGE-L metrics, and we also report the average number of utterances generated per second (U/s) for speed evaluation.

The results show that, our proposed EMAT outperforms parametric models while retaining a similar inference speed. EMAT-FKSV outperforms T5-base by 2.25 F1 and 2.28 ROUGE-L points, while generating 141 utterances per second. Surprisingly, EMAT models also outperform retrieval-augmented models such as RAG and BART+DPR, which exploits Wikipedia as knowledge source. It indicates that our method that encodes PAQ as key-value memory is capable of represent crucial information in Wikipedia, and generalises well to tasks beyond ODQA. We also implement a RePAQ-equivalent retrieval-only model using EMAT's key encoder. Since PAQ is a collection of QA pairs, simply retrieving relevant QA pairs for dialogue does not work well. The large gap between EMAT and RePAQ with EMAT key encoder, together with the qualitative analysis in Section 6.2, demonstrates that EMAT decoder does not simply copy information from the key-value memory, but exploits the key-value embeddings to generate novel response.

Long-Form Question Answering Results on the LFQA task ELI5 (Fan et al., 2019) (shown in Table 3) reveals similar conclusions as in WoW. Both EMAT-FKSV and EMAT-SKSV outperform the T5-base baseline by a large margin, 2.41pp and 3.01pp in F1, while retaining an inference speed to 67 Q/s and 71 Q/s, respectively. EMAT is both faster and more accurate than retrieval-augmented models on ELI5 too. Compared to RAG, EMAT-SKSV is 4.52pp better in F1, 6.86pp better in ROUGE-L, and more than 160 times faster in inference speed.

6 Analysis

6.1 Ablation Study

We conduct ablation study on the pre-training steps and the results are shown in Table 4. Without fine-tuning, the pre-trained EMAT outperforms fine-

Model	NQ	TQA	WQ
EMAT-FKSV	44.3	44.4	36.7
– fine-tune	30.6	32.4	25.6
– auto-encoding tasks	28.5	34.6	12.9
– generation task	28.7	24.7	31.4
– all pre-training tasks	27.1	17.7	6.0

Table 4: Ablation on the pre-training steps used by EMAT, described in Section 4.1, measured using EM on NQ, TQA, and WQ: we analyse the impact of removing auto-encoding, generation, and all pre-training tasks from EMAT's pre-training phase.

tuned T5-large on NQ and TQA, and has a competitive result on WQ. When we remove the auto-encoding (KAE and VAE) tasks, the performance on NQ and WQ drops significantly (36.7 → 12.9 on WQ). Ablating the generation task results in substantially worse EM on NQ and TQA (44.4 → 24.7 on TQA) The ablation results demonstrate that both auto-encoding task and generation task are crucial to EMAT's performance. Without all the pre-training tasks, EMAT perform very poorly, and even worse than T5-base baseline. This may be due to the fact that the key-value memory is not well learned and hence incorporating them will introduce noise to the model, thus leads to poor predictions.

6.2 Qualitative Analysis

Table 5 shows some examples from NQ and WoW. The presented QA pairs correspond to the top-5 retrieved dense key-value pairs. In NQ, we can see that EMAT retrieves useful key-value and generates correct answer from the first example. Different from retrieval-only models that only output the top-1 retrieved QA, EMAT conducted some sort of reranking, and the decoder manages to use the right key-value to generate the answer. In another example presented in Table 6, it demonstrates that EMAT's output is not always from retrieved values. It will ignore the irrelevant key-value pairs, also uses evidences from keys, which are impossible for retrieval-only models.

In the example from WoW, it requires using the fine-grained knowledge *19th century* to generate response. We can see that EMAT retrieves context-related key-value pairs, and mainly uses the two underlined evidences to generate response. In contrast, T5-base generates hallucinated response, producing the wrong time "18th century". This shows that, with memory augmentation, EMAT generates

Natural Question	
Q: who plays the judge in drop dead diva?	**A**: *Lex Medlin*
EMAT: *Lex Medlin*	
Retrieved Key-Values	
Q: who plays jane on drop dead diva?	**A**: Brooke Elliott
Q: who played fred in drop dead diva?	**A**: Beverly Hills
Q: who played empress katia on drop dead diva?	**A**: Tobin
Q: who plays judge french in drop dead divorce season 4?	**A**: <u>*Lex Medlin*</u>
Q: who played ian holt on drop dead diva?	**A**: Jeffrey Pierce
Wizard-of-Wikipedia	
Dialogue History	
Apprentice:	I like jazz.
Wizard:	That's great! Jazz ... is originated in theafrican-american communitie.
Apprentice:	When did it originate?
Response	
Target:	Jazz originated in the *late 19th century*
T5-base:	It was first recorded in the late 18th century
EMAT:	It originated in *late 19th century* in *new orleans*
Retrieved Key-Values	
Q: where did the genre of jazz originate?	**A**: <u>*New Orleans*</u>, US
Q: when did jazz music start in the united states?	**A**: <u>1920s</u>
Q: what type of music was jazz originally?	**A**: dance music
Q: what genre of music does rock come from?	**A**: blues
Q: what genre of music is hip hop?	**A**: rap

Table 5: Examples from NQ and WoW. Noting that EMAT only retrieves and integrates dense key-value pairs, but not accesses the presented text-based QAs.

a more faithful and informative response than T5-base.

More examples can be found in Table 6 in the appendix. We find that EMAT retrieves useful key-value pairs and makes full use of them to generate answers. This analysis also demonstrates the interpretability of EMAT, and the feasibility of only using dense key-value embeddings to provide knowledge.

7 Conclusions

In this work, we propose the Efficient Memory-Augmented Transformer (EMAT) that combines the strength of parametric model and retrieval-augmented model. It encodes external knowledge into a key-value memory and exploits the fast MIPS search for memory querying. We introduce pre-training tasks to learn better key-value representations and integration of multiple memory slots into transformer. Experiments on knowledge intensive tasks, including open-domain question answering, dialogue and long-form question answering, show both the accuracy and speediness of EMAT. In the future, we will seek to improve integrate more diverse knowledge into the memory and generalise our method to more downstream tasks.

Limitations

One limitation is that the memory retrieval module requires weak supervision to train with. This may mean that we define different weak supervision labels when apply to different downstream tasks. One could use end-to-end training techniques such as the ones proposed by Paranjape et al. (2021); Lewis et al. (2020b), to train the memory retrieval module with gradients from the decoder, and we leave this as future work. Another potential limitation is that, we need to store the dense key-value memory $\mathbf{M}$, which requires around 300GB CPU RAM. But since it is relatively easy to get machine with more CPU RAM than GPU memory, and the fact that most deep learning workstations can reach this requirement, we believe this is not too much a constraint. Besides, we can use LRU cache to save RAM in low memory resource situations.

Acknowledgements

Pasquale and Pontus were partially funded by the European Union's Horizon 2020 research and innovation programme under grant agreement no. 875160, and by an industry grant from Cisco. Baotian Hu and Yu Zhao were funded by grants: Natural Science Foundation of China (No.62006061), Stable Support Program for Higher Education Institutions of Shenzhen (No. GXWD20201230155427003-20200824155011001). The authors would also like to thank Patrick Lewis, Wenhu Chen, and Zetian Sun for their help and feedback.

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1533–1544, Seattle, Washington, USA. Association for Computational Linguistics.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Wenhu Chen, Pat Verga, Michiel de Jong, John Wieting, and William Cohen. 2022. Augmenting pre-trained language models with qa-memory for open-domain question answering. *CoRR*, abs/2204.04581.

Rajarshi Das, Patrick Lewis, Sewon Min, June Thai, and Manzil Zaheer, editors. 2022. *Proceedings of the 1st Workshop on Semiparametric Methods in NLP: Decoupling Logic from Knowledge*. Association for Computational Linguistics, Dublin, Ireland and Online.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2019. Wizard of wikipedia: Knowledge-powered conversational agents. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net.

