FiE: Building a Global Probability Space by Leveraging Early Fusion in Encoder for Open-Domain Question Answering

Akhil Kedia and **Mohd Abbas Zaidi** and **Haejun Lee**
Samsung Research, Seoul
{akhil.kedia, abbas.zaidi, haejun82.lee}@samsung.com

Abstract

Generative models have recently started to outperform extractive models in Open Domain Question Answering, largely by leveraging their decoder to attend over multiple encoded passages and combining their information. However, generative models tend to be larger than extractive models due to the need for a decoder, run slower during inference due to auto-regressive decoder beam search, and their generated output often suffers from hallucinations. We propose to extend transformer encoders with the ability to fuse information from multiple passages, using global representation to provide cross-sample attention over all tokens across samples. Furthermore, we propose an alternative answer span probability calculation to better aggregate answer scores in the global space of all samples. Using our proposed method, we outperform the current state-of-the-art method by 2.5 Exact Match score on the Natural Question dataset while using only 25% of parameters and 35% of the latency during inference, and 4.4 Exact Match on WebQuestions dataset. When coupled with synthetic data augmentation, we outperform larger models on the TriviaQA dataset as well. The latency and parameter savings of our method make it particularly attractive for open-domain question answering, as these models are often compute-intensive.

1 Introduction

Open-Domain Question-Answering is the task of answering an input question given a large external knowledge-base, such as the entire Wikipedia. This problem is typically approached by leveraging a retriever model to first retrieve a set of relevant documents/passages using some IR method, which are then passed on to a reader model (Lee et al., 2019; Karpukhin et al., 2020; Guu et al., 2020; Lewis et al., 2020; Xiong et al., 2021; Balachandran et al., 2021).

The reader model then encodes all the passages through a transformer encoder separately, as transformers have quadratic computation with input sequence length. Extractive span-based methods (Lee et al., 2019; Karpukhin et al., 2020; Guu et al., 2020; Lewis et al., 2020; Xiong et al., 2021) use these encoded representations for a per-passage selection of the answer span. Generative models such as FiD(Izacard and Grave, 2021b), and derivative models utilizing a similar reader, such as Izacard and Grave (2021a); Sachan et al. (2021); Singh et al. (2021); Lee et al. (2021) use their decoder to attend to all the passages simultaneously by concatenating their encoded representations. The encoder-decoder cross attention enables the generative readers to fuse information globally across different passages.

So far, the existing works have tried to perform global information fusion by adding a decoder and hence adopting the generative approach. We hypothesize that this is not efficient for extractive tasks where the answer exists in the provided context passages. It may be helpful to restrict the answer probability space to the given context instead of using the universal vocabulary, which might also lead to issues such as hallucinations (Xu et al., 2021; Longpre et al., 2021; Mielke et al., 2020; Zellers et al., 2019). Moreover, adding a decoder and performing auto-regressive answer generation increases the latency of the model. In this work, we propose to extend the transformer encoder of extractive models with the ability for early global fusion of information across input samples. We achieve this by adding global representation tokens to the input of the encoder, which can attend to all tokens in all passages with cross-sample attention. By doing so, not only do we remove the need for a decoder but also outperform existing generative approaches.

Extractive reader models typically (Lee et al., 2019; Karpukhin et al., 2020; Guu et al., 2020)

Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, pages 4246–4260
December 7-11, 2022

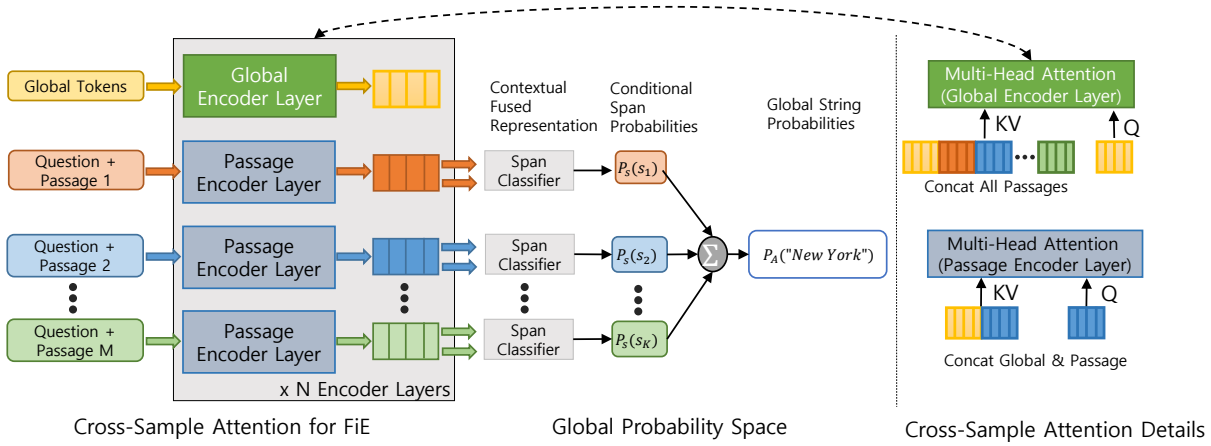

Figure 1: The overall architecture of our proposed model FiE: Fusion in Encoder. The global representation tokens attend to all the tokens from all the passages, while all passages attend to themselves and to the global representation.

marginalize the probability of a given span on a per-passage level. However, several works (Cheng et al., 2020, 2021) show that more information may be gleamed from all the occurrences of an answer span across multiple passages. The authors change probability space from a per-passage level to a global level, achieving large performance gains. We also adopt a similar approach to take the best advantage of the global information flow in our model. Moreover, extractive models classify each token as the start/end of the answer span. The probability of a span is then the probability of its start token multiplied by its end token. This inherently assumes that the start and end probabilities of a span are independent of each other. We modify the probability space by directly calculating the score for each span rather than multiplying the start and end token scores. This approach enables us to calculate a separate score for each span and enables better aggregation of answer probability scores.

We evaluate our proposed method on the most commonly used open-domain datasets - Natural Questions (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017) and WebQuestions (Berant et al., 2013). On Natural Questions, we outperform the previous best models having $4x$ more parameters by 2.5 Exact Match, achieving a new state-of-the-art score of 58.4EM. We also achieve an improvement in score over the previous best models by 4.4 Exact Match points on WebQuestions, and on TriviaQA outperform the best performing models when coupled with data augmentation. Our main contributions in this paper are -

- Fusing global information in the encoder by leveraging cross-sample attention, eliminating the need for a large and expensive decoder.
- Building a more robust global probability space for answer span probabilities.

2 Proposed Method

2.1 Background Nomenclature

Given input query tokens $\boldsymbol{q}$, n contexts $\boldsymbol{C_j}$ with tokens $C_{i,j}$, and a transformer encoder E with l layers E_ℓ, extractive methods pass them through the encoder to obtain encoded representations $\mathbf{h_{i,j}}$ for each token :

$$\mathbf{h_{i,j}} = \mathrm{E}(\boldsymbol{q}, \boldsymbol{C_j})[i]$$

Each token representation is then passed through a classifier $\mathrm{W}_{\text{start}}$ to obtain the logit scores $s_{\text{start}}(i,j)$, for each token being the start of the answer span. These are then softmaxed across i (on a per-passage level) to obtain probabilities, $p_{\text{start}}(i,j)$.

$$s_{\text{start}}(i,j) = \mathrm{W}_{\text{start}}(\mathbf{h_{i,j}}) \quad (1)$$

$$p_{\text{start}}(i,j) = \sigma_{\text{j}}(s_{\text{start}}(i,j)), \quad (2)$$

where σ_j is the softmax operation, across all tokens i in the context j. Similarly, we obtain probabilities $p_{\text{end}}(i,j)$ for the end token. In Cheng et al. (2020), the authors instead softmaxed across all tokens i across all contexts j, and showed that this global probability space helps.

The probability $p_{\text{span}}(st, en, j)$ for an answer span from $\boldsymbol{C_j}[st : en]$ are thereafter modeled as the product of the independent start and end token probabilities -

$$p_{\text{span}}(st, en, j) = p_{\text{start}}(st, j) * p_{\text{end}}(en, j) \quad (3)$$

Let $\boldsymbol{A}$ be the answer to the query $\boldsymbol{q}$, and let Y_A be the set off all spans in all passages which exactly

match the string $\boldsymbol{A}$ -

$$Y_A = \{(st, en, j) \mid \boldsymbol{C_j}[st : en] = \boldsymbol{A}\}$$

In Cheng et al. (2020), the authors show that aggregating the probabilities of all the spans that match the same string, $p_{\text{string}}(\boldsymbol{A})$, helps improve performance:

$$p_{\text{string}}(\boldsymbol{A}) = \sum\{p_{\text{span}}(st, en, j) \mid (st, en, j) \in Y_A\} \quad (4)$$

2.2 Cross-Sample Attention for Fusion in Encoder

To extend the transformer with the ability for early global fusion of information in the encoder across input samples, we add k extra "global representation tokens" $\boldsymbol{G}$ as an input to the encoder E. The input embeddings for these tokens are initialized with untrained extra vocabulary tokens, one for each global representation token. By modifying the transformer attention mechanism's key and value, these global representation tokens attend to all tokens in all passages with cross-sample attention, and all tokens can attend to these global tokens.

In the transformer attention block in each transformer layer $\mathrm{E_l}$, let $\mathbf{Q}_\ell$, $\mathbf{K}_\ell$ and $\mathbf{V}_\ell$ be the attention's query, key and values respectively. Recall the attention function is then (Vaswani et al., 2017):

$$\text{Attention}(\mathbf{Q}_\ell, \mathbf{K}_\ell, \mathbf{V}_\ell) = \text{softmax}(\frac{\mathbf{Q}_\ell {\mathbf{K}_\ell}^T}{\sqrt{d_K}})\mathbf{V}_\ell$$

When the attention's query tokens $\mathbf{Q}_\ell$ are our global representation tokens, we change the key $\mathbf{K}_\ell^{\mathbf{G}}$ and Value $\mathbf{V}_\ell^{\mathbf{G}}$ to be the concatenation of all the encoded tokens of all the passages from the previous layer, as well as the global tokens, i.e.,

$$\mathbf{K}_\ell^{\mathbf{G}} = (\oplus_{\mathrm{i}}(\mathrm{E}_{\ell-1}(\boldsymbol{q}, \boldsymbol{C_j}))) \oplus (\mathrm{E}_{\ell-1}(\boldsymbol{G}))$$

where $\oplus$ is the concatenation operation across all the passages and the global tokens. The same process is also repeated for the Value $\mathbf{V}_\ell^{\mathbf{G}}$. This enables the representations for the tokens $\boldsymbol{G}$ to attend and fuse information from all the passages.

Similarly, tokens from any passage $\boldsymbol{C_j}$ can attend to the global tokens, along with other tokens from the same passage, i.e., the key for each passage is the concatenation of global tokens and passage tokens representations.

$$\mathbf{K}_\ell^{\mathbf{i}} = \mathrm{E}_{\ell-1}(\boldsymbol{q}, \boldsymbol{C_j}) \oplus \mathrm{E}_{\ell-1}(\boldsymbol{G}),$$

Because only the global representation tokens attend to all tokens, our method results in only 10% overhead compared to a vanilla transformer, as we show theoretically and empirically in Appendix A.

2.3 Global Probability Space

Our model now has information flow across samples, but when calculating the final probabilities in Equation (1), Equation (2) and Equation (3), we ignore the information from presence of the answer span in other passages. Cheng et al. (2020) attempts to fix this by modifying Equation (2) to softmax over all passages. However, due to Equation (3), separate scores for start and end probability assigns incorrectly high score to long spans made by combining start/end of separate answer occurrence spans within the same passage.

To address this issue, we modify the span probability calculation to first concatenate the start and end embeddings, and classify this score for a span directly, and finally softmax across all the spans across all the passages -

$$\begin{aligned}
\mathbf{h_{span}}(\mathbf{st}, \mathbf{en}, \mathbf{j}) &= \mathbf{h_{st,j}} \oplus \mathbf{h_{en,j}} \\
s_{\text{span}}(st, en, j) &= \mathrm{W}_{\text{span}}(\mathbf{h_{span}}(\mathbf{st}, \mathbf{en}, \mathbf{j})) \\
p_{\text{span}}(st, en, j) &= \sigma_{\text{st,en,j}}(s_{\text{span}}(st, en, j))
\end{aligned} \quad (5)$$

where W_{span} is non-linear classifier, $s_{\text{span}}(st, en, j)$ is the logit score for the span $\boldsymbol{C_j}[st : en]$, and the softmax is over all possible spans st, en, j across all passages.

In practice, because answer-span lengths are rarely longer than some constant len_A, we can assign a score of zero to all such spans in Equation (5) and skip calculating their actual scores and probabilities.

Furthermore, following Cheng et al. (2020), we also aggregate the scores for all spans that are the same strings, as done in Equation (4). This approach, combined with our changes to the probability space in Equation (5) and the global representation tokens introduced in Section 2.2, enables us to calculate probabilities for the answers in the global space of all strings in all passages. The span embeddings and corresponding probabilities are refined using information from other passages from the very first layer. The model is then trained with Maximum Marginal Likelihood (MML) objective of the correct answer string.

Model	Model Type	# Params	NQ	TQA	WebQ
Base Models					
REALM (Guu et al., 2020)	Extractive	110M	40.4	-	40.7
DPR (Karpukhin et al., 2020)	Extractive	110M	41.5	56.8	34.6
ANCE (Xiong et al., 2021)	Extractive	110M	46.0	57.5	-
UnitedQA (Cheng et al., 2021)	Extractive	110M	47.7	66.3	-
FiD (Izacard and Grave, 2021b)	Generative	220M	48.2	65.0	45.2
KG-FiD (Yu et al., 2022)	Generative	220M	49.6	66.7	-
FiD-KD (Izacard and Grave, 2021a)	Generative	220M	49.6	68.8	46.8
EMDR^2 (Singh et al., 2021)	Generative	220M	52.5	**71.4**	48.7
FiE (Ours)	Extractive	110M	**54.9**	68.2	50.8
FiE + PAQ (Ours)	Extractive	110M	53.3	68.2	**53.9**
Larger Models					
RAG (Lewis et al., 2020)	Generative	400M	44.5	56.1	45.2
R1-D1 (Fajcik et al., 2021)	Extractive	330M	50.8	65.0	-
FiD (Izacard and Grave, 2021b)	Generative	770M	51.4	67.6	-
UnitedQA (Cheng et al., 2021)	Extractive	330M	51.8	68.9	48.0
KG-FiD (Yu et al., 2022)	Generative	770M	53.4	69.8	-
FiD-KD (Izacard and Grave, 2021a)	Generative	770M	53.7	72.1	-
UnitedQA (Cheng et al., 2021)	Hybrid	1.87B	54.7	70.5	-
R2-D2 (Fajcik et al., 2021)	Hybrid	1.29B	55.9	69.9	-
FiE (Ours)	Extractive	330M	**58.4**	71.6	52.4
FiE + PAQ (Ours)	Extractive	330M	**58.4**	**72.6**	**56.3**

Table 1: End-to-end Open QA Exact-Match results on Natural Questions (NQ), TriviaQA(TQA) and Web Questions(WebQ) test sets. PAQ is data-augmentation described in Section 4.2. Our model outperforms all other models on Natural Questions and Web Questions, and all models of the same size on TriviaQA.

3 Experimental Setup

3.1 Datasets

We perform our experiments on three of the most popular open domain question answering datasets - Natural Questions (NQ) (Kwiatkowski et al., 2019), TriviaQA (Joshi et al., 2017) and WebQ (Berant et al., 2013). For NQ and TriviaQA, we used the short answer subsets processed by ORQA (Lee et al., 2019). For WebQ, we use the dataset version provided by EMDR2 Singh et al. (2021). We use the pre-processed Wikipedia dump from Dec. 20, 2018, provided by FiD-KD Izacard and Grave (2021a) as our external knowledge base. Dataset statistics and download links can be found in the supplementary material Table 11 and Appendix I.

3.2 Models

We demonstrate the effectiveness of our method on a range of model sizes and capacities, from Bert (Devlin et al., 2019) tiny (4M params), Electra (Clark et al., 2020) small (14M), base (110M) to large (330M). We use Electra model instead of BERT as Clark et al. (2020) and UnitedQA (Ma et al., 2022) show that Electra outperforms BERT as a reader in open-domain QA, and previous SOTA R2-D2 (Fajcik et al., 2021) also used Electra. We carry out most of our ablations and analysis on Electra base, as it was faster to train compared to the large model, while still representing a reasonably strong backbone.

3.3 Training Details

We use all original hyper-parameters of Electra, use $k = 10$ global tokens due to GPU memory constraints with the large model, and set the maximum number of answer tokens $len_A = 15$. We use $n = 100$ passages retrieved from the retriever of Izacard and Grave (2021a), except for WebQuestions, for which we use the retriever from Singh et al. (2021) with $n = 50$.

We run no hyper-parameter tuning and use all

the original hyper-parameters except for the total number of training steps, details of which are in Table 10 in the supplementary. As our experiments are compute-intensive, we ran only a few run with varying seeds, as reported in Table 9 in the supplementary. The experiments were run on 8x80GB Nvidia A100s with 800GB RAM and 4x32-core CPUs, and each experiment took around 1 day for NQ and 2 days for TriviaQA with large models. Inference was run on the same system, and took 2 minutes.

4 Results

4.1 Open-domain QA

As we show in Table 1, both our base and large models outperform all previous approaches on the Natural Questions dataset, achieving a new State-of-the-art Exact Match scores of 58.4 for the large model and 54.9 for the base model, with gains of 2.5 and 2.4 EM over prior works. Our method even outperforms hybrid ensemble models (Cheng et al., 2021) with $6x$ more parameters while achieving much higher throughput during inference.

We also outperform all previous models on WebQuestions, with scores 52.4 and 50.8 for large and base models, respectively, beating previous scores by 4.4 and 2.1 EM, respectively. On TriviaQA, our method outperforms all equal-sized models while being competitive with the current SOTA model (Izacard and Grave, 2021a), with $2x$ more parameters, achieving a score of 71.6 Exact Match with the large model.

We observed that generative models perform much better on TriviaQA. Let us compare the performance drop of extractive models on the NQ and TriviaQA datasets with respect to the generative Fid-KD model. In Table 1, for each dataset where we have both NQ and TriviaQA results, we calculate the performance drop with respect to the generative FiD-KD model and take the average. We observe an average drop of 3.7 EM score on NQ compared to a much higher drop of 7.2 EM score on TriviaQA. This might explain the relatively smaller improvements of FiE for TriviaQA.

4.2 Data Augmentation for Open-domain QA

To study the impact of synthetic data augmentation on our model, we randomly sample 6M Question-and-Answer samples from PAQ (Lewis et al., 2021), use a pre-trained retriever from FiD-KD (Izacard and Grave, 2021a) to retrieve passages, and use this data to pre-train our model for 1 epoch. We show the results of this data augmentation in Table 1.

We observe large gains of 3.9 EM and 2.1 EM in WebQuestions for large and base models, respectively, establishing new state-of-the-art results, perhaps due to the small size of the WebQuestions data training set. The results on TriviaQA are mixed - we observe no improvements for the base model, but the large model scores improve by 1 EM, resulting in a new state-of-the-art score of 72.6 EM. On NQ, we surprisingly observe a drop in model performance, perhaps because the model is over-fitting to the synthetic data.

4.3 Open-domain QA across Model Sizes and Capacities

Stronger models may not benefit as much from methods that improve the performance of weaker models. Table 2 provides the performance of different extractive readers augmented with our global information fusion approach. These include the Bert (Devlin et al., 2019) tiny, and Electra (Clark et al., 2020) small, base and large, ranging in size from 4M parameters to 330M. These results demonstrate the efficacy of our proposed approach across a wide range of different model sizes and capacities.

Model	Params	EM	Δ EM
Bert Tiny	4M	26.0	+11.1
Electra Small	14M	43.1	+2.6
Electra Base	110M	54.9	+6.9
Electra Large	330M	58.4	+7.7

Table 2: Effect of Model Size on Performance (Exact Match) on Natural Questions test set. The last column is the improvement of our method over a baseline without any global representation tokens.

5 Ablation Studies

5.1 Ablation of Model Components

Both components of our approach, Cross-Sample Attention for Fusion in Encoder, and the Global Probability Space, provide significant improvements over the baseline (Table 3). Our proposed probability space increases scores by 2.3 EM on NQ, while fusion in encoder increases the scores by 3.2 EM.

Furthermore, our proposed approaches strongly complement each other. Global representation fusion allows information flow across documents to

get global representations, and the global probability space provides a better training signal for FiE by necessitating information aggregation of the answer spans across documents. Adding both of these together results in a total increase of 9.2 EM.

Model	EM(NQ)
Baseline (Electra base)	45.7
Baseline + global prob. space	48.0
Baseline + global repr. fusion	48.9
Baseline + both (FiE)	**54.9**

Table 3: Ablation of FiE Model Components on Natural Questions with Electra Base model, Exact Match scores.

5.2 Alternatives for Global Representation Fusion

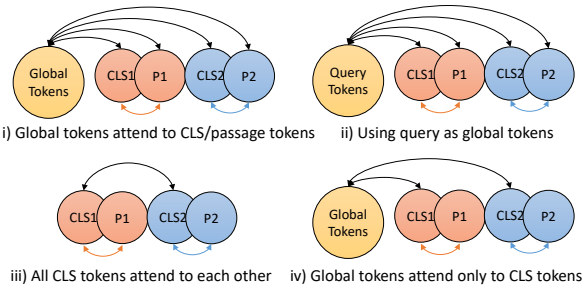

Figure 2: Alternatives for Global Representation Fusion

We use cross-sample attention to enable global information fusion in the encoder. In addition to the approach described in Section 2.2, we also study different alternative variants of the fusion mechanism, as shown in Figure 2. Using the query token as input for the global tokens (instead of separate trained embeddings for each global token) may allow for better contextualization(Figure 2 (ii)).

Alternatively, as the "CLS" token can capture some information for the entire passage, simply allowing all the "CLS" tokens to attend to one another can enable some global information flow(Figure 2 (iii)). Lastly, we restrict the cross attention of global tokens, such that they attend only to the CLS tokens of different passages(Figure 2 (iv)).

The results corresponding to these variations are reported in Table 4. Using dedicated tokens for the global tokens, rather than re-using the query tokens, results in better performance. We conjecture that it is easier for the model to learn how to fuse information if these tokens are consistently the same across different examples, rather than changing for each query. All these approaches improve the performance over the baseline model (no fusion), highlighting the importance of enabling information fusion in the encoder itself. Moreover, the model performance keeps increasing as we increase the scope of information fusion.

Model	EM(NQ)
i) Ours (FiE)	**54.9**
ii) Using query as global tokens	52.2
iii) All CLS attend to each other	51.5
iv) Global toks attend only to CLS	50.0
No fusion	48.0
Simple concatenation (10 context)	41.6

Table 4: Exact Match scores of alternatives to our encoder fusion tokens, with Electra Base model, on Natural Questions Test set. All models were trained with the global probability space. The numbers at the beginning of the rows correspond to the figures shown in Figure 2.

We also tested a simple baseline of concatenating the input passages. Electra is trained with maximum sequence length 512. Since 10 concatenated passages have a sequence length approximately 1500, we extended and re-trained the position embeddings. This method obtained 41.6 EM on NQ, as shown in Table 4, compared to our method's score of 49.1 EM when given 10 passages as shown in Figure 5. We conjecture that such a low score may be due to non-pretrained position embeddings, suggesting that approaches such as concatenating are perhaps best used with models trained with longer sequence length. Our method on the other hand, suffers from no such limitation.

5.3 Alternatives for Global Probability Space

We investigate several alternatives for our global probability space for span score calculation (Figure 3). Figure 3 (i) shows our approach, as described in Section 2.3. In "Non-conditional start & end logits" (Figure 3 (ii)), we consider separately calculating the start and end span scores. In "Separate start and end probability space" (Figure 3 (iii)), the start and end tokens are separately softmaxed across all possible tokens, as done in Cheng et al. (2020).

In "string probability space" (Figure 3 (iv, v)), we make our probability space the space of all strings, softmaxing across the aggregated scores of

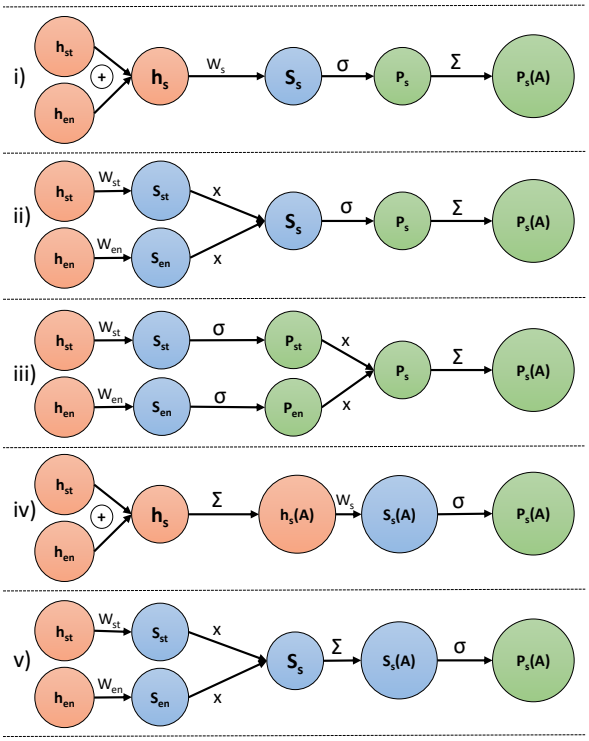

Figure 3: Alternatives for Global Probability Space. ($\oplus$: Concatenation, W: classifier, x: multiplication, σ: softmax, $\sum$: Summation, $\mathbf{h}$: encoding, s: logit score (obtained using a classifier), p: probability (calculated via softmax)), st: start position, en: end position, $\boldsymbol{A}$: answer string)

all strings. We have two variants, in "Span representation sum" (Figure 3 (iv)), the score for a string is classified by the aggregated embeddings of all spans. Figure 3 (v) shows "Logits sum", where the string score is calculated by aggregating the scores of the spans.

Our probability space and score calculation approach outperforms all the other approaches (Table 5). Cheng et al. (2020)'s separate start and end probability space tends to incorrectly assign a high probability to spans made by combining start and end probabilities of different occurrences of the answer string. The string probability space methods also under-perform, perhaps because they aggregate before the softmax.

5.4 Combination of HardEM and Maximum Likelyhood Objectives

Previous work (Cheng et al., 2020, 2021) has shown that adding global HardEM objective, using the maximum logit in positive paragraphs, may yield better results. We combine multiple variations of this HardEM with 0.1 weightage added to our MML objective. All variations of HardEM decrease model performance (Table 6).

Model	EM
i) Ours	**52.2**
ii) Non-conditional start & end logits	50.8
iii) Separate start and end prob space	48.1
iv) Span repr. sum + string prob space	47.3
v) Logits sum + string prob space	44.1

Table 5: Exact Match scores of alternatives to the global probability space, with Electra Base model, on Natural Questions Test set. All models were trained with using query as global tokens.

Model	EM(NQ)
MML	**52.2**
HardEM max	49.5
HardEM min	49.7
HardEM 80% probability mass	49.9

Table 6: Exact Match scores of alternatives to the MML training objective, with Electra Base model, on Natural Questions Test set. All models were trained with using query as global tokens.

Because the HardEM objective may be more susceptible to false-positive occurrences in the paragraphs, it may negatively impact performance when fusing across passages. Vanilla MML assumes all spans are correct, while HardEM assumes only one is. In our modeling, the model is free to decide the relative correctness of each answer occurrence, with direct supervision only from the answer string without any additional assumptions.

6 Analysis

6.1 Information Content of Global Tokens

The global tokens were motivated to enable information flow across passages, by gathering information from all tokens. A very pertinent information to gather would possibly be the answer representation. To analyze the information content of global representation tokens, we rank all the tokens $C_{i,j}$ in the input passages by the similarity of their final encoded representations $\mathbf{h_{i,j}}$ to the final encoder representations of the global tokens $\mathbf{h_G}$. Let us call a token an "answer token" if it is present in the ground truth answer $\boldsymbol{A}$.

We find that in 49% of examples from NQ test set for Electra Base (53% for Large), the most similar input token to the global tokens is an answer

token. This increases to 70% if we only consider the examples the model answers correctly. Furthermore, for 43% of examples, all the answer tokens are present in the 10 tokens most similar to global tokens after removing duplicates. Even without any explicit training objective to do so, the global tokens implicitly gather the answer representation.

6.2 Importance of Global Tokens

The model places a large importance on the global tokens, with the average attention to global tokens 2.4 and 2.8 times expected attention value, as shown in Table 7 using NQ test set. This is even higher than the attention to the query tokens, and 17% of the attention probability mass is on the global tokens, despite the number of the global tokens being only 6% of the input sequence length.

Model Size	Base	Large
Attn. to query token	2.1x	2.2x
Attn. to global token	2.4x	2.8x
Attn. prob. mass query token	14%	17%
Attn. prob. mass global token	17%	18%

Table 7: Analysis of attention to global tokens. The first two rows show the ratio of attention to global/query tokens compared to expected average attention. The last two lines show the total attention probability mass.

6.3 Cross-sample Fusion by Global Tokens

We use attention-rollout (Abnar and Zuidema, 2020) to measure how much cross-sample fusion is enabled by our model. Attention rollout models the information flow via attention as a directed graph through the layers, and assumes the representations of input tokens are linearly combined through the layers based on the attention weights. In our model, effective attention to other passages is achieved via passage tokens attending to the global tokens, which in turn attend to other passages.

For the base model, we find that 46% of the attention rollout probability mass for a given input passage is on other passages. This increases to 69% for the large model, clearly demonstrating that our method successfully achieves cross-sample fusion.

6.4 Effect of Number of Global Tokens

Figure 4 provides variation in the model performance with the number of global tokens. More global tokens will lead to more model capacity for fusion across passages, and the results show that increasing the number of global representation tokens leads to better performance. Although we observe significant improvements up to 50 tokens, we use 10 global tokens in all other experiments due to GPU memory constraints with the large model.

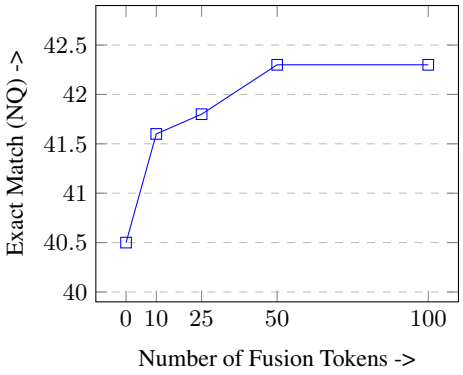

Figure 4: Effect of Number of Global Tokens on Model performance, in Exact-Match on NQ test set, with Electra-Small Model.

6.5 Effect of Number of Passages

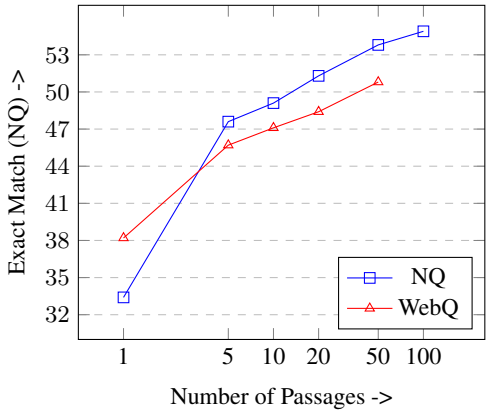

Figure 5: Effect of Number of Passages on a logarithmic scale, in Exact-Match on NQ and WebQ test set, with Electra-Base Model.

Increasing the number of passages will directly provide more information to the model, allowing potentially better results. However, this may come with increased false positives and noise from irrelevant passages. Wang et al. (2019) shows that the performance of extractive models peaks around 10 to 20 passages.

Figure 5 shows that the performance of our proposed model consistently improves with increase in the number of passages, across multiple datasets. With the addition of global information fusion, our extractive models are able to utilize the extra infor-

mation from multiple passages and hence improve the overall performance. The performance does not seem to have saturated, and seems to almost be in a power-law relationship to the number of passages. We only use 100 passages in all our experiments for fair comparison with prior works (50 for WebQ).

7 Related Works

7.1 Open-domain Reader Models

Reader models for Open-domain QA are required to read multiple documents (often more than 100 passages) (Izacard and Grave, 2021b; Fajcik et al., 2021) to avoid missing the target passage from the large-scale knowledge base. Reading passages jointly at such a scale using a transformer encoder is computationally intractable due to the quadratic complexity of the transformer. To reduce the complexity, Knowledge-GPT (Zhao et al., 2020) selects key sentences from multiple passages and then encodes them all joint using GPT (Radford et al., 2019). OrQA (Lee et al., 2019), REALM (Guu et al., 2020), and RAG (Lewis et al., 2020) encode each passage separately and marginalize the predicted answer probabilities. However, these early approaches perform poorly due to a lack of information exchange across multiple passages for more contrastive representations during a prediction.

7.2 Information Fusion for Reader Models

Cross-sample information fusion in reader models is most commonly achieved by letting the decoder attend to the encoded inputs of multiple passages, as first proposed by FiD (Izacard and Grave, 2021b), and then adopted by Izacard and Grave (2021a); Singh et al. (2021); Yu et al. (2022); Fajcik et al. (2021); Cheng et al. (2021). Our proposed model achieves early fusion by adding global tokens to the encoder, ensuring the encodings of different sample passages are aware of global information.

BigBird (Zaheer et al., 2020), Longformer (Beltagy et al., 2020), LongT5 (Guo et al., 2021) focused on efficient attention to long sequences by allowing only a few tokens to have the full attention on all tokens in a single input passage, but these approaches did not consider fusing information across samples and often require expensive pre-training.

A contemporaneous work Zcode++ (He et al., 2022) proposes encoder fusion by concatenating all input tokens in the last encoder layers, also called fusion-in-encoder(FiE). However, the quadratic complexity of attention results in an overhead of $8x$ compared to our method, as shown in Appendix A.

7.3 Global Probability Space for Readers

To deal with multiple input passages, initial approaches for extractive models used a pipeline first to select a paragraph and then extract the answer (Guu et al., 2020; Karpukhin et al., 2020; Xiong et al., 2021). (Clark and Gardner, 2018; Wang et al., 2019) added global normalization where the answer probability was normalized over multiple candidates, while (Wang et al., 2018) also aggregated answer confidence from multiple passages based on strength and coverage. However, these approaches focus on reranking/normalizing the answer probability from candidate spans after the passages have been processed independently.

7.4 Cross-sample Information Fusion

TaBERT (Yin et al., 2020) proposed cross-sample attention across different rows of tabular data. Unlike TaBERT, we show that our approach is applicable to general textual data instead of vertically aligned tabular data. Cross-attention is often used in multi-modal models, to fuse information across modalities such as text and image (Ilinykh and Dobnik, 2022; Miech et al., 2021). Most similar to our approach, Chen et al. (2021) fuses information from multiple modalities or from different patches of the same image by fusing the CLS tokens. Our approach explores fusing information across different samples and shows that using global tokens outperforms using CLS tokens in this context.

8 Conclusion

We propose FiE: Fusion-in-Encoder, where we extend transformer encoders with the ability to fuse information across multiple input samples using global representation via cross-sample attention and propose a global probability space for answer spans. Facilitating this information flow across samples enables our proposed method to achieve state-of-the-art performance in 3 popular open-domain questions answering datasets by 2.5 EM on NQ, and 0.5 EM on TriviaQA, while simultaneously reducing parameter count and inference latency, and 4.4 EM on WebQ. Detailed ablations and analyses further demonstrate the effectiveness of our proposed method across multiple model sizes and the capability to fuse information from a large number of samples.

Limitations

Adding more global representation tokens almost linearly increases our model's GPU memory usage and compute requirements. While higher number of these fusion tokens (up to 50) results in better performance, we use a smaller number (10) to limit the compute and memory requirements. Furthermore, our global probability space requires classifying the embeddings for all possible spans; therefore, increasing maximum answer length will also linearly increase the number of possible spans and hence the memory, making our model unsuitable for datasets with very long answers.

Similarly, while our model continues to improve in performance and seems to have not converged even on using 100 passages, using a higher number of passages is difficult with large models due to GPU memory constraints. Also, our extractive model is not suitable for generative QA tasks, where the model may be expected to somewhat rephrase the spans in the passages.

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, Online. Association for Computational Linguistics.

Vidhisha Balachandran, Ashish Vaswani, Yulia Tsvetkov, and Niki Parmar. 2021. Simple and efficient ways to improve REALM. In *Proceedings of the 3rd Workshop on Machine Reading for Question Answering*, pages 158–164, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *CoRR*, abs/2004.05150.

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.

Chun-Fu (Richard) Chen, Quanfu Fan, and Rameswar Panda. 2021. Crossvit: Cross-attention multi-scale vision transformer for image classification. In *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*, pages 347–356. IEEE.

Hao Cheng, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2020. Probabilistic assumptions matter: Improved models for distantly-supervised document-level question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5657–5667, Online. Association for Computational Linguistics.

Hao Cheng, Yelong Shen, Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2021. UnitedQA: A hybrid approach for open domain question answering. 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 3080–3090, Online. Association for Computational Linguistics.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855, Melbourne, Australia. Association for Computational Linguistics.

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Martin Fajcik, Martin Docekal, Karel Ondrej, and Pavel Smrz. 2021. R2-D2: A modular baseline for open-domain question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 854–870, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Mandy Guo, Joshua Ainslie, David C. Uthus, Santiago Ontañón, Jianmo Ni, Yun-Hsuan Sung, and Yinfei Yang. 2021. Longt5: Efficient text-to-text transformer for long sequences. *CoRR*, abs/2112.07916.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Retrieval augmented language model pre-training. In *Proceedings of the 37th International Conference on Machine Learning, ICML 2020, 13-18 July 2020, Virtual Event*, volume 119 of *Proceedings of Machine Learning Research*, pages 3929–3938. PMLR.

Pengcheng He, Baolin Peng, Liyang Lu, Song Wang, Jie Mei, Yang Liu, Ruochen Xu, Hany Hassan Awadalla, Yu Shi, Chenguang Zhu, Wayne Xiong, Michael Zeng, Jianfeng Gao, and Xuedong Huang. 2022. Z-code++: A pre-trained language model optimized for abstractive summarization. *CoRR*, abs/2208.09770.

Nikolai Ilinykh and Simon Dobnik. 2022. Attention as grounding: Exploring textual and cross-modal attention on entities and relations in language-and-vision transformer. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 4062–4073, Dublin, Ireland. Association for Computational Linguistics.

Gautier Izacard and Edouard Grave. 2021a. Distilling knowledge from reader to retriever for question answering. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Gautier Izacard and Edouard Grave. 2021b. 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.

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.

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.

Haejun Lee, Akhil Kedia, Jongwon Lee, Ashwin Paranjape, Christopher D. Manning, and Kyoung-Gu Woo. 2021. You only need one model for open-domain question answering.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics.

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*, 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. 2020. 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*.

Shayne Longpre, Kartik Perisetla, Anthony Chen, Nikhil Ramesh, Chris DuBois, and Sameer Singh. 2021. Entity-based knowledge conflicts in question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7052–7063, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Kaixin Ma, Hao Cheng, Xiaodong Liu, Eric Nyberg, and Jianfeng Gao. 2022. Open domain question answering with a unified knowledge interface. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1605–1620, Dublin, Ireland. Association for Computational Linguistics.

Antoine Miech, Jean-Baptiste Alayrac, Ivan Laptev, Josef Sivic, and Andrew Zisserman. 2021. Thinking fast and slow: Efficient text-to-visual retrieval with transformers. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2021, virtual, June 19-25, 2021*, pages 9826–9836. Computer Vision Foundation / IEEE.

Sabrina J. Mielke, Arthur Szlam, Y-Lan Boureau, and Emily Dinan. 2020. Linguistic calibration through metacognition: aligning dialogue agent responses with expected correctness. *ArXiv preprint*, abs/2012.14983.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Devendra Sachan, Mostofa Patwary, Mohammad Shoeybi, Neel Kant, Wei Ping, William L. Hamilton, and Bryan Catanzaro. 2021. End-to-end training of neural retrievers for open-domain question answering. 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 6648–6662, Online. Association for Computational Linguistics.

Devendra Singh, Siva Reddy, Will Hamilton, Chris Dyer, and Dani Yogatama. 2021. End-to-end training of multi-document reader and retriever for open-domain question answering. *Advances in Neural Information Processing Systems*, 34.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all

you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008.

Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2018. Evidence aggregation for answer re-ranking in open-domain question answering. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net.

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.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N. Bennett, Junaid Ahmed, and Arnold Overwijk. 2021. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net.

Peng Xu, Davis Liang, Zhiheng Huang, and Bing Xiang. 2021. Attention-guided generative models for extractive question answering. *ArXiv preprint*, abs/2110.06393.

Pengcheng Yin, Graham Neubig, Wen-tau Yih, and Sebastian Riedel. 2020. Tabert: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*, pages 8413–8426. Association for Computational Linguistics.

Donghan Yu, Chenguang Zhu, Yuwei Fang, Wenhao Yu, Shuohang Wang, Yichong Xu, Xiang Ren, Yiming Yang, and Michael Zeng. 2022. Kg-fid: Infusing knowledge graph in fusion-in-decoder for open-domain question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pages 4961–4974. Association for Computational Linguistics.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*.

Rowan Zellers, Ari Holtzman, Hannah Rashkin, Yonatan Bisk, Ali Farhadi, Franziska Roesner, and Yejin Choi. 2019. Defending against neural fake news. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 9051–9062.

Xueliang Zhao, Wei Wu, Can Xu, Chongyang Tao, Dongyan Zhao, and Rui Yan. 2020. Knowledge-grounded dialogue generation with pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3377–3390, Online. Association for Computational Linguistics.

A Memory, Latency and Compute Comparisons

Our proposed method has minimal overheads of approximately 10% compared to vanilla Electra. We can verify this both empirically, as we show in Table 8 as well as theoretically, as shown in Equation (9). The previous SOTA method R2-D2 (Fajcik et al., 2021) reported approximately $3x$ latency compared to vanilla Electra, making FiE's latency approximately 35% R2-D2.

Model	TrainIter/s	Mem.(GB)
FiD	2.4	33.4
Vanilla Electra	2.6	35.0
No Global Tokens	2.5	36.1
FiE	2.3	37.4

Table 8: Training and memory overheads of our method compared to vanilla Electra during training on NQ, bench-marked with 1 batch size (of 100 passages) on 1 A100 40GB. TrainIter/s refers to number of training iteration per second, and Mem. refers to the GPU memory utilized by the model.

Let L be the number of layers, N be the number of input passages, each of sequence length S, and G be the number of global fusion tokens. For a base model on NQ, these values are $L = 12$, $N = 100$,

$S = 250$, and $G = 10$. The compute requirement of a vanilla transformer will approximately be :

$$\text{Vanilla} = \mathcal{O}(LNS^2) \quad (6)$$

For FiE, because the S passage tokens each attend to S passage tokens and G global tokens, and because the G global tokens each attend to NS passage tokens and G global tokens, the compute will approximately be -

$$\text{FiE} = \mathcal{O}(LNS(S+G) + LG(NS+G)) \quad (7)$$
$$= \mathcal{O}(LNS^2 + 2LNSG + LG^2) \quad (8)$$

Diving Equation (8) with Equation (6), the compute cost of FiE compared to vanilla transformer is approximately 10% :

$$\approx (1 + \frac{2G}{S}) = (1 + \frac{2 * 10}{250}) \approx 1.1 \quad (9)$$

A contemporaneous work Zcode++ (He et al., 2022) proposes encoder fusion by concatenating all input tokens in the last encoder layers. The last layer then has quadratic complexity of $(NS)^2$. This approach results as in a much higher overhead however, as we show below :

$$\text{Zcode++} = \mathcal{O}((L-1)NS^2 + (NS)^2) \quad (10)$$
$$= \mathcal{O}(LNS^2 + (N-1)NS^2) \quad (11)$$

Diving Equation (11) with Equation (6), the compute cost of Zcode compared to vanilla transformer is approximately $9x$ for these settings :

$$\approx \mathcal{O}(1 + \frac{N-1}{L}) = (1 + \frac{99}{12}) \approx 9.3 \quad (12)$$

B Measures of Central Tendency and Model Stability

As our experiments are compute-intensive (due to encoding 100 context passages for every example), it is not feasible to perform multiple runs of all the experiments. We provide detailed ablations of our components instead.

We also ran a few runs to verify the model stability to different seeds/initialization and to check the effect of training steps, the results of which we provide in Table 9. The standard error is over 2 runs for each parameter. The model's performance is not affected much by different seeds.

Update Steps	EM	Std Err
3750	50.9	0.4
7500	50.6	0.2
11250	51.7	0.7
15000	**52.0**	0.2
22500	50.6	0.1

Table 9: Exact Match scores and Standard Error of runs with varying seeds of our method, with Electra Base model, on Natural Questions test set. Note that these runs were performed with query tokens as inputs for the global representation tokens.

C Hyper-Parameters

We used the all the original model/training hyper-parameters from Electra (Clark et al., 2020), and data-related parameters from FiD (Izacard and Grave, 2021b). Hyper-parameter search was performed for the number of update steps on NQ, 2 runs for each value in Table 9. Full details of hyper-parameters are shown in Table 10. The Context Length was restricted to 220 for the large model instead of 250 due to GPU memory constraints.

Parameters	Values
Optimization	
Warmup steps	10%
Update Steps	15000 [Table 9]
Learning rate	5e-5(large), 1e-4(base)
Layerwise LR Decay	0.9(large), 0.8(base)
Drop-out	0.1
Gradient clipping	1.0
Batch Size / GPU	1
Num GPUs	8
Grad Accum Steps	8
Batch Size (effective)	64
Scheduler	linear
Data/Modelling	
Context Length	220(large), 250(small/base/tiny)
Num Context	100(NQ, TriviaQA), 50(WebQ)
Max Answer Tokens	15 (train)
Max Query Tokens	28
Global Fusion Tokens	10

Table 10: Training Parameters.

D Dataset Descriptions

Table 11 provides the statistics and retrieval recall for all three datasets. The recall was calculated corresponding to top-100 documents for NQ and TriviaQA and top-50 for WebQ.

Dataset	# Train	# Dev	# Test	Recall
NQ	79K	8.8K	3.6K	88.7
TriviaQA	79K	8.8K	11K	87.3
WebQ	3.4K	361	2K	89.5

Table 11: Dataset Statistics

E WebQuestions Scores Starting from NQ

Due to the small size of the WebQ dataset, we also initialized our model by training it on NQ. The results are reported in Table 12. The large model surprisingly under-performs compared to the base model - we conjecture it may be because it is overfitting on the NQ dataset. The base model score of 53.5 on WebQ is higher than the previous SOTA of 48.7 by 4.8 EM, but this is not a fair comparison as this score uses extra data.

Model	WebQ
FiE Base	53.5
FiE Large	52.8

Table 12: EM scores on WebQ test set.

F Dev Scores for Corresponding Test Scores

Table 13, Table 14, Table 15, and Table 16 provide the dev results corresponding to the test numbers in the main paper.

Model	NQ	TQA	WebQ
Base Models			
FiE (Ours)	48.4	68.3	44.3
PAQ + FiE (Ours)	50.0	69.5	51.5
Large Models			
FiE (Ours)	51.4	71.6	49.9
PAQ + FiE (Ours)	53.0	72.7	50.1

Table 13: EM scores on NQ, TQA and WebQ development sets for Table 1.

G Training the Bert Tiny Model

We observed that the model trained using Bert Tiny did not converge after 15k update steps on the NQ dataset, and the dev performance was still increasing. Hence, we trained the models with different number of total steps. These results have been provided in Table 19. The results in Table 2 correspond to 50k update steps.

Model	EM
Electra Base (Baseline)	45.1
+ Global Prob Space	47.2
+ Global Repr Fusion	**48.4**

Table 14: Model Components Ablation on NQ with Electra Base model, EM scores on dev set for Table 3.

Model	NQ(EM)
Ours	**48.5**
Question Repr as Fusion Tokens	48.1
All CLS attend to all CLS	48.0
Fusion Tokens only attend to CLS	48.1
No Fusion	47.2

Table 15: Exact Match scores of alternatives to our encoder fusion tokens, with Electra Base model, on Natural Questions dev set, corresponding to Table 4

Model	NQ(EM)
Ours	**48.1**
Non-conditional start & end logits	47.8
Separate start and end prob space	46.8
Span repr. sum + string prob space	45.3
Logits sum + string prob space	44.0

Table 16: Exact Match scores of alternatives to the global probability space, with Electra Base model, on Natural Questions dev set, corresponding to Table 5

Model	NQ(EM)
MML	48.2
HardEM max	48.7
HardEM min	48.7
HardEM 80% probability mass	48.6

Table 17: Exact Match scores of alternatives to the MML training objective, with Electra Base model, on Natural Questions dev set, corresponding to Table 6

H Raw Values for Plots

The raw values used in the plots in Figure 4 and Figure 5 can be found in Table 20 and Table 21

Model	Params	NQ
Bert Tiny	4M	26.3
Electra Small	14M	42.4
Electra Base	110M	48.4
Electra Large	330M	51.4

Table 18: Effect of Model Size on Performance (Exact Match) on NQ dev sets, corresponding to Table 2.

# Steps	Dev	Test
15k	18.7	18.5 (+ 6.7)
30k	22.1	21.9 (+ 7.9)
50k	**26.3**	**26.0** (+11.1)

Table 19: Bert Tiny results with different number of steps on the NQ test set. The improvements over the baselines have also been provided similar to Table 2.

respectively.

# Global Fusion Tokens	NQ
0	40.5
10	41.6
25	41.8
50	42.3
100	42.3

Table 20: Effect of Number of Global Tokens on Model performance in Exact-Match on NQ dataset, with Electra-Small Model, corresponding to Figure 4.

# Passages	NQ	WebQ
1	33.4	38.2
5	47.6	45.7
10	49.1	47.1
20	51.3	48.4
50	53.8	**50.8**
100	**54.9**	-

Table 21: Effect of Number of Passages on Model performance in Exact-Match on NQ and WebQ dataset, with Electra-Base Model, corresponding to Figure 5.

I Links to Source Code and Datasets

The source code is based on the original implementation of FiD (Izacard and Grave, 2021b), which can be found at their Github. The modeling for the fused Electra model was implemented using HuggingFace (Wolf et al., 2020) by modifying the ElectraModel.

Data for the Wikipedia dump, Natural Questions, and TriviaQA were also downloaded from FiD's github.

For WebQ, the QA pairs were downloaded from EMDR^2's github, as well as their pre-trained WebQ checkpoint and Wikipedia context embeddings index. These were then used to retrieve the top-50 passages used in our experiments.

For PAQ, QA pairs were downloaded from PAQ's github, and then 6M pairs were randomly selected. FiD-KD's retriever pre-trained on NQ from their github was used to retrieve top-100 passages.

J Details of Evaluation Metrics

The evaluation script from FiD was used, which can be found here. The evaluation metric is "Exact Match".

This metric is the average of the per-example exact match score for a dataset. The per-example exact match score is either 0 or 1; 1 if the model's answer is exactly the same as the text of any ground truth answer after lower-casing, removing punctuation, and normalizing spaces; otherwise 0.