Knowledge Bases

NLP

Baymurzina Dilyara,
materials are by Dmitry Evseev, DeepPavlov

Telegram

Github of NLP Course

Feedback

Don’t hesitate to contact us!

AIRI | Университет Сириус

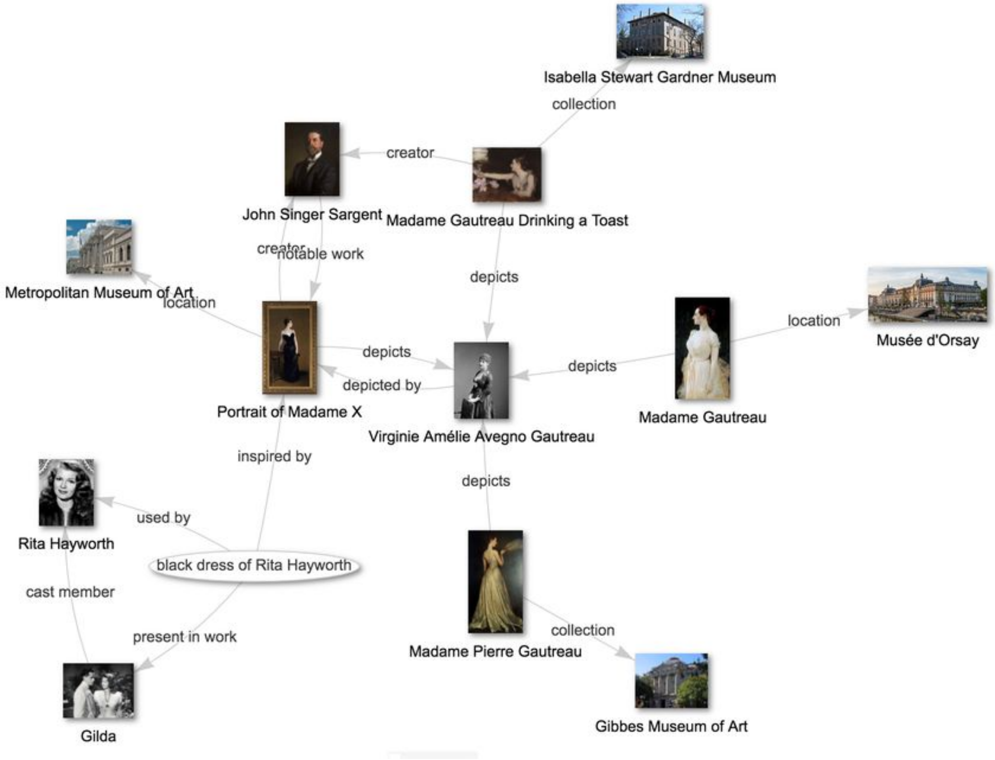
Agenda

01 Entity Linking

02 TextQA

03 KBQA

Knowledge Base (KB)



Leonardo da Vinci

From Wikipedia, the free encyclopedia

"Da Vinci" redirects here. For other uses, see Da Vinci (disambiguation) and Leonardo da Vinci (disambiguation).

In this Renaissance Florentine name, the name da Vinci *is an indicator of birthplace, not a family name; the person is properly referred to by the given name,* Leonardo.

Leonardo di ser Piero da Vinci[b] (15 April 1452 – 2 May 1519) was an Italian polymath of the High Renaissance who was active as a painter, draughtsman, engineer, scientist, theorist, sculptor, and architect.[3] While his fame initially rested on his achievements as a painter, he also became known for his notebooks, in which he made drawings and notes on a variety of subjects, including anatomy, astronomy, botany, cartography, painting, and paleontology. Leonardo is widely regarded to have been a genius who epitomized the Renaissance humanist ideal,[4] and his collective works compose a contribution to later generations of artists matched only by that of his younger contemporary, Michelangelo.[3][4]

Born out of wedlock to a successful notary and a lower-class woman in,

Leonardo da Vinci

This portrait attributed to Francesco Melzi, c. 1515–1518, is the only certain contemporary depiction of Leonardo.[1][2]

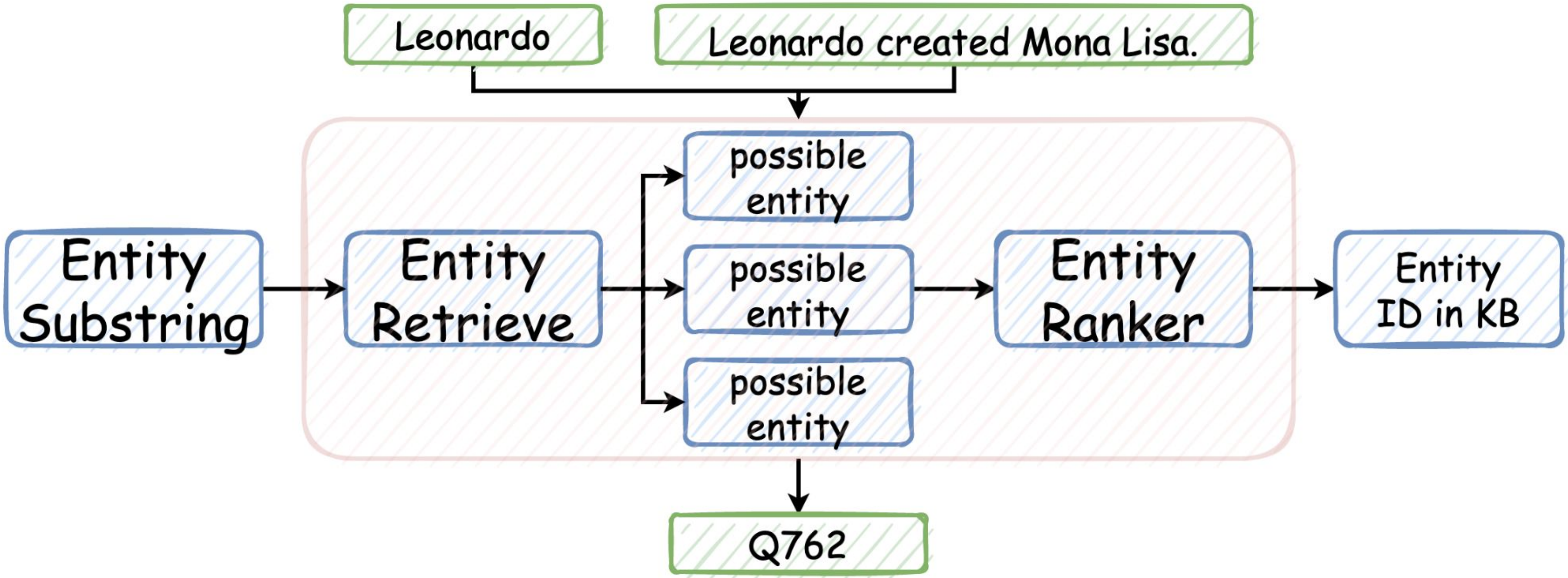
Born	Leonardo di ser Piero da Vinci 15 April 1452 (Anchiano?)[a] Vinci, Republic of Florence
Died	2 May 1519 (aged 67) Clos Lucé, Amboise, Kingdom of France

Main Tasks

- **Entity Linking** for each extracted entity determines identifier from the Knowledge Base (KB)
- **Question Answering**
- **Response Generation** based on Knowledge Bases
  - Triplets from KB can be used to fill templates
  - Triplets and knowledge paragraphs can be used as as additional input for NN generation
- **Fact Checking**

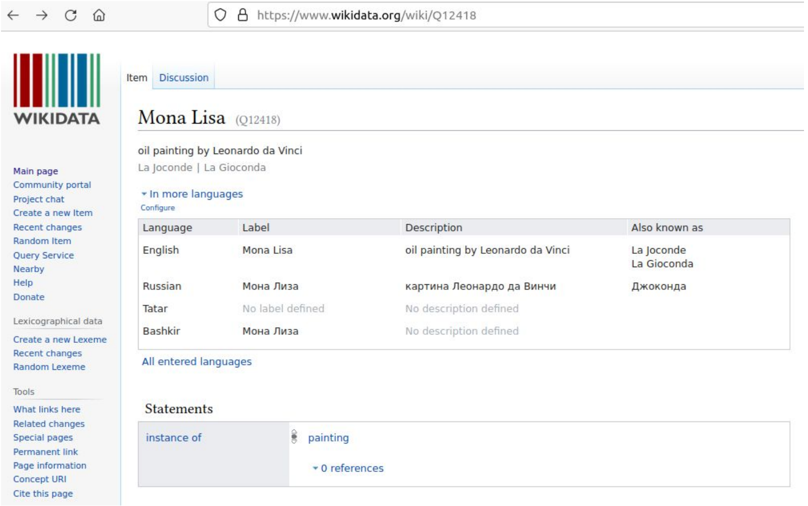
EL:
Entity Linking

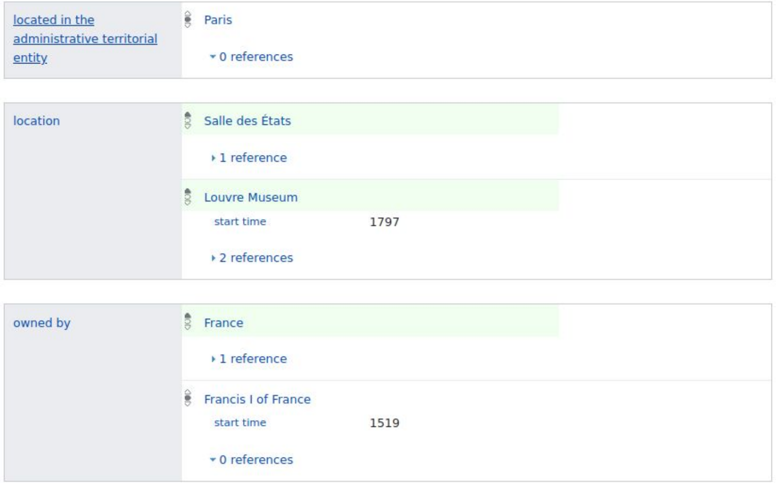
Entity Linking



Entity Linking Example

“The Mona Lisa is a sixteenth century oil painting created by ***Leonardo****. It's held at the Louvre in Paris.”*





Modular Entity Linking

*“The Mona Lisa is a sixteenth century oil painting created by **Leonardo**. It's held at the Louvre in Paris.”*

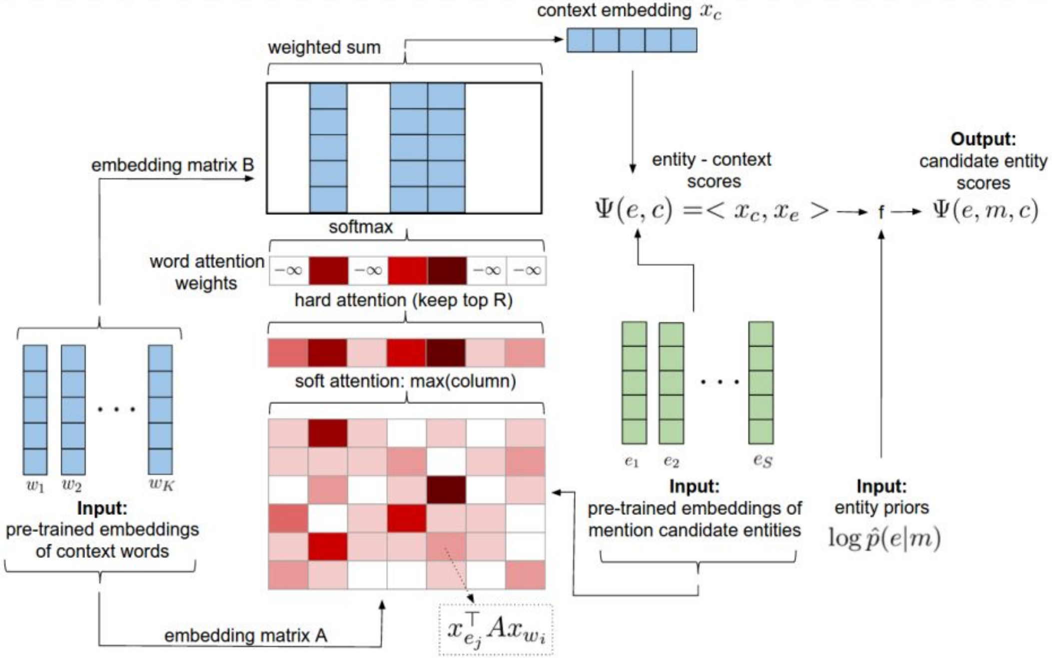
1. Retrieve from KB possible entities (substring or embedding similarity)
   a. Leonardo DiCaprio (Q38111), Leonardo da Silva Souza (Q14091212), Leonardo da Vinci (Q762), ...
2. Entity Disambiguation to determine which entity is the most appropriate.
   a. Example of possible entities with probabilities: 0.96 Leonardo da Vinci; 0.41 Leonardo DiCaprio; 0.35 Leonardo da Silva Souza.

Modular EL examples:

BLINK, REL, ExtEnD, OpenTapioca, Deeptype, etc.

Deep Joint Entity Disambiguation with Local Neural Attention

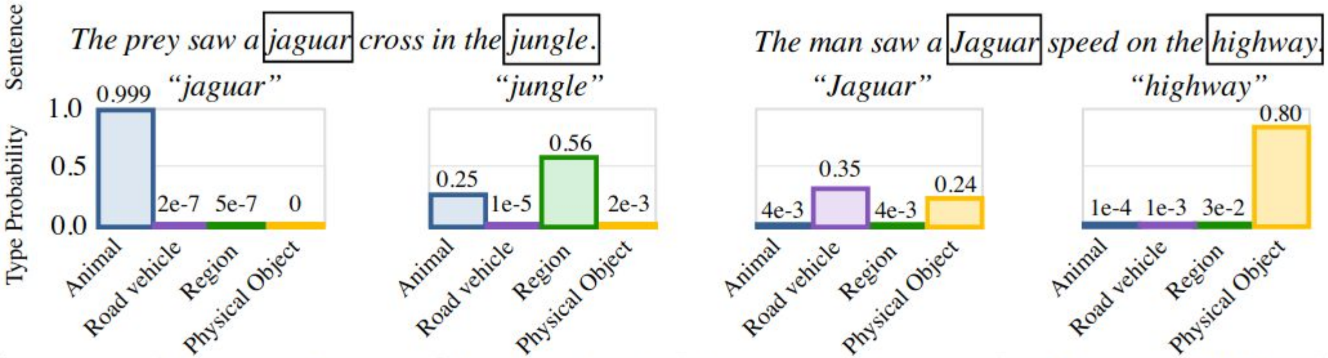
Scalar product of the context's and possible entities embeddings.



Ganea, O. E., & Hofmann, T. (2017). Deep joint entity disambiguation with local neural attention.

DeepType

Utilizes entity type to rank entities



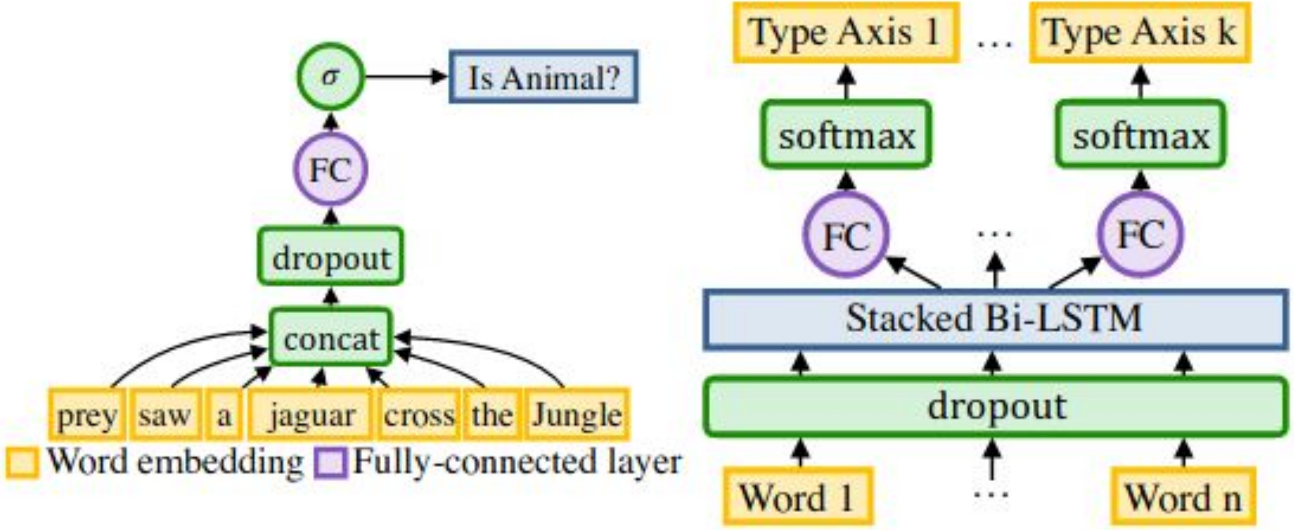
Entity	**jaguar**	**Jaguar**	**jungle**	**jungle**	**jaguar**	**Jaguar**	**highway**	**Highway**
Type	Animal	Road vehicle	Region	Music	Animal	Road vehicle	Physical Object	Film
only link Prob.	0.29	**0.60**	**0.35**	0.17	0.29	**0.60**	**0.85**	0.04
Prob. w/. types	**1.0**	0.0	**1.0**	0.0	0.0	**1.0**	**1.0**	0.0

Figure 1: Example model output: "jaguar" refers to different entities depending on context. Predicting the type associated with each word (e.g. animal, region, etc.) helps eliminate options that do not match, and recover the true entity. Bar charts give the system's belief over the type-axis "`IsA`", and the table shows how types affects entity probabilities given by Wikipedia links.

https://github.com/openai/deeptype

DeepType

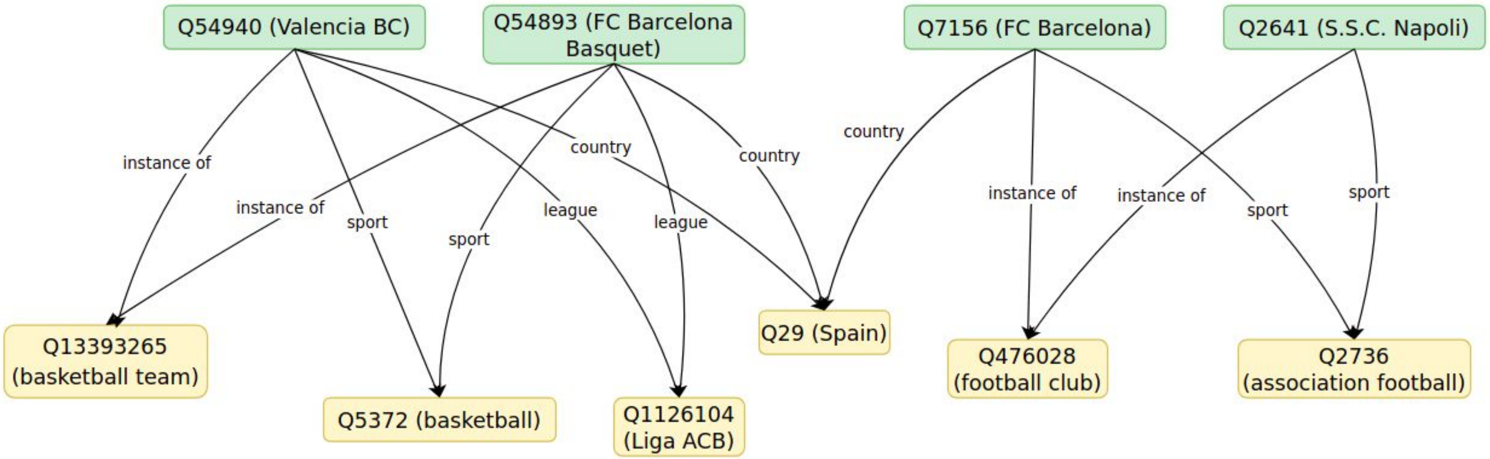
Entity Type classification



https://github.com/openai/deeptype

DPEX, TagME, OpenTapioca

Utilize relations from the Knowledge Graph (KG) for entity disambiguation



Ferragina, P., & Scaiella, U. (2011). Fast and accurate annotation of short texts with wikipedia pages. IEEE software, 29(1), 70-75.
Delpeuch, A. (2019). Opentapioca: Lightweight entity linking for wikidata. arXiv preprint arXiv:1904.09131.

BLINK

BLINK steps:

- entity retrieval from Faiss-index based on closest BERT embedding
- entity disambiguation:
  - Input: `[CLS] The Mona Lisa is a sixteenth century oil painting created by [START_ENT] Leonardo [END_ENT] . [SEP] Italian Renaissance polymath (1452-1519) [SEP]`
  - If entity fits context, assign 1, otherwise 0.

https://github.com/facebookresearch/BLINK

ExtEnD

Figure 1: Illustration of ExtEnD on the example sentence *After a long fight Superman saved Metropolis*. The model takes as input a sentence with the target mention to disambiguate, *Metropolis*, explicitly marked (for better visualization, we resort here to highlighting with a different color rather than surrounding it with special tokens) along with the text representation of each candidate. As in our experiments, the knowledge base here is Wikipedia and the candidate text representations are Wikipedia page titles. Then, the model performs the disambiguation by indicating the start and end token of the span containing the predicted entity representation.

https://github.com/SapienzaNLP/extend

End-to-end Entity Linking

- End-to-end EL implies that search for possible entities and their ranking is performed by the same component.

End-to-end EL examples:
- [GENRE]
- [Kolitsas et al., 2018]

GENRE

BART-based architecture

Two modes:

- wikipage title generation for the given entity
  Einstein was a [START_ENT] German [END_ENT] physicist. → Germany
- text generation where for each entity substring, entity origin is generated
  Einstein was a German physicist. → [Einstein] (Albert Einstein) was a [German] (Germany) physicist.

https://github.com/facebookresearch/GENRE

Entity Linking Summary

- Scalar product of the context and entities embeddings.
- Entity Types can be used to find possible entities.
- Entity Descriptions can be used to get entity embeddings.
- Relations from KG can be used for entity disambiguation.
- Entity titles can be generated end-to-end with BART.
- EL can be performed as a span detection of the entity title in the joint string of context and entity title candidates.

Model	RAM, Gb	GPU, Gb	WNED, micro F1	Support of custom KB
ExtEnD	4.5	2.5	88.8	No
GENRE	9.7	2.8	87.4	No
DPEX	1.9	1.4	68.2	Yes
BLINK	37.5	1.1	75.5	Yes
REL	2.0	0.95	41.4	No
OpenTapioca	4.4	0	26.8	Yes

TextQA:
Text Question Answering

Question Answering Sources

- Knowledge Graphs
  - Wikidata, DBPedia, etc.
- Text Knowledge Bases
  - Wikipedia, etc.
- Joint Knowledge Bases

Text-based Question Answering Task

***Task**: answer to the given question based on the knowledge paragraph.*

SQuAD - Stanford Question Answering Dataset 100k+ question-answer pairs on 500+ articles.

Passage: Tesla later approached Morgan to ask for more funds to build a more powerful transmitter. **When asked where all the money had gone, Tesla responded by saying that he was affected by the Panic of 1901**, which he (Morgan) had caused. Morgan was shocked by the reminder of his part in the stock market crash and by Tesla's breach of contract by asking for more funds. Tesla wrote another plea to Morgan, but it was also fruitless. Morgan still owed Tesla money on the original agreement, and Tesla had been facing foreclosure even before construction of the tower began.

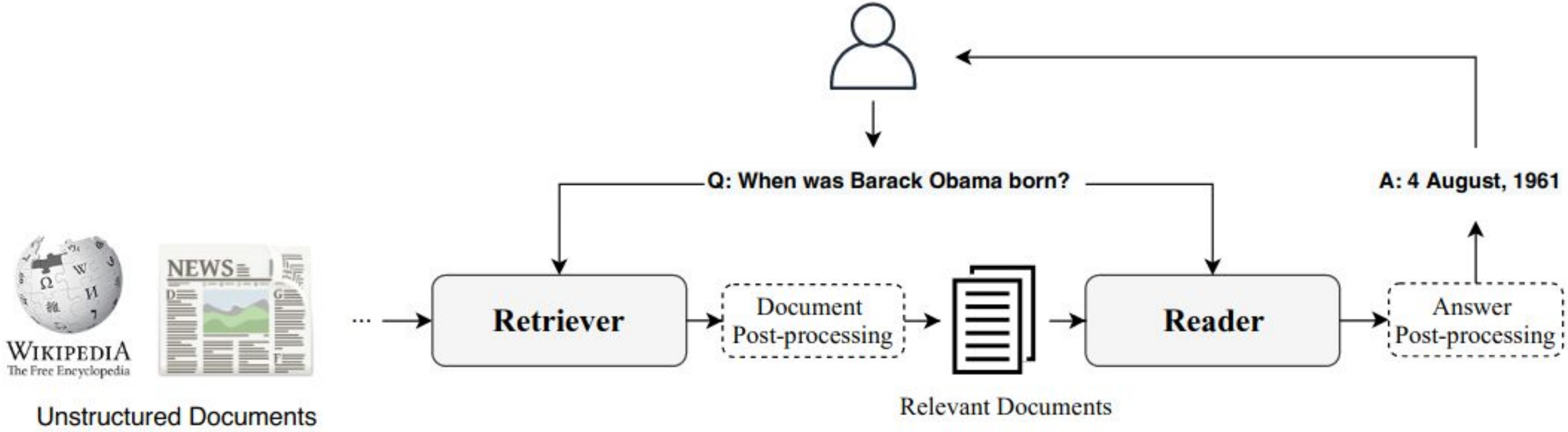
Question: On what did Tesla blame for the loss of the initial money?

Answer: Panic of 1901

https://rajpurkar.github.io/SQuAD-explorer/

Questions Answering Components

- Retrieve the text paragraphs containing response to the question.
- Retrieve the responses from the paragraphs, determination of the most relevant response.



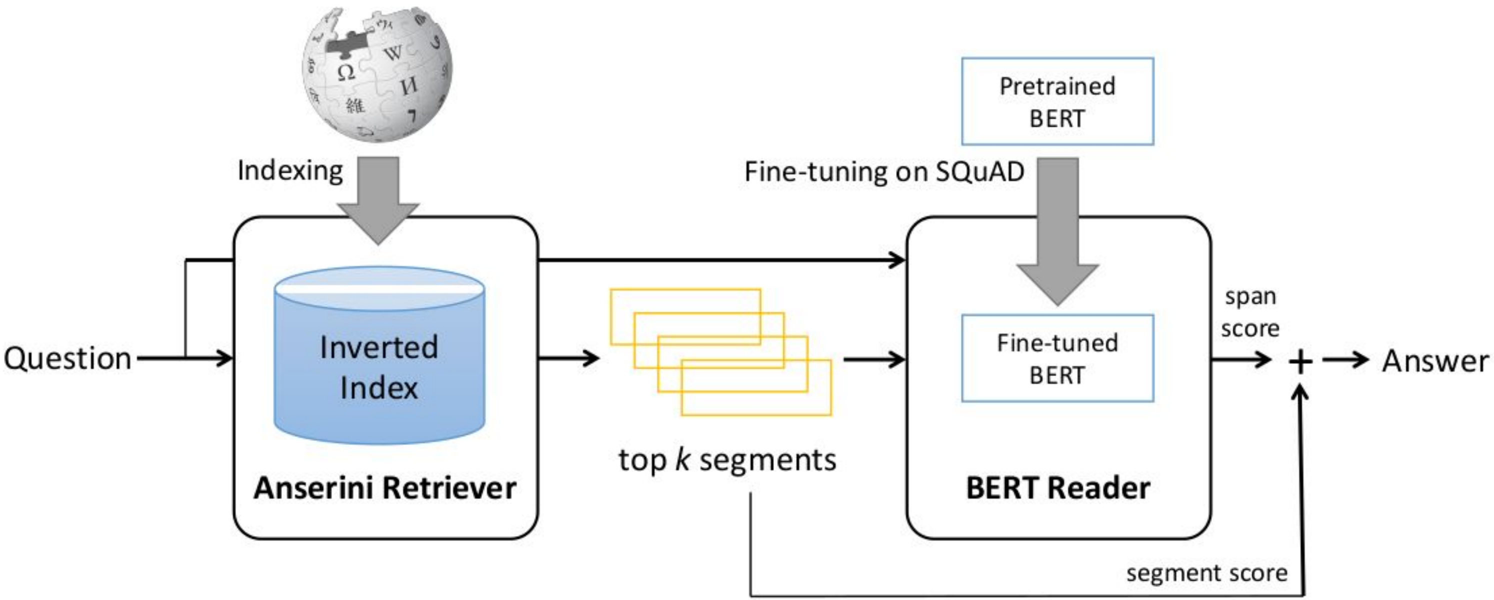
Tf-idf Retrieve

- Wikipedia Pages are vectorized as follows:
  - Extract n-grams from text page
  - Compute hashes of the extracted n-grams
  - Compose a sparse matrix where a raw corresponds to pages, columns to hashes, elements to tf-idf of the n-grams.
- The question is vectorized in the same way.
- Compute scalar products of the question’s and pages’ vectors.

Example system: DrQA

Chen, D., Fisch, A., Weston, J., & Bordes, A. (2017). Reading wikipedia to answer open-domain questions. arXiv preprint arXiv:1704.00051.

Anserini Library for Sparse Retrieve



Yang, W., Xie, Y., Lin, A., Li, X., Tan, L., Xiong, K., ... & Lin, J. (2019). End-to-end open-domain question answering with bertserini. arXiv preprint arXiv:1902.01718.

Dense Retrieve

- Separate encoders for question and paragraphs embeddings.
- The model is trained to maximize scalar product of the embeddings, if the paragraph contains the response to the question, and minimize otherwise.

Top-k passages	Original DPR NQ model	New DPR model
1	45.87	52.47
5	68.14	72.24
20	79.97	81.33
100	85.87	87.29

Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., ... & Yih, W. T. (2020). Dense passage retrieval for open-domain question answering. arXiv preprint arXiv:2004.04906.

Retrieve Methods Comparison

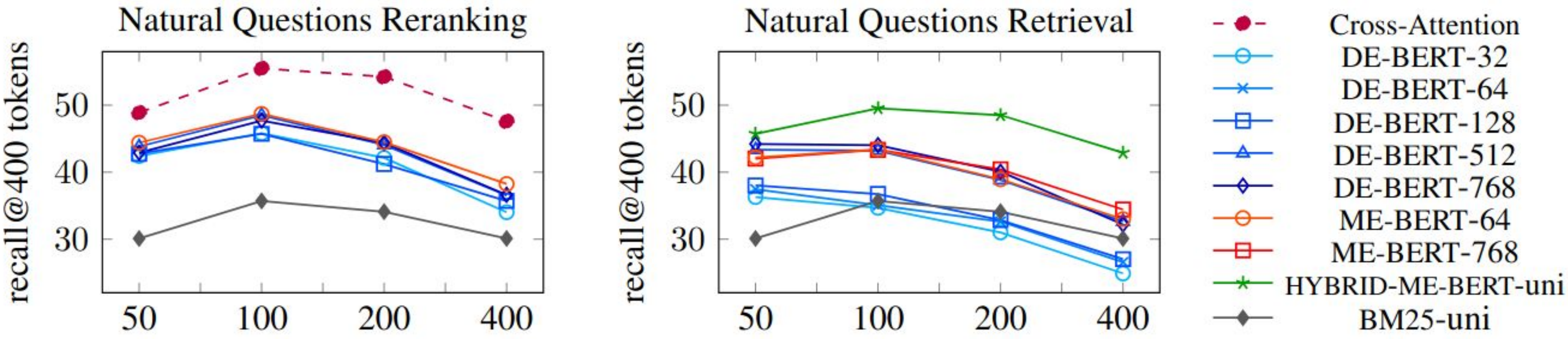


Figure 5: Results on NQ passage recall as maximum passage length varies (50 to 400 tokens). *Left*: Reranking of 200 passages; *Right*: Open domain retrieval result on all of (English) Wikipedia. Exact numbers refer to Table 3.

Luan, Y., Eisenstein, J., Toutanova, K., & Collins, M. (2020). Sparse, dense, and attentional representations for text retrieval. arXiv preprint arXiv:2005.00181.

Binary Passage Retrieval

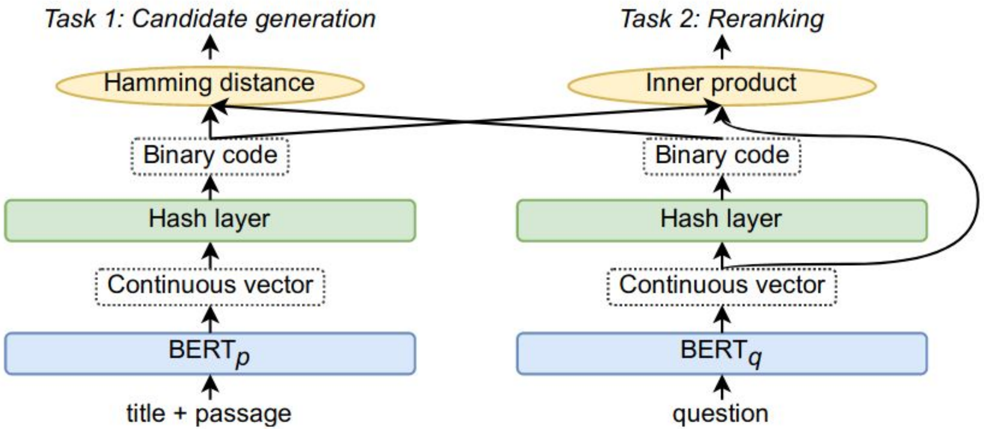
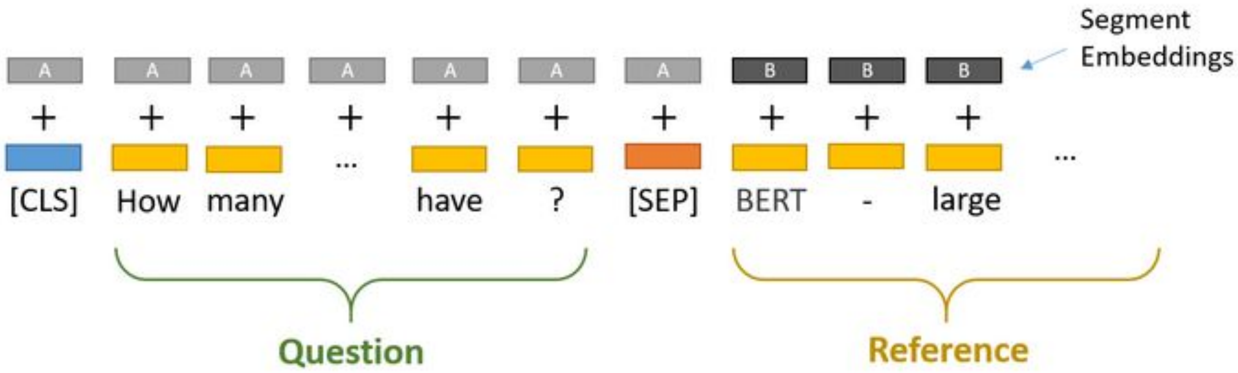


Figure 1: Architecture of BPR, a BERT-based model generating compact binary codes for questions and passages. The passages are retrieved in two stages: (1) efficient candidate generation based on the Hamming distance using the binary code of the question and (2) accurate reranking based on the inner product using the continuous embedding of the question.

Yamada, I., Asai, A., & Hajishirzi, H. (2021). Efficient passage retrieval with hashing for open-domain question answering. arXiv preprint arXiv:2106.00882.

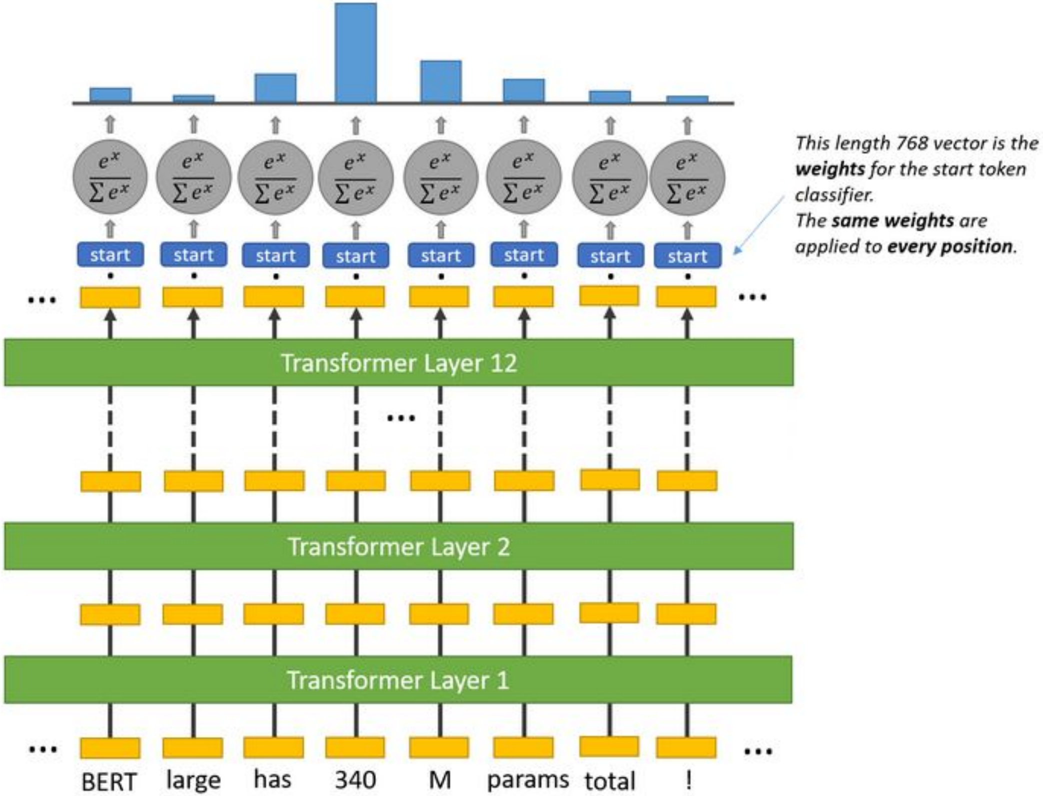
BERT for Paragraph-based QA

- 

Question: How many parameters does BERT-large have?

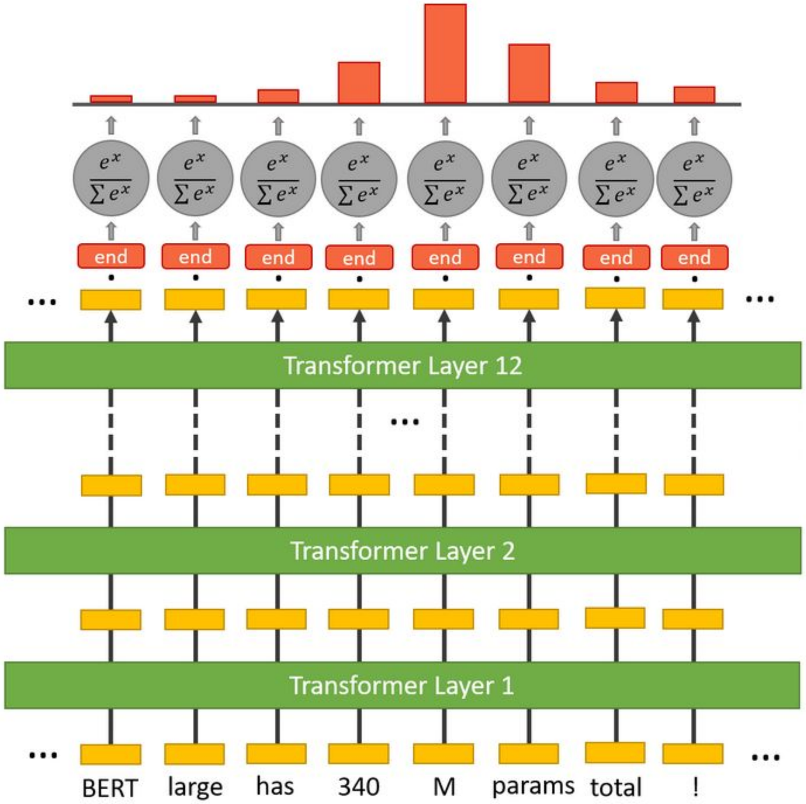
Reference Text: BERT-large is really big... it has 24 layers and an embedding size of 1,024, for a total of 340M parameters! Altogether it is 1.34GB, so expect it to take a couple minutes to download to your Colab instance.

BERT for Paragraph-based QA. The First Token

- 

BERT for Paragraph-based QA. The Last Token

-



Many Paragraphs to Find the Aswer

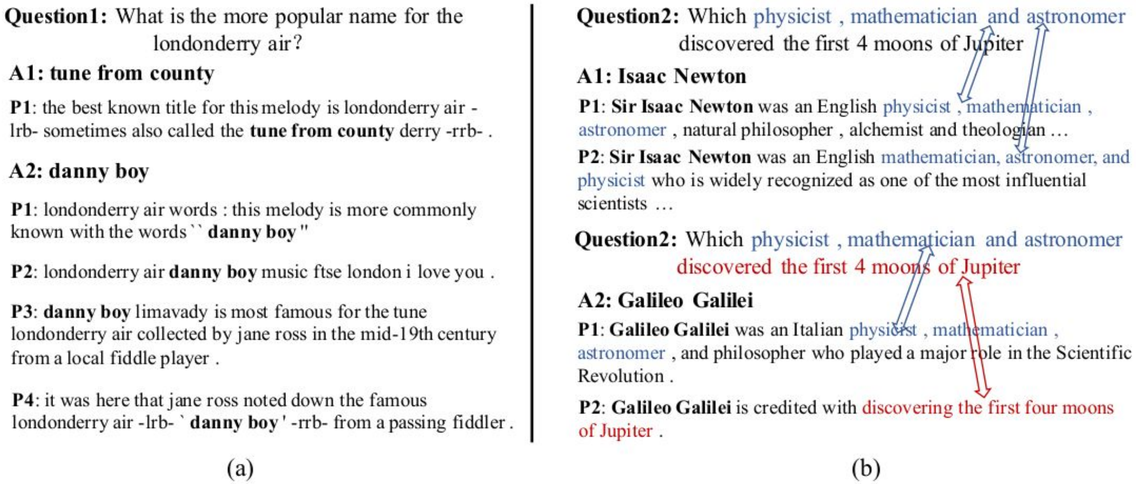


Figure 1: Two examples of questions and candidate answers. (a) A question benefiting from the repetition of evidence. Correct answer A2 has multiple passages that could support A2 as answer. The wrong answer A1 has only a single supporting passage. (b) A question benefiting from the union of multiple pieces of evidence to support the answer. The correct answer A2 has evidence passages that can match both the first half and the second half of the question. The wrong answer A1 has evidence passages covering only the first half.

Wang, S., Yu, M., Jiang, J., Zhang, W., Guo, X., Chang, S., ... & Campbell, M. (2017). Evidence aggregation for answer re-ranking in open-domain question answering. arXiv preprint arXiv:1711.05116.

Many Paragraphs to Find the Aswer

Reasoning Type	%	Example(s)
Inferring the ***bridge entity*** to complete the 2nd-hop question (Type I)	42	**Paragraph A:** The 2015 Diamond Head Classic was a college basketball tournament ... ***Buddy Hield*** *was named the tournament's MVP*. **Paragraph B:** ***Chavano Rainier "Buddy" Hield*** is a Bahamian professional basketball player for the **Sacramento Kings** of the NBA... **Q:** Which team does the player named 2015 Diamond Head Classic's MVP play for?
Comparing two entities (Comparison)	27	**Paragraph A:** LostAlone were a British rock band ... consisted of *Steven Battelle, Alan Williamson, and Mark Gibson*... **Paragraph B:** Guster is an American alternative rock band ... Founding members *Adam Gardner, Ryan Miller, and Brian Rosenworcel* began... **Q:** Did LostAlone and Guster have the same number of members? (**yes**)
Locating the **answer entity** by checking multiple properties (Type II)	15	**Paragraph A:** Several *current and former members of the Pittsburgh Pirates* – ... John Milner, **Dave Parker**, and Rod Scurry... **Paragraph B:** **David Gene Parker**, *nicknamed "The Cobra"*, is an American former player in Major League Baseball... **Q:** Which former member of the Pittsburgh Pirates was nicknamed "The Cobra"?
Inferring about the property of an entity in question through a ***bridge entity*** (Type III)	6	**Paragraph A:** *Marine Tactical Air Command Squadron 28* is a United States Marine Corps aviation command and control unit based at ***Marine Corps Air Station Cherry Point***... **Paragraph B:** ***Marine Corps Air Station Cherry Point*** ... is a United States Marine Corps airfield located in **Havelock, North Carolina**, USA ... **Q:** What city is the Marine Air Control Group 28 located in?
Other types of reasoning that require more than two supporting facts (Other)	2	**Paragraph A:** ... the towns of Yodobashi, **Okubo, Totsuka, and Ochiai town** *were merged into Yodobashi ward*. ... ***Yodobashi Camera*** *is a store with its name taken from the town and ward*. **Paragraph B**: ***Yodobashi Camera*** Co., Ltd. is *a major Japanese retail chain specializing in electronics, PCs, cameras and photographic equipment*. **Q:** Aside from Yodobashi, what other towns were merged into the ward which gave the major Japanese retail chain specializing in electronics, PCs, cameras, and photographic equipment it's name?

Table 3: Types of multi-hop reasoning required to answer questions in the HOTPOTQA dev and test sets. We show in ***orange bold italics*** bridge entities if applicable, *blue italics* supporting facts from the paragraphs that connect directly to the question, and **green bold** the answer in the paragraph or following the question. The remaining 8% are single-hop (6%) or unanswerable questions (2%) by our judgement.

Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W. W., Salakhutdinov, R., & Manning, C. D. (2018). Hotpotqa: A dataset for diverse, explainable multi-hop question answering. arXiv preprint arXiv:1809.09600.

Retrieve of Several Paragraphs

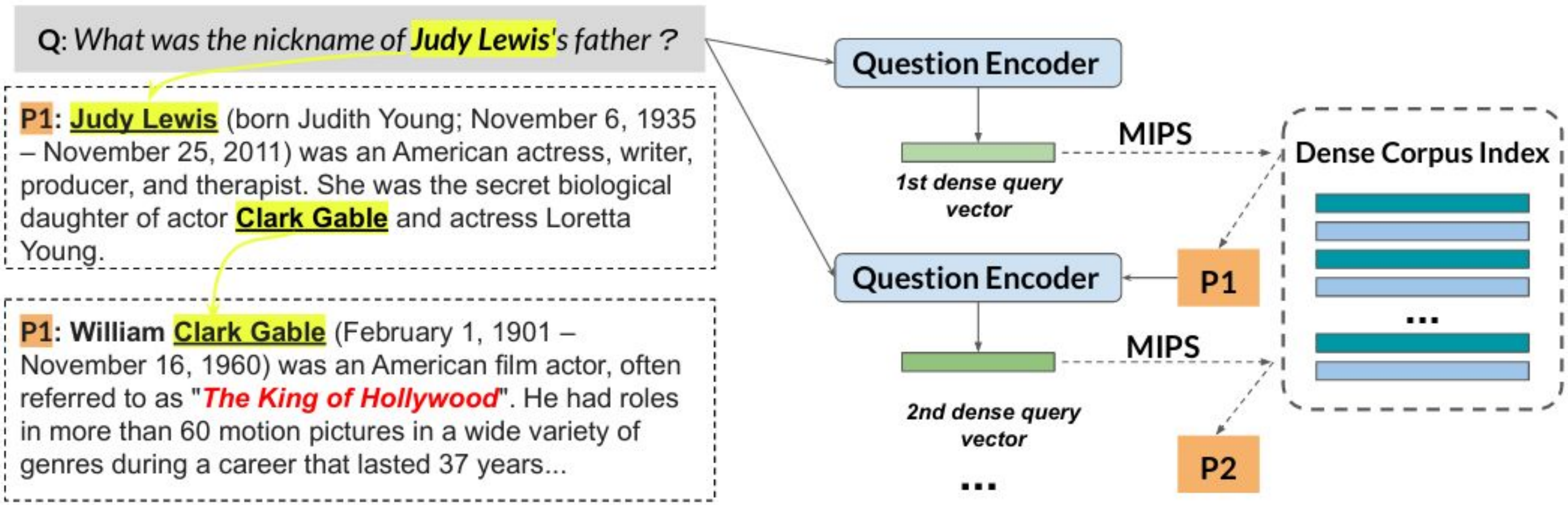


Figure 1: An overview of the multi-hop dense retrieval approach.

Xiong, W., Li, X. L., Iyer, S., Du, J., Lewis, P., Wang, W. Y., ... & Oğuz, B. (2020). Answering complex open-domain questions with multi-hop dense retrieval. arXiv preprint arXiv:2009.12756.

AISO (Adaptive Information Seeking)

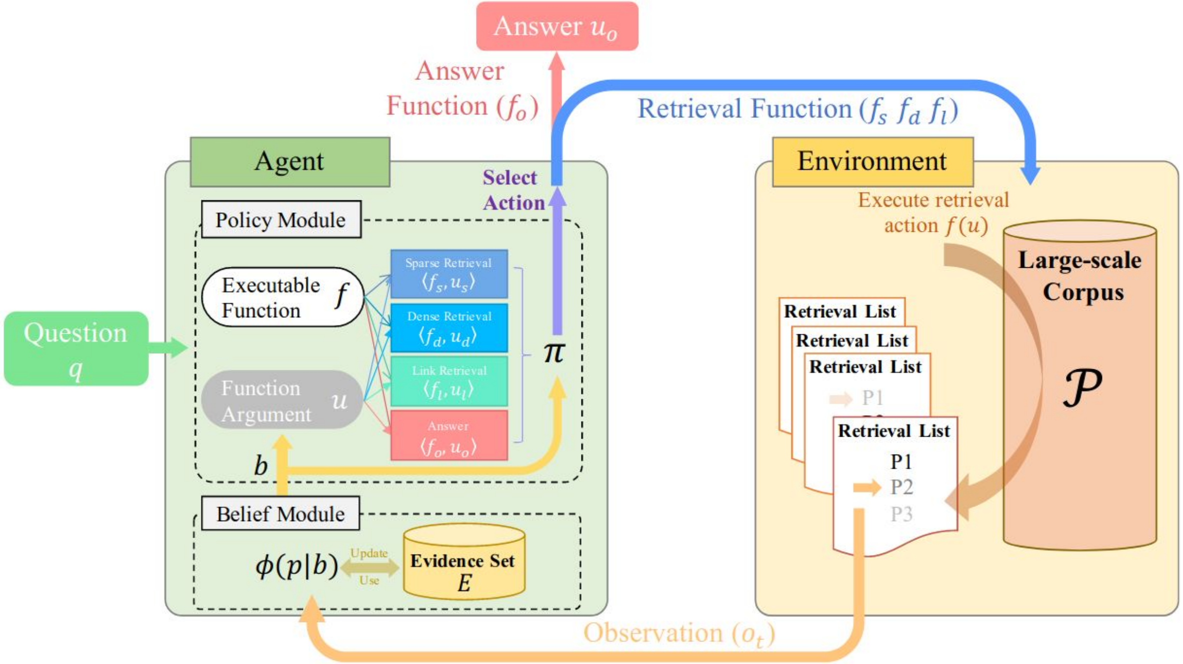
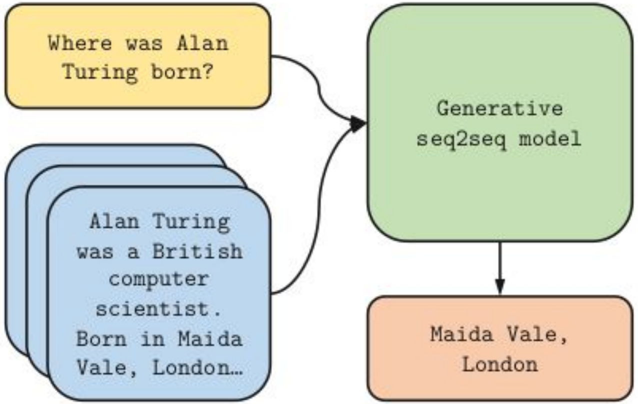


Figure 2: The overview of the AISO.

Zhu, Y., Pang, L., Lan, Y., Shen, H., & Cheng, X. (2021). Adaptive Information Seeking for Open-Domain Question Answering. arXiv preprint arXiv:2109.06747.

Fusion-in-Decoder

T5-based response generation



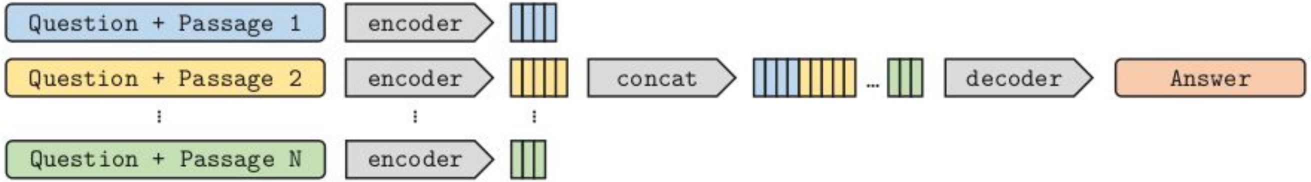


Figure 2: Architecture of the Fusion-in-Decoder method.

Izacard, G., & Grave, E. (2020). Leveraging passage retrieval with generative models for open domain question answering. arXiv preprint arXiv:2007.01282.

Closed-book Question Answering

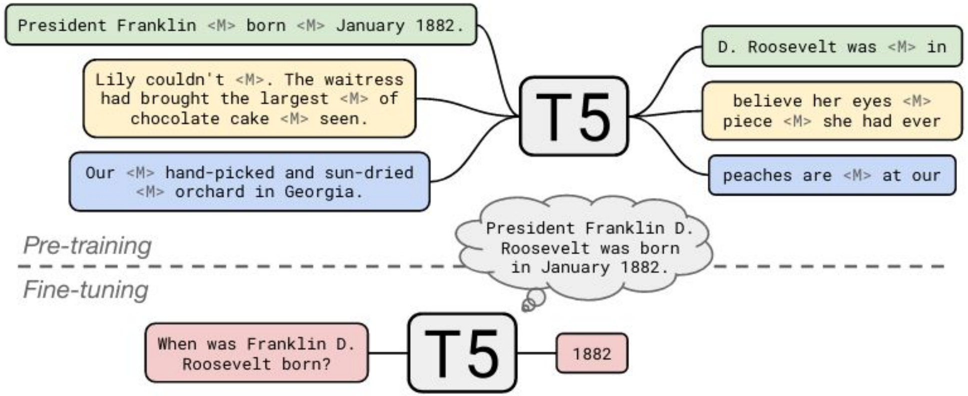


Figure 1: T5 is pre-trained to fill in dropped-out spans of text (denoted by <M>) from documents in a large, unstructured text corpus. We fine-tune T5 to answer questions without inputting any additional information or context. This forces T5 to answer questions based on "knowledge" that it internalized during pre-training.

Table 1: Scores achieved by fine-tuning T5 on the open-domain Natural Questions (NQ), WebQuestions (WQ), and TriviaQA (TQA) tasks.

	NQ	WQ	TQA dev	TQA test
Chen et al. (2017)	–	20.7	–	–
Lee et al. (2019)	33.3	36.4	47.1	–
Min et al. (2019a)	28.1	–	50.9	–
Min et al. (2019b)	31.8	31.6	55.4	–
Asai et al. (2019)	32.6	–	–	–
Ling et al. (2020)	–	–	35.7	–
Guu et al. (2020)	40.4	40.7	–	–
Févry et al. (2020)	–	–	43.2	53.4
Karpukhin et al. (2020)	**41.5**	42.4	**57.9**	–
T5-Base	25.9	27.9	23.8	29.1
T5-Large	28.5	30.6	28.7	35.9
T5-3B	30.4	33.6	35.1	43.4
T5-11B	32.6	37.2	42.3	50.1
T5-11B + SSM	34.8	40.8	51.0	60.5
T5.1.1-Base	25.7	28.2	24.2	30.6
T5.1.1-Large	27.3	29.5	28.5	37.2
T5.1.1-XL	29.5	32.4	36.0	45.1
T5.1.1-XXL	32.8	35.6	42.9	52.5
T5.1.1-XXL + SSM	35.2	**42.8**	51.9	**61.6**

Roberts, A., Raffel, C., & Shazeer, N. (2020). How Much Knowledge Can You Pack Into the Parameters of a Language Model?. arXiv preprint arXiv:2002.08910.

Text QA Summary

Retrieve:

- sparse
- dense
- hybrid (extracting top-N paragraphs in sparse-index, re-ranking top-N paragraphs with dense-index).

Answer extraction:

- answer span detection (extractive QA)
- answer generation (generative QA)

Closed-book QA:

