UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

# REALM: Retrieval-Augmented Language Model Pre-Training

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, Ming-Wei Chang

Presenter: Haozhong Guan

# Motivation

## Language models store a surprising amount of world knowledge

- BERT is able to predict the missing word in the following sentence:

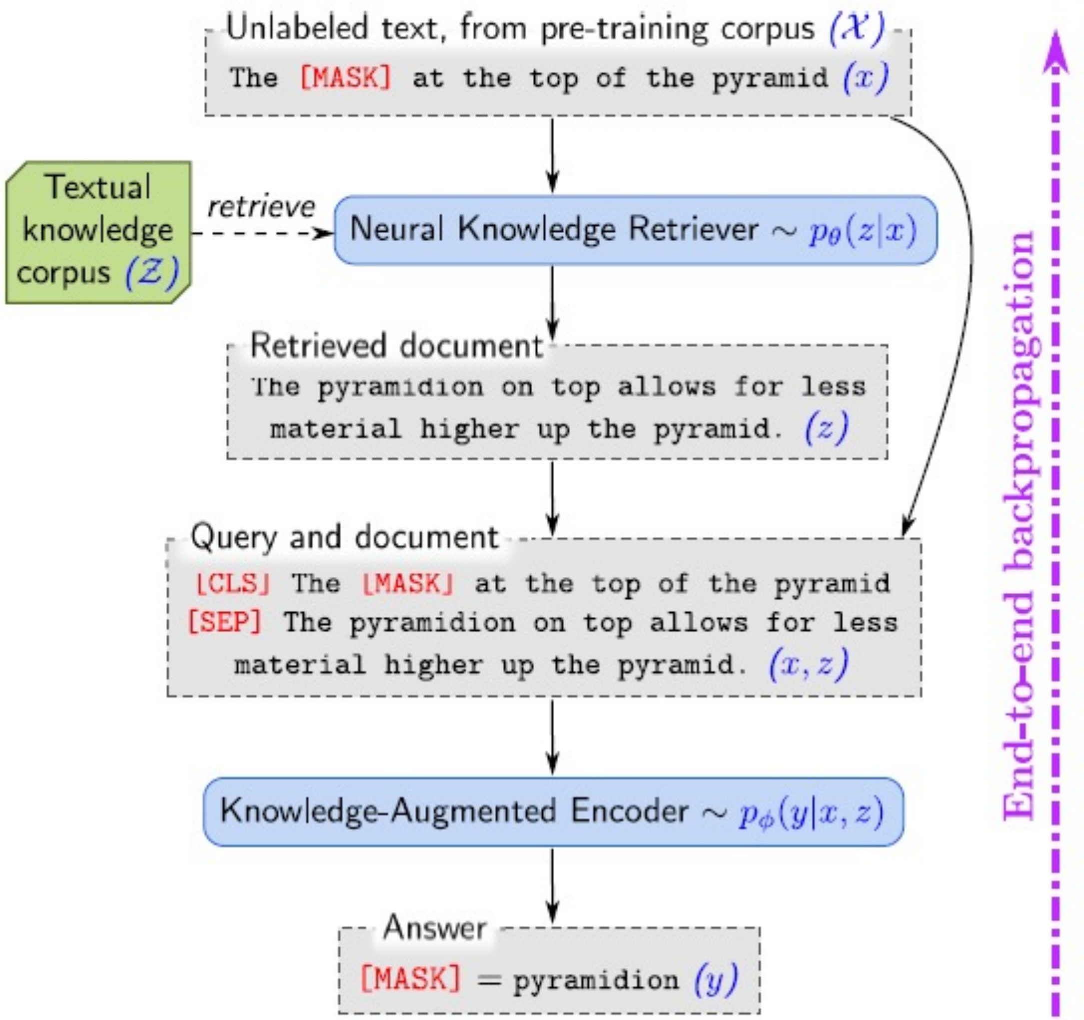
  “The ______ is the currency of the United Kingdom.” (answer: “pound”).

## Difficult to capture those knowledge

- The learned world knowledge is stored implicitly in the parameters of the underlying neural network.
- Storage space is limited by the size of the network

## Retrieval-Augmented Language Model (REALM) pre-training

- It augments language model pre-training algorithms with a learned textual knowledge retriever.
- Explicitly expose the role of world knowledge by asking the model to decide what knowledge to retrieve and use during inference.
- The key intuition of REALM is to train the retriever using a performance-based signal from unsupervised text



## Language model pre-training

- Focus on the masked language model (MLM) variant of pre-training popularized by BERT
- MLM is trained to predict the missing tokens in an input text passage

Given an unlabeled pre-training corpus (e.g., Wikipedia text), a training example $(x, y)$ can be generated by randomly masking tokens in a sampled piece of text:

$$x = \text{“}The\ [MASK]\ is\ the\ currency\ [MASK]\ the\ UK\text{”}$$

$$y = (\text{“}pound\text{”}, \text{“}of\text{”})$$

## Fine-tuning: Open-domain question answering (Open-QA)

- Objective: measure the model’s ability to incorporate world knowledge
- “Open domain” refers to the fact that the model does not receive a preidentified document that is known to contain the answer

An example can be:

$$x = \text{“}What\ is\ the\ currency\ of\ the\ UK?\text{”}$$
$$y = \text{“}pound\text{”}$$

# Approach

## REALM's generative process

- REALM takes some input $x$ and learns a distribution $p(y \mid x)$ over possible outputs
- Given an input $x$, we first retrieve possibly helpful documents $z$ from a knowledge corpus $Z$, which can be modeled as a sample from the distribution $p(z \mid x)$
- Then condition on both the retrieved $z$ and the original input $x$ to generate the output $y$, which can be modeled as $p(y \mid z, x)$
- To obtain the overall likelihood of generating $y$, we treat $z$ as a latent variable and marginalize over all possible documents $z$, yielding:

$$P(y \mid x) = \sum_{x \in Z} p(y \mid z, x)\, p(z \mid x)$$

## Knowledge Retriever: $p(z \mid x)$

The retriever is defined using a dense inner product model:

$$p(z \mid x) = \frac{\exp f(x, z)}{\sum_{z'} \exp f(x, z')},$$
$$f(x, z) = \mathtt{Embed}_{\mathtt{input}}(x)^\top \mathtt{Embed}_{\mathtt{doc}}(z)$$

The $f(x, z)$ is the relevance score between $x$ and $z$, which is defined as the inner product of the vector embeddings.

## Embedding Functions

The embedding functions are implemented using BERT-style Transformers.

Join spans of text by applying word-piece tokenization, separating them with [SEP] tokens, prefixing a [CLS] token, and appending a final [SEP] token.

$$\texttt{join}_{\texttt{BERT}}(x) = \texttt{[CLS]}\,x\,\texttt{[SEP]}$$
$$\texttt{join}_{\texttt{BERT}}(x_1, x_2) = \texttt{[CLS]}\,x_1\,\texttt{[SEP]}\,x_2\,\texttt{[SEP]}$$

Pass these into a Transformer, which produces one vector for each token and perform a linear projection to reduce the dimensionality of the vector, denoted as a projection matrix W:

$$\texttt{Embed}_{\texttt{input}}(x) = \mathbf{W}_{\texttt{input}}\mathrm{BERT}_{\mathrm{CLS}}(\texttt{join}_{\texttt{BERT}}(x))$$
$$\texttt{Embed}_{\texttt{doc}}(z) = \mathbf{W}_{\texttt{doc}}\mathrm{BERT}_{\mathrm{CLS}}(\texttt{join}_{\texttt{BERT}}(z_{\mathrm{title}}, z_{\mathrm{body}}))$$

## Knowledge-Augmented Encoder: $p(y \mid z, x)$

For the masked language model pretraining task:

$$p(y \mid z, x) = \prod_{j=1}^{J_x} p(y_j \mid z, x)$$

$$p(y_j \mid z, x) \propto \exp\left(w_j^\top \mathrm{BERT}_{\mathtt{MASK}(j)}(\mathtt{join}_{\mathrm{BERT}}(x, z_{\mathrm{body}}))\right)$$

where $BERT_{MASK(j)}$ denotes the Transformer output vector corresponding to the $j^{th}$ masked token, $J_x$ is the total number of [MASK] tokens in $x$, and $w_j$ is a learned word embedding for token $y_j$.