Angela Fan, Claire Gardent, Chloé Braud, and Antoine Bordes. 2021. Augmenting transformers with KNN-based composite memory for dialog. *Transactions of the Association for Computational Linguistics*, 9:82–99.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Mor Geva, Roei Schuster, Jonathan Berant, and Omer Levy. 2021. Transformer feed-forward layers are key-value memories. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 5484–5495, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. REALM: retrieval-augmented language model pre-training. *CoRR*, abs/2002.08909.

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

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

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Norman P. Jouppi, Cliff Young, Nishant Patil, David A. Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, Rick Boyle, Pierre-luc Cantin, Clifford Chao, Chris Clark, Jeremy Coriell, Mike Daley, Matt Dau, Jeffrey Dean, Ben Gelb, Tara Vazir Ghaemmaghami, Rajendra Gottipati, William Gulland, Robert Hagmann, C. Richard Ho, Doug Hogberg, John Hu, Robert Hundt, Dan Hurt, Julian Ibarz, Aaron Jaffey, Alek Jaworski, Alexander Kaplan, Harshit Khaitan, Daniel Killebrew, Andy Koch, Naveen Kumar, Steve Lacy, James Laudon, James Law, Diemthu Le, Chris Leary, Zhuyuan Liu, Kyle Lucke, Alan Lundin, Gordon MacKean, Adriana Maggiore, Maire Mahony, Kieran Miller, Rahul Nagarajan, Ravi Narayanaswami, Ray Ni, Kathy Nix, Thomas Norrie, Mark Omernick, Narayana Penukonda, Andy Phelps, Jonathan Ross, Matt Ross, Amir Salek, Emad Samadiani, Chris Severn, Gregory Sizikov, Matthew Snelham, Jed Souter, Dan Steinberg, Andy Swing, Mercedes Tan, Gregory Thorson, Bo Tian, Horia Toma, Erick Tuttle, Vijay Vasudevan, Richard Walter, Walter Wang, Eric Wilcox, and Doe Hyun Yoon. 2017. In-datacenter performance analysis of a tensor processing unit. In *ISCA*, pages 1–12. ACM.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 655–665, Baltimore, Maryland. Association for Computational Linguistics.

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

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Guillaume Lample, Alexandre Sablayrolles, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. 2019. Large memory layers with product keys. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8546–8557.

Jinhyuk Lee, Mujeen Sung, Jaewoo Kang, and Danqi Chen. 2021. Learning dense representations of phrases at scale. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6634–6647, Online. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020a. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, Online. Association for Computational Linguistics.

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2021a. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1000–1008, Online. Association for Computational Linguistics.

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021b. PAQ: 65 million probably-asked questions and what you can do with them. *Transactions of the Association for Computational Linguistics*, 9:1098–1115.

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020b. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Linqing Liu, Patrick S. H. Lewis, Sebastian Riedel, and Pontus Stenetorp. 2021. Challenges in generalization in open domain question answering. *CoRR*, abs/2109.01156.

Yury A. Malkov and Dmitry A. Yashunin. 2020. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 42(4):824–836.

Sewon Min, Jordan L. Boyd-Graber, Chris Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin

Guu, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, Colin Raffel, Adam Roberts, Tom Kwiatkowski, Patrick S. H. Lewis, Yuxiang Wu, Heinrich Küttler, Linqing Liu, Pasquale Minervini, Pontus Stenetorp, Sebastian Riedel, Sohee Yang, Minjoon Seo, Gautier Izacard, Fabio Petroni, Lucas Hosseini, Nicola De Cao, Edouard Grave, Ikuya Yamada, Sonse Shimaoka, Masatoshi Suzuki, Shumpei Miyawaki, Shun Sato, Ryo Takahashi, Jun Suzuki, Martin Fajcik, Martin Docekal, Karel Ondrej, Pavel Smrz, Hao Cheng, Yelong Shen, Xiaodong Liu, Pengcheng He, Weizhu Chen, Jianfeng Gao, Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Sejr Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Wen-tau Yih. 2020. Neurips 2020 efficientqa competition: Systems, analyses and lessons learned. In *NeurIPS (Competition and Demos)*, volume 133 of *Proceedings of Machine Learning Research*, pages 86–111. PMLR.

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019. A discrete hard EM approach for weakly supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2851–2864, Hong Kong, China. Association for Computational Linguistics.

Ashwin Paranjape, Omar Khattab, Christopher Potts, Matei Zaharia, and Christopher D. Manning. 2021. Hindsight: Posterior-guided training of retrievers for improved open-ended generation. *ArXiv preprint*, abs/2110.07752.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

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

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 3104–3112.

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A globally normalized BERT model for open-domain question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5878–5882, Hong Kong, China. Association for Computational Linguistics.

Yuxiang Wu, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2021. Training adaptive computation for open-domain question answering with computational constraints. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 447–453, Online. Association for Computational Linguistics.

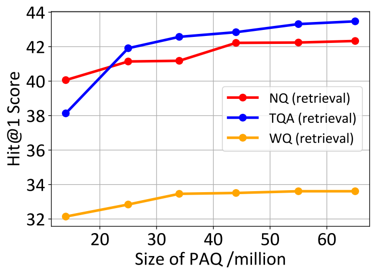
Yuxiang Wu, Sebastian Riedel, Pasquale Minervini, and Pontus Stenetorp. 2020. Don't read too much into it: Adaptive computation for open-domain question answering. In *EMNLP (1)*, pages 3029–3039. Association for Computational Linguistics.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota. Association for Computational Linguistics.

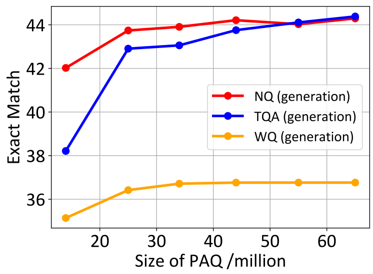
Yunzhi Yao, Shaohan Huang, Ningyu Zhang, Li Dong, Furu Wei, and Huajun Chen. 2022. Kformer: Knowledge injection in transformer feed-forward layers. *CoRR*, abs/2201.05742.

A Data Efficiency

Fig. 2 shows how the number of retrieved key-value pairs from PAQ-L1 influences the downstream EM score on Natural Questions, TriviaQA, and WebQuestions. The results show that, as the number of retrieved memory entries increases, EMAT's EM score monotonically increases, which indicates that the model can handle noise in the retrieved memory entries and benefit from larger number of retrieved memory entries. In Fig. 3 we analyse the scaling effects induced by using larger subsets of PAQ for creating the key-value memory $\mathbf{M}$. The results demonstrate that EMAT's predictive accuracy increases with the number of PAQ questions across all considered ODQA datasets.



a) EMAT-retriever Hit@1



b) EMAT-generator EM

Figure 3: Analysis of how the number of PAQ entries used to populate the memory $\mathbf{M}$ influences the downstream predictive accuracy on several ODQA datasets.

B Hyperparameters

Model Settings The length of PREFIX is 2 in EMAT. EMAT contains 225M parameters, and T5-base contains 221M parameters. The memory cache size is set to 384 in all downstream tasks. The retrieval loss weight and generation loss weight are both set to 1.

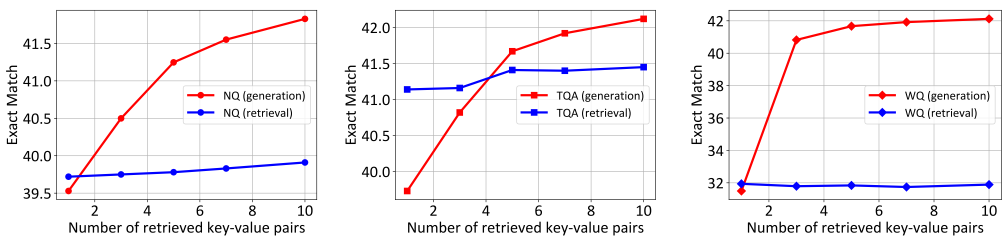
Pre-Training We pre-train for 5 epochs on PAQ-L1, using learning rate warm-ups for the first 5000 training steps to 10^{-4}, and linear rate decay in the remaining steps. For each QA in PAQ-L1, we use RePAQ to retrieve 10 relevant QAs from PAQ-L1. To force the model use relevant QAs' information, we sample 10% examples to retain itself in the relevant QA set. The weights of auto-encoding loss and generation loss is set to 0.5 and 1.0.

ODQA For NQ and TQA, the learning rate warm-ups for the first 1000 steps to 5×10^{-5}, and linear rate decay in the remaining steps. For WQ, the learning rate is fixed to 4×10^{-5} during training. We fine-tune 30 epochs on ODQA tasks, using early stop with patients of 8 epochs. We use greedy decoding algorithm to generate answers.

WoW We fine-tune 20 epochs on WoW with 8×10^{-5} learning rate. The scheduler is same to ODQA. We use greedy decoding algorithm to generate responses.

ELI5 We fine-tune 8 epochs on ELI5 with 5×10^{-5} learning rate. The scheduler is same to ODQA. We use beam-sample decoding algorithm to generate answers, where beam-size is 5, top-k is 64. We force the model do not generate repeat phrases by setting no_repeat_n_gram to 8.

Hardware The machine used to measure the speed is a machine learning workstation with Intel(R) Xeon(R) Platinum 8358 CPU, 512GB of CPU RAM and one 40GB NVIDIA A100 GPU.



a) Natural Questions b) TriviaQA c) WebQuestions

Figure 2: Analysis of how changing the number of retrieved key-value pairs influences the downstream Exact Match accuracy on several ODQA datasets.

<table>
<tr><th colspan="2">Natural Questions</th></tr>
<tr><td>Question
Answer
EMAT Predict:
Retrieved</td><td>who plays the judge in drop dead diva
[Lex Medlin]
Lex Medlin
question: who plays jane on drop dead diva? answer: Brooke Elliott
question: who plays judge french in drop dead divorce season 4? answer: <u>Lex Medlin</u>
question: who played fred in drop dead diva? answer: Beverly Hills, California</td></tr>
<tr><td>Question
Answer
EMAT Predict:
Retrieved</td><td>how long did the menendez brothers get in prison for killing their parents
[life imprisonment, life]
life
question: when did the menendez brothers kill their parents? answer: 1989
question: where did the menendez brothers kill their parents? answer: Beverly Hills, California
question: who sentenced the menendez brothers to <u>life in prison</u>? answer: Judge Weisberg</td></tr>
<tr><td>Question
Answer
Predict:
Retrieved</td><td>how long is a whale shark in meters
[12.65m, estimated at 9.7m, 9.7m]
few meters
question: how long does a whale shark live? answer: 70 to 100 years
question: how long does it take a whale shark to mature? answer: around 30 years
question: how long does it take a blue whale to dive? answer: 10 minutes</td></tr>
<tr><th colspan="2">Wizard-of-Wikipedia</th></tr>
<tr><td>Dialogue history

Ground Truth
T5 Predict
EMAT Predict
Retrieved</td><td>Wizard: Red the color at the end of the visible light spectrum looks good on everyone.
Apprentice: I am more of a fan of green. That would leave us only one primary color: Blue.
But the dominant wavelength of red is approximately 625–740. That’s impressive!
I agree. It is the color between green and red.
it is color between violet and green on spectrum of visible light
question: what is the next color in this series: green, white, red, green, ? answer: Blue
question: what is the color of <u>light between violet and green</u>? answer: Blue
question: what color looks more blue as it brightens? answer: Violet</td></tr>
<tr><td>Dialogue history

Ground Truth
T5 Predict
EMAT Predict
Retrieved</td><td>Apprentice: I like jazz.
Wizard: That’s great! Jazz is a music genre that originated in the african-american communities.
Apprentice: When did it originate?
Jazz originated in the late 19th century
It was first recorded in the late 18th century
It originated in late 19th century in new orleans
question: where did the genre of jazz originate? answer: <u>New Orleans</u>, United States
question: when did jazz music start in the united states? answer: <u>1920s</u>
question: what genre of music does rock come from? answer: blues</td></tr>
</table>

Table 6: More examples of EMAT’s prediction on NQ and WoW.