# Reading Wikipedia to Answer Open-Domain Questions

Authors - Danqi Chen

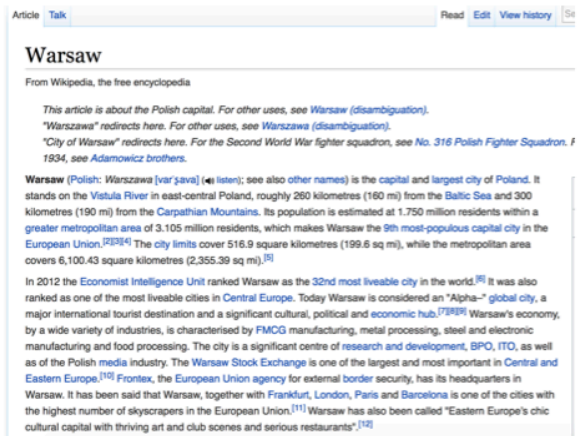
# Open-domain QA

SQuAD, TREC, WebQuestions, WikiMovies

Q: How many of Warsaw's inhabitants spoke Polish in 1933?

WIKIPEDIA
The Free Encyclopedia

**Document Retriever**



Article Talk Read Edit View history

Warsaw

From Wikipedia, the free encyclopedia

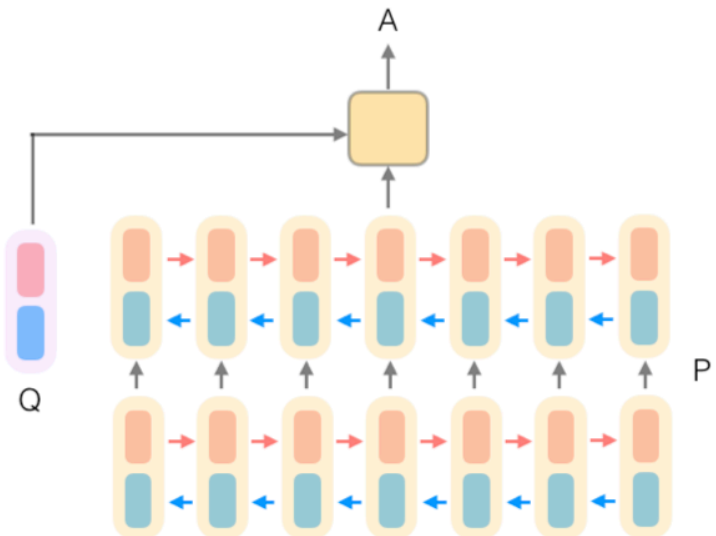
*This article is about the Polish capital. For other uses, see Warsaw (disambiguation).*
*"Warszawa" redirects here. For other uses, see Warszawa (disambiguation).*
*"City of Warsaw" redirects here. For the Second World War fighter squadron, see No. 316 Polish Fighter Squadron.*
*1934, see Adamowicz brothers.*

**Warsaw** (Polish: *Warszawa* [varˈʂava] (listen); see also other names) is the capital and largest city of Poland. It stands on the Vistula River in east-central Poland, roughly 260 kilometres (160 mi) from the Baltic Sea and 300 kilometres (190 mi) from the Carpathian Mountains. Its population is estimated at 1.750 million residents within a greater metropolitan area of 3.105 million residents, which makes Warsaw the 9th most-populous capital city in the European Union.[2][3][4] The city limits cover 516.9 square kilometres (199.6 sq mi), while the metropolitan area covers 6,100.43 square kilometres (2,355.39 sq mi).[5]

In 2012 the Economist Intelligence Unit ranked Warsaw as the 32nd most liveable city in the world.[6] It was also ranked as one of the most liveable cities in Central Europe. Today Warsaw is considered an "Alpha–" global city, a major international tourist destination and a significant cultural, political and economic hub.[7][8][9] Warsaw's economy, by a wide variety of industries, is characterised by FMCG manufacturing, metal processing, steel and electronic manufacturing and food processing. The city is a significant centre of research and development, BPO, ITO, as well as of the Polish media industry. The Warsaw Stock Exchange is one of the largest and most important in Central and Eastern Europe.[10] Frontex, the European Union agency for external border security, has its headquarters in Warsaw. It has been said that Warsaw, together with Frankfurt, London, Paris and Barcelona is one of the cities with the highest number of skyscrapers in the European Union.[11] Warsaw has also been called "Eastern Europe's chic cultural capital with thriving art and club scenes and serious restaurants".[12]

**Document Reader**

833,500



# Introduction

- Answering factoid questions in an open-domain setting
- Using Wikipedia as the unique knowledge source

# Document Retriever

- Articles and questions are compared as TF-IDF weighted bag-of-word vectors. Additionally uses bigram counts for retrieval.
- Return 5 Wikipedia articles given any question

# Document Reader

- A question q of l tokens , a paragraph p of m tokens
- Paragraph encoding
- Question encoding
- Prediction

# Paragraph encoding

$$\{\mathbf{p}_1, \ldots, \mathbf{p}_m\} = \text{RNN}(\{\tilde{\mathbf{p}}_1, \ldots, \tilde{\mathbf{p}}_m\}), \quad \tilde{\mathbf{p}}_i \in \mathbb{R}^d$$

$f_{emb}(p_i) = E(p_i)$ 300-dimensional

$f_{exact\_match}(p_i) = \mathbb{I}(p_i \in q)$ 3-dimensional original, lowercase or lemma form

$f_{token}(p_i) = (POS(p_i), NER(p_i), TF(p_i))$ term frequency (TF)

$$f_{align}(p_i) = \sum_j a_{i,j} E(q_j) \qquad a_{i,j} = \frac{exp(\alpha(E(p_i)) \cdot \alpha(E(q_j)))}{\sum_{j'} exp(\alpha(E(p_i)) \cdot \alpha(E(q_{j'})))}$$

$\alpha(\cdot)$ is single dense layer with ReLU nonlinearity. $a_{i,j}$ captures the similarity between. $p_i$ and each question words $q_j$

# Question encoding

- Recurrent layer on top of word embeddings of questions words.

- Attention.

# Prediction

- predict the two ends of the span that is most likely the correct answer
- input : paragraph vectors {p1, . . . , pm} and the question vector **q**
- two classifiers

$$P_{start}(i) \propto \exp(\mathbf{p}_i \mathbf{W}_s \mathbf{q})$$
$$P_{end}(i) \propto \exp(\mathbf{p}_i \mathbf{W}_e \mathbf{q})$$

the best span from token to token $max P_{start}(i) \times P_{end}(i') \quad i \leq i' \leq i + 15$

| Dataset | Example | Article / Paragraph |
|---|---|---|
| SQuAD | **Q**: How many provinces did the Ottoman empire contain in the 17th century?<br>**A**: 32 | **Article**: Ottoman Empire<br>**Paragraph**: ... At the beginning of the 17th century the empire contained 32 provinces and numerous vassal states. Some of these were later absorbed into the Ottoman Empire, while others were granted various types of autonomy during the course of centuries. |
| CuratedTREC | **Q**: What U.S. state's motto is "Live free or Die"?<br>**A**: New Hampshire | **Article**: Live Free or Die<br>**Paragraph**: "Live Free or Die" is the official motto of the U.S. state of New Hampshire, adopted by the state in 1945. It is possibly the best-known of all state mottos, partly because it conveys an assertive independence historically found in American political philosophy and partly because of its contrast to the milder sentiments found in other state mottos. |
| WebQuestions | **Q**: What part of the atom did Chadwick discover?†<br>**A**: neutron | **Article**: Atom<br>**Paragraph**: ... The atomic mass of these isotopes varied by integer amounts, called the whole number rule. The explanation for these different isotopes awaited the discovery of the neutron, an uncharged particle with a mass similar to the proton, by the physicist James Chadwick in 1932. ... |
| WikiMovies | **Q**: Who wrote the film Gigli?<br>**A**: Martin Brest | **Article**: Gigli<br>**Paragraph**: Gigli is a 2003 American romantic comedy film written and directed by Martin Brest and starring Ben Affleck, Jennifer Lopez, Justin Bartha, Al Pacino, Christopher Walken, and Lainie Kazan. |

Wikipedia as knowledge source, curated Trec , webquestions and wiki movies doesn't contain Training paragraphs, so distant supervision is used to create training data.

| Dataset | Train | | Test |
|---|---|---|---|
| | Plain | DS | |
| SQuAD | 87,599 | 71,231 | 10,570† |
| CuratedTREC | 1,486* | 3,464 | 694 |
| WebQuestions | 3,778* | 4,602 | 2,032 |
| WikiMovies | 96,185* | 36,301 | 9,952 |

| Dataset | Wiki Search | Doc. Retriever | |
|---|---|---|---|
| | | plain | +bigrams |
| SQuAD | 62.7 | 76.1 | **77.8** |
| CuratedTREC | 81.0 | 85.2 | **86.0** |
| WebQuestions | 73.7 | **75.5** | 74.4 |
| WikiMovies | 61.7 | 54.4 | **70.3** |

| Method | Dev | | Test | |
|---|---|---|---|---|
| | EM | F1 | EM | F1 |
| Dynamic Coattention Networks (Xiong et al., 2016) | 65.4 | 75.6 | 66.2 | 75.9 |
| Multi-Perspective Matching (Wang et al., 2016)† | 66.1 | 75.8 | 65.5 | 75.1 |
| BiDAF (Seo et al., 2016) | 67.7 | 77.3 | 68.0 | 77.3 |
| R-net† | n/a | n/a | 71.3 | 79.7 |
| DrQA (Our model, Document Reader Only) | **69.5** | **78.8** | 70.0 | 79.0 |

| Dataset | YodaQA | DrQA | | |
|---|---|---|---|---|
| | | SQuAD | +Fine-tune (DS) | +Multitask (DS) |
| SQuAD *(All Wikipedia)* | n/a | 27.1 | 28.4 | 29.8 |
| CuratedTREC | 31.3 | 19.7 | 25.7 | 25.4 |
| WebQuestions | 39.8 | 11.8 | 19.5 | 20.7 |
| WikiMovies | n/a | 24.5 | 34.3 | 36.5 |

# REALM: Retrieval-Augmented Language Model Pre-Training

Kelvin Guu *  Kenton Lee *

# Motivation

- Pre-trained model in like BERT and T5 contains large amount of world knowledge implicitly in their network parameters.
- Larger models for storing more world knowledge.
- Capture knowledge in more interpretable and modular way

# Background

- Language model pre-training - Bert (Masked LM)
- Open domain question - answering
  - Retrieve top k document and predict the answer from them.

# Approach

- For both pre-training and fine-training, REALM learns- P(ylx)
  - For pre-training , x is masked sentence , y is missing token
  - For fine-tuning , task -OpenQA, x is question- y is answer

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

- Z- helpful documents

# Knowledge Retriever

- Learn distribution of documents again question.

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

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

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

$Z_{title}$ - document's title
$Z_{body}$ - document's body

# Knowledge Augmented Encoder

- Given input x, retrieved document z.
  - KAE defines p(ylz,x)
  - x and z are joined into single sentence and feed into transformer.
- Different architectures for pre-training and fine-tuning.

# Pre-training

- Masked Language model
  - Predict original value of missing token in x.

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

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

  - Jx is the total number of [MASK] tokens in x,
  - W are learnable parameters.

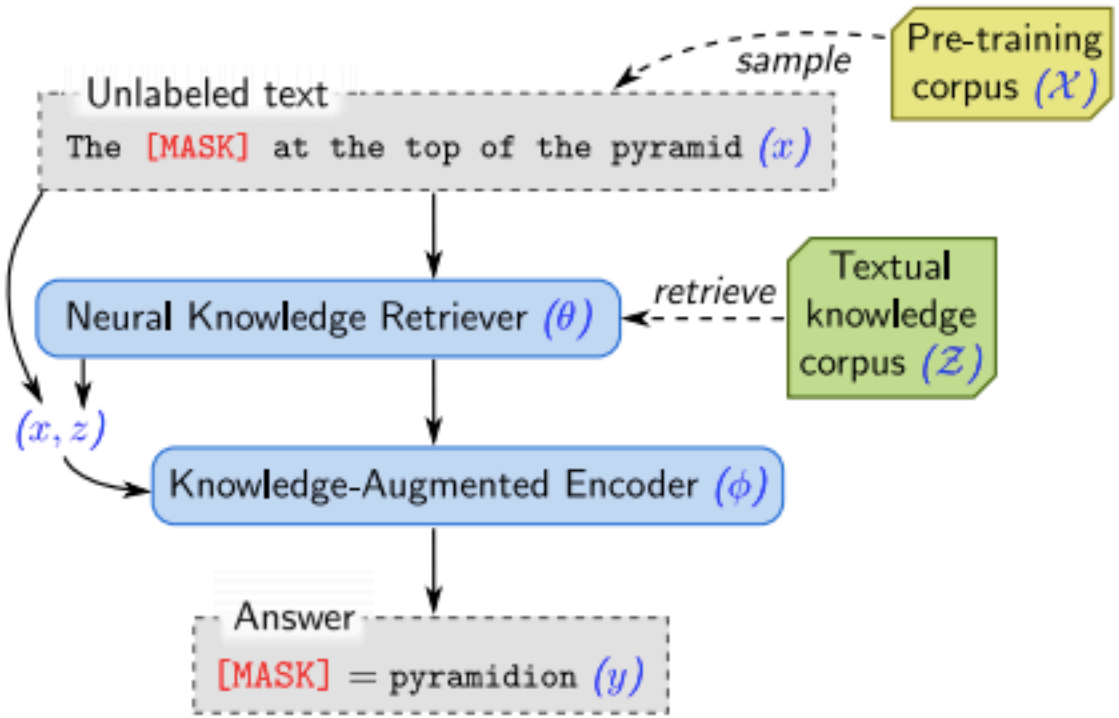
# Fine - tuning

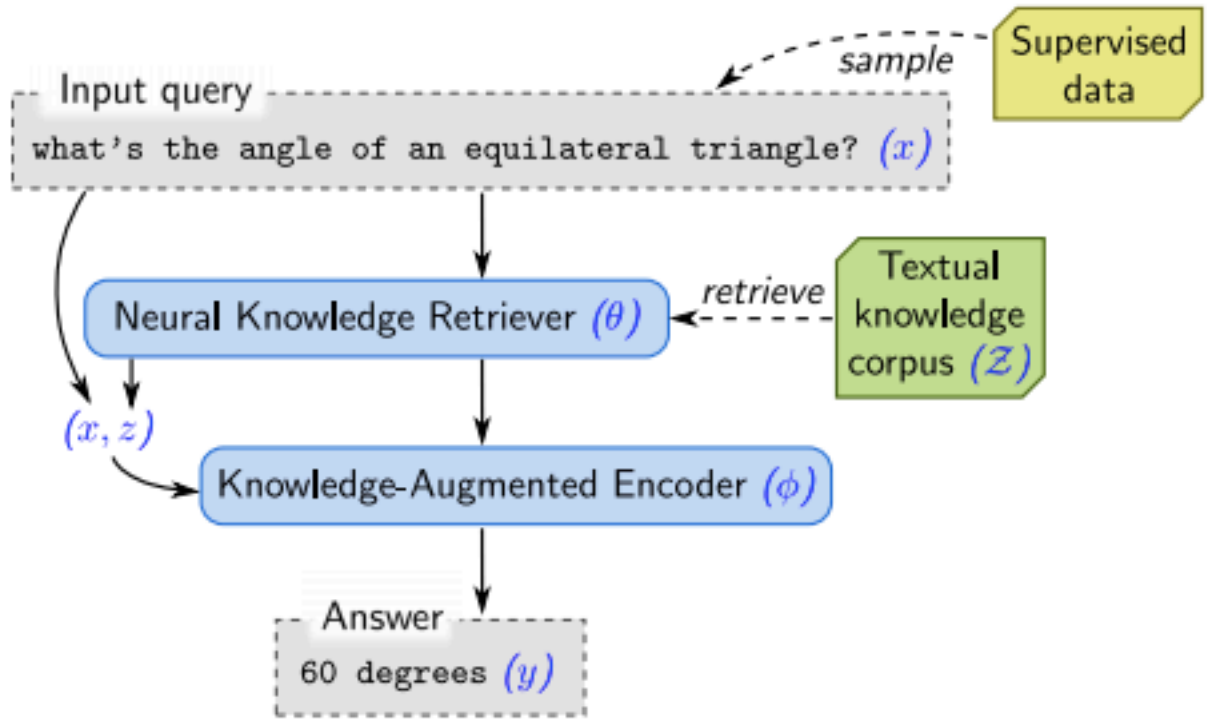
- Task - OpenQA
  - y  is answer string.
  - Assumption - y is contagious sequence of tokens in z
  - S(z, y) be the set of spans matching y in z.

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

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

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


Pre-training corpus $(\mathcal{X})$
sample
Unlabeled text
The [MASK] at the top of the pyramid $(x)$
Neural Knowledge Retriever $(\theta)$
retrieve
Textual knowledge corpus $(\mathcal{Z})$
$(x, z)$
Knowledge-Augmented Encoder $(\phi)$
Answer
[MASK] = pyramidion $(y)$


Supervised data
sample
Input query
what's the angle of an equilateral triangle? $(x)$
Neural Knowledge Retriever $(\theta)$
retrieve
Textual knowledge corpus $(\mathcal{Z})$
$(x, z)$
Knowledge-Augmented Encoder $(\phi)$
Answer
60 degrees $(y)$

# Training

- By maximizing the log-likelihood log p(ylx) of the correct output wrt to model parameters.
- Key challenge-
  - Marginal probability - $p(y \mid x) = \sum_{z \in Z} p(y \mid x, z)\, p(z \mid x)$
  - Involves summation over all documents in knowledge source.
  - Approximate this by selecting top k under highest p(zlx).
  - Reasonable since most of documents will have 0 probability.

# Training

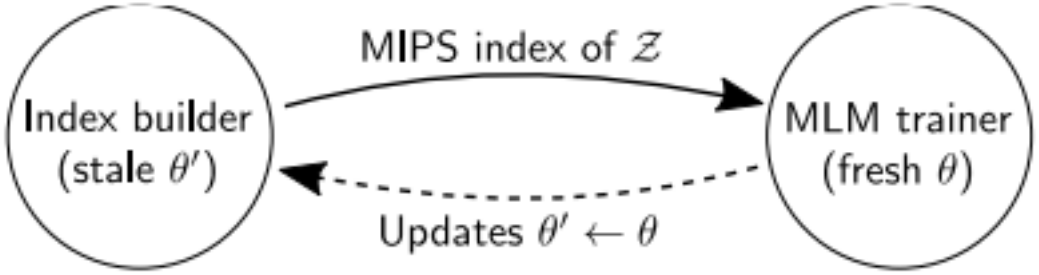
- p(zlx) is equal to f(x,z).

$$f(x, z) = \texttt{Embed}_{\texttt{input}}(x)^{\intercal} \texttt{Embed}_{\texttt{doc}}(z)$$

  - Employ maximum inner product search to find approx top k documents.
  - Need to precompute $Embed_{doc}(z)$ for every document.
  - Construct an efficient search index.

# Training

- But this index will become stale after update in parameters.
  - It only used to compute top-k documents.
  - Assuming no drastic change in parameters, index will slightly be stale.
  - Update the index asynchronously and train the MLM model.



# Training

- MIPS index is refreshed after every few hundred training epochs for pre-training.
- Fine-tuning: index is built once and parameters of $Embed_{doc}(z)$ are not re-trained. $Embed_{input}$ is still fine-tuned to update retrieval function from query side.

# What does retriever learn?

- Gradient of knowledge retriever wrt to parameters -

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

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

- p(ylz, x) - probability of predicting the correct output y given z.
- p(ylx) - is the expected value of p(ylx,z).

# Training strategies -

- Salient span masking
  - Some tokens only requires only local context.
  - Mask tokens which requires world knowledge.
    - “United Kingdom” or “July 1969”.
  - Identify such entities using NER and dates to mask them during pre training.

# Training strategies

- Null document
  - Add empty document at top of k retrieved document.
  - This allows for cases where no-retrieval is necessary.
- Prohibiting trivial retrievals during pre-training -
  - If pre-training corpus and knowledge source are sames,
  - KAE can trivially predict y by looking at unmasked version of x in z ( which contains x).
  - This might result in KAE looking for string matches of x.
  - Remove such documents z during training.

# Training strategies

- Initialization -
  - If not initialised,
  - Retriever doesn’t retrieve relevant documents.
  - KAE starts ignoring documents by retriever.
  - Retriever will not receive any meaningful gradients.
  - Retriever can’t improve
  - Vicious cycle.

# Training strategies

- Initialization
  - Train the retriever using inverse cloze task.
  - Given a sentence, figure out from which document it came from.
  - Warm-start KAE using pre-trained BERT.

# Experiments

- Open QA datasets -
    - Focus on datasets where authors didn’t know the answer.
    - Avoid issues when questions is formulated with answer in mind.
    - Natural questions-Open(NQ) - google queries and their answers.
    - WebQuestions ( WQ) - google suggest api and their answer from amazon mechanical turk.
    - Curated Trec (CT)- collection of question answer pair from sites like MSNSearch and AskJeeves.

# Experiments

- Approaches compared -
  - Retrieval based OpenQA  - like DrQA
  - Generation based OpenQA -
    - Text-to-text, encode question and predict answer token by token.
    - fine-tuned T5 for openQA
- Pre-training-
  - 200k steps on 64 TPUs, batch size 512, lr  3e-5 and Bert's default optimizer.
- For each candidate , retrieve 8 candidate using MIPS including null document.

# Results

| 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 |

# Results

- REALM outperforms T5 when approx 30 times lower in size.
- T5 has access to Squad data during pre-training.

| 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 |

# Reviews (Pros)

- Thorough comparisons, experiments, training strategy,(Atishya, Jigyasa,Rajas, Lovish,Vipul)
- Dot product to retrieve documents, this allow for use of MIPS(Soumya,
- Improve SOTA(Soumya, Rajas,Saransh,Makkunda)
- Pre-training in retrieval phase(keshav)
- Provide context to language model(pawan)
- Explainability (Saransh, Siddhant,Pratyush)
- Ability to adapt to new knowledge(Siddhant)
- Greener alternative to T5(Vipul)
- Modular approach(Pratyush,Vipul)

# Reviews (Cons)

- Lot of hyper-parameters (Atishya)
- Answer to be continuous span of keywords (Atishya,Siddhant,saransh
- Doesn’t allow multi-hop reasoning(Soumya,Rajas,Jigyasa,Siddhant,saransh
- Conflicting information during retrieval due to time( Rajas )
- Oversell their paper(Keshav)
- Pre-training before pre-training(Lovish)
- Not actually explainable(Pawan)
- Started with issues with Bert and used BERT in the end(Pawan,Makkunda)
- Document embedding is fixed but input embedding is allowed to train during fine-tuning resulting these embeddings might go into different spaces.(Vipul)

# Reviews(Extension)

- Using attention, copy mechanism to copy certain entities from retrieved documents - not vocab dependent and no need of answer span to be continous ( Atishya,siddhant)
    - How will you define P(y/z,x) (Lovish)
- Retrieve Subgraph of big KB to augment sentence generation(Atishya)
    - Combining text is better then graphs, graph2text? (Soumya)
- Concat top k retrieved document to allow for multi-hop answering.(Soumya)
    - Extend current SOTA for multi-hop answering with current paper(Keshav)
    - May exceed Bert's capacity(Rajas)
        - Extract top N sentences/paragraph instead of documents(Saransh)
        - Multiple - retrieve-and-rank framework, in second retrieve only select from top documents selected in 1st step ( Pratyush)
        - Retrieval Multiple times for multi-hop answering. Append the answer of 1st hop to retrieve relevant documents for 2nd hop(Makkunda)

# Reviews(Extensions)

- Separate pre-training and fine-tuning to make system actually modular(Soumya)
  - Use openIE triplets, construct a graph and then use of GNNs to predict missing nodes for pre-training. Similarly GNN can operate on retrieved graph for fine-tuning(keshav)
    - Use of GNNs is moving away from the focus which is knowledge learning by adapting pre-training in language models. How do we incorporate multi-hop answering in pre-training(Saransh)
    - We should focus on building end - to -end pipelines for Graphs similar to current task.(Vipul)
- Instead of using Bert like architecture for retrieve/rank, how to extract knowledge from its pre-trained parameters (Pratyush)

# Reviews(Extensions)

- Add time component to documents/questions to counter conflicting answers after updating knowledge source(Rajas)
- Multiple hstart/hend over multiple documents for multi-hop answering(Jigyasa)

Thanks !!!