# Model architecture

## Knowledge-Augmented Encoder: $p(y \mid z, x)$

For Open-QA fine-tuning:

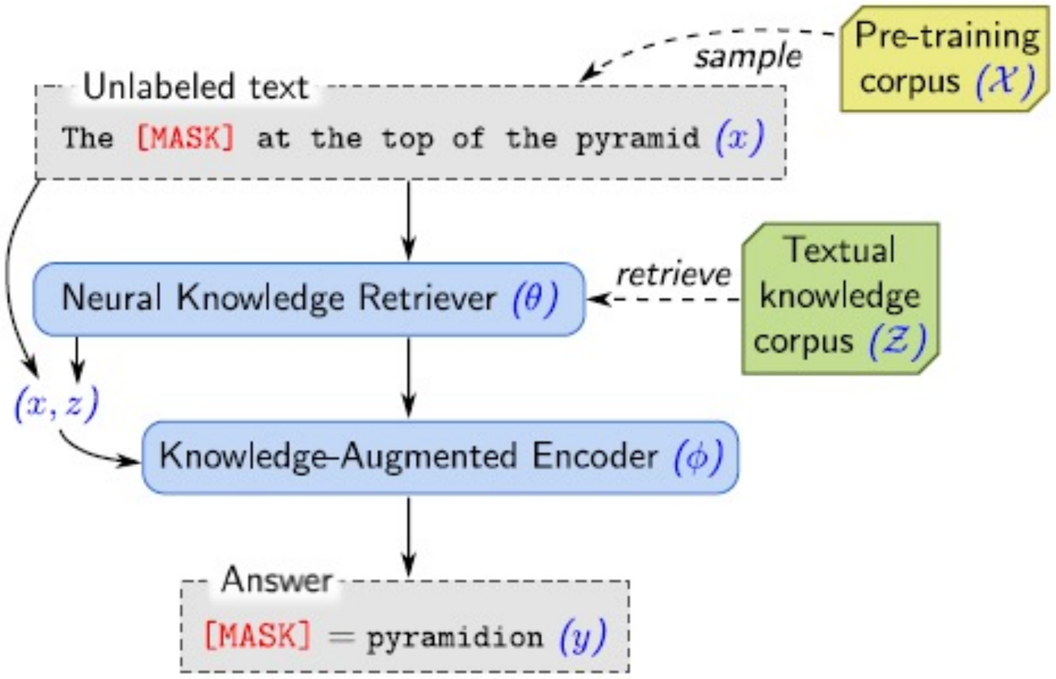
$$p(y \mid z, x) \propto \sum_{s \in S(z,y)} \exp\left(\mathrm{MLP}\left(\left[h_{\mathtt{START(s)}}; h_{\mathtt{END(s)}}\right]\right)\right)$$

$$h_{\mathtt{START(s)}} = \mathrm{BERT}_{\mathtt{START(s)}}(\mathtt{join}_{\mathtt{BERT}}(x, z_{\mathrm{body}})),$$

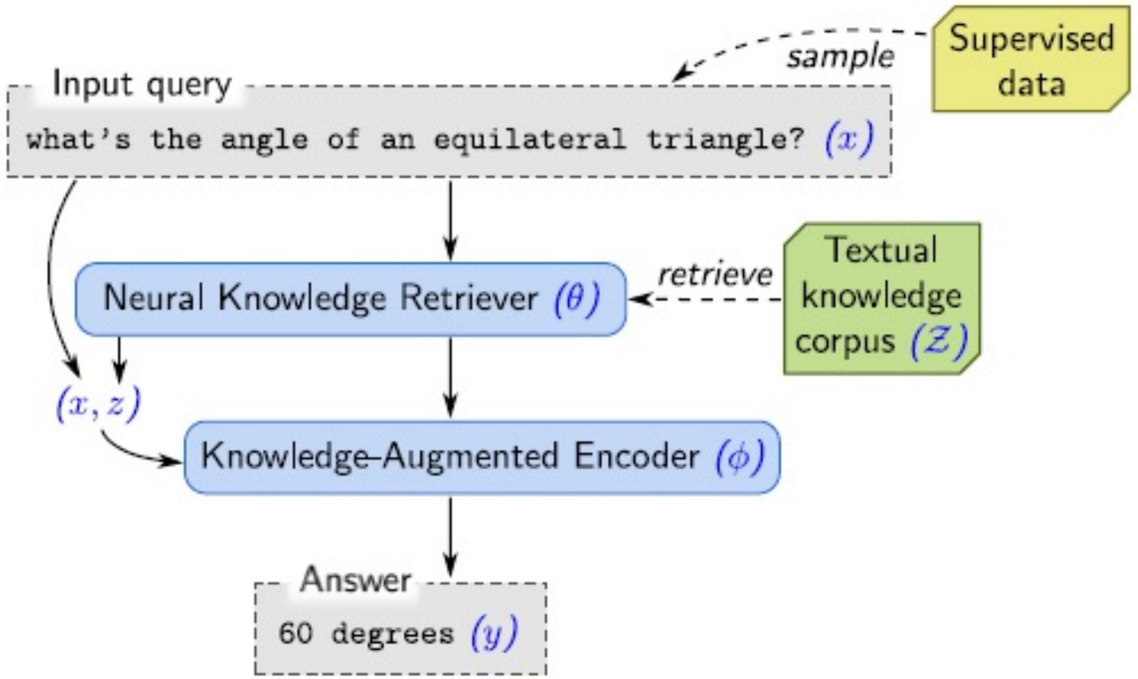
$$h_{\mathtt{END(s)}} = \mathrm{BERT}_{\mathtt{END(s)}}(\mathtt{join}_{\mathtt{BERT}}(x, z_{\mathrm{body}})),$$

where $S(z, y)$ is the set of spans matching answer string $y$ in some document $z$, $BERT_{START(s)}$ and $BERT_{END(s)}$ denote the Transformer output vector corresponding to the start and end token of span $s$, MLP denotes a feed-forward neural network.

# Model architecture




Pre-training




Fine-tuning

## Training Approach

- Objective: Maximize the log-likelihood $\log p(y \mid x)$ of the correct output $y$
- Compute the gradient of $\log p(y \mid x)$ with respect to the model parameters $\theta$ and $\Phi$ and  optimize using stochastic gradient descent
- Challenges: The marginal probability $p(y \mid x)$ involves a summation over all documents $z$ in the knowledge corpus $Z$, which significant computational challenge when $Z$ is large
- Approximation: sum over the top k documents with highest probability under $p(z \mid x)$

Why is this approximation reasonable?

## Find the top k documents

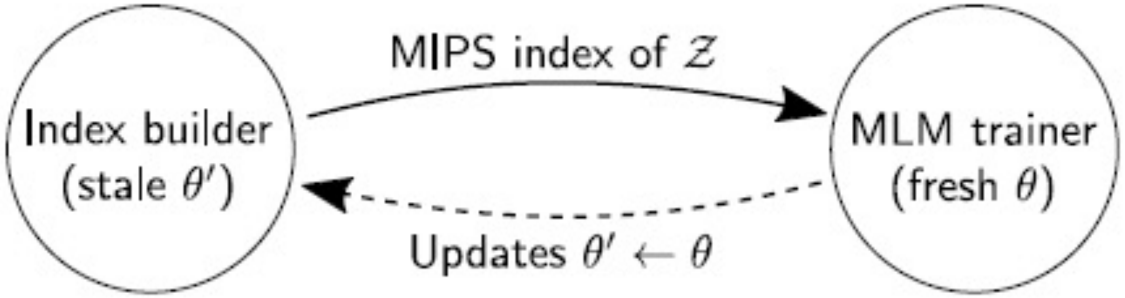
- Note: ordering of documents under $p(z \mid x)$ is the same as under the relevance score $f(x, z)$
- Employ Maximum Inner Product Search (MIPS) algorithms to find the approximate top k documents since $f(x, z)$ is an inner product
- Pre-compute $Embed_{doc}(z)$ for every $z \in Z$ and construct an efficient search index over these embeddings.
- Challenges: This data structure is not consistent with $p(z \mid x)$ if the parameters $\theta$ of $Embed_{doc}(z)$ are later updated
- Solution: refresh the index by asynchronously re-embedding and re-indexing all documents every several hundred training steps.

## Implementing asynchronous MIPS refreshes

Asynchronously refresh the MIPS index by running two jobs in parallel:

(1) Primary MLM trainer: performs gradient updates on the parameters

(2) Index builder: embeds and index the document



## What does the retriever learn?

A single step of gradient descent during REALM pre-training alters the relevance score by analyzing the gradient with respect to $\theta$:

$$\nabla \log p(y \mid x) = \sum_{z \in \mathcal{Z}} r(z) \nabla f(x, z)$$

$$r(z) = \left[ \frac{p(y \mid z, x)}{p(y \mid x)} - 1 \right] p(z \mid x).$$

For each document $z$, the gradient encourages the retriever to change the score $f(x, z)$ by $r(z)$

The multiplier r(z) is positive if and only if $p(y \mid z, x) > p(y \mid x)$, where $p(y \mid z, x)$ is the probability of predicting the correct output $y$ when using document $z$.

# Experiment

## Open-QA Benchmarks

This paper focus on datasets where the question writers did not already know the answer. This yields questions that reflect more realistic information-seeking needs.

### NaturalQuestions-Open

It consists of naturally occurring Google queries and their answers. Only questions that are categorized as "short answer type" with at most five tokens were kept .

### WebQuestions

It was collected from the Google Suggest API, using one seed question and expanding the set to related questions.

### CuratedTrec

It is a collection of question-answer pairs drawn from real user queries issued on sites such as MSNSearch and AskJeeves.

# Approaches

## Retrieval-based Open-QA

- Approaches use non-learned heuristic retrieval to select a small set of relevant documents:

  DrQA, HardEM, GraphRetriever, and PathRetriever

- Recent approaches have proposed to implement learnable retrieval using a MIPS index:

  ORQA, REALM

## Generation-based Open-QA

- Model Open-QA as a sequence prediction task: encode the question, and then decode the answer token-by-token based on the encoding.
- Most competitive and comparable generation-based baseline is fine-tuning T5 for Open-QA

## Results

*Table 1.* Test results on Open-QA benchmarks. The number of train/test examples are shown in paretheses below each benchmark. Predictions are evaluated with exact match against any reference answer. Sparse retrieval denotes methods that use sparse features such as TF-IDF and BM25. Our model, REALM, outperforms all existing systems.

| **Name** | **Architectures** | **Pre-training** | **NQ** (79k/4k) | **WQ** (3k/2k) | **CT** (1k /1k) | **# params** |
|---|---|---|---|---|---|---|
| BERT-Baseline (Lee et al., 2019) | Sparse Retr.+Transformer | BERT | 26.5 | 17.7 | 21.3 | 110m |
| T5 (base) (Roberts et al., 2020) | Transformer Seq2Seq | T5 (Multitask) | 27.0 | 29.1 | - | 223m |
| T5 (large) (Roberts et al., 2020) | Transformer Seq2Seq | T5 (Multitask) | 29.8 | 32.2 | - | 738m |
| T5 (11b) (Roberts et al., 2020) | Transformer Seq2Seq | T5 (Multitask) | 34.5 | 37.4 | - | 11318m |
| DrQA (Chen et al., 2017) | Sparse Retr.+DocReader | N/A | - | 20.7 | 25.7 | 34m |
| HardEM (Min et al., 2019a) | Sparse Retr.+Transformer | BERT | 28.1 | - | - | 110m |
| GraphRetriever (Min et al., 2019b) | GraphRetriever+Transformer | BERT | 31.8 | 31.6 | - | 110m |
| PathRetriever (Asai et al., 2019) | PathRetriever+Transformer | MLM | 32.6 | - | - | 110m |
| ORQA (Lee et al., 2019) | Dense Retr.+Transformer | ICT+BERT | 33.3 | 36.4 | 30.1 | 330m |
| Ours ($\mathcal{X}$ = Wikipedia, $\mathcal{Z}$ = Wikipedia) | Dense Retr.+Transformer | REALM | 39.2 | 40.2 | **46.8** | 330m |
| Ours ($\mathcal{X}$ = CC-News, $\mathcal{Z}$ = Wikipedia) | Dense Retr.+Transformer | REALM | **40.4** | **40.7** | 42.9 | 330m |

## Ablation Experiments

*Table 2.* Ablation experiments on NQ's development set.

| Ablation | Exact Match | Zero-shot Retrieval Recall@5 |
|---|---|---|
| REALM | 38.2 | 38.5 |
| REALM retriever+Baseline encoder | 37.4 | 38.5 |
| Baseline retriever+REALM encoder | 35.3 | 13.9 |
| Baseline (ORQA) | 31.3 | 13.9 |
| REALM with random uniform masks | 32.3 | 24.2 |
| REALM with random span masks | 35.3 | 26.1 |
| 30× stale MIPS | 28.7 | 15.1 |

### Encoder or Retriever

To determine whether REALM pre-training improves the retriever or the encoder, or both, reset the encoder or retriever to baseline.

### Masking scheme

During REALM pre-training, we want to focus on examples x that require world knowledge to predict the masked tokens. (i.e. salient spans such as “United Kingdom” or “July 1969”.)

## Results

Table 3 shows an example of the REALM masked language model prediction.

In this example, “Fermat” is the correct word, and REALM (row (c)) gives the word a much high probability compared to the BERT model (row (a)).

This shows that REALM is able to retrieve document to fill in the masked word even though it is trained with unsupervised text only.

*Table 3.* An example where REALM utilizes retrieved documents to better predict masked tokens. It assigns much higher probability (0.129) to the correct term, “Fermat”, compared to BERT. (Note that the blank corresponds to 3 BERT wordpieces.)

| | | | |
|---|---|---|---|
| $x$: | An equilateral triangle is easily constructed using a straightedge and compass, because 3 is a ___ prime. | | |
| (a) BERT | $p(y = \text{“Fermat”} \mid x)$ | $= 1.1 \times 10^{-14}$ | (No retrieval.) |
| (b) REALM | $p(y = \text{“Fermat”} \mid x, z)$ | $= 1.0$ | (Conditional probability with document $z$ =“257 is … a Fermat prime. Thus a regular polygon with 257 sides is constructible with compass … ”) |
| (c) REALM | $p(y = \text{“Fermat”} \mid x)$ | $= 0.129$ | (Marginal probability, marginalizing over top 8 retrieved documents.) |

## Summary

- The effectiveness of REALM is demonstrated by fine-tuning on the Open-QA task
- REALM could be connected to a broader set of ideas beyond Open-QA
- REALM could be generalized to:
  (1) structured knowledge: learn the decision of which entities are informative
  (2) multi-lingual setting: retrieving knowledge in a high-resource language to better represent text in a low-resource language
  (3) multi-modal setting: retrieving images or videos that can provide knowledge rarely observed in text

The Grainger College of Engineering

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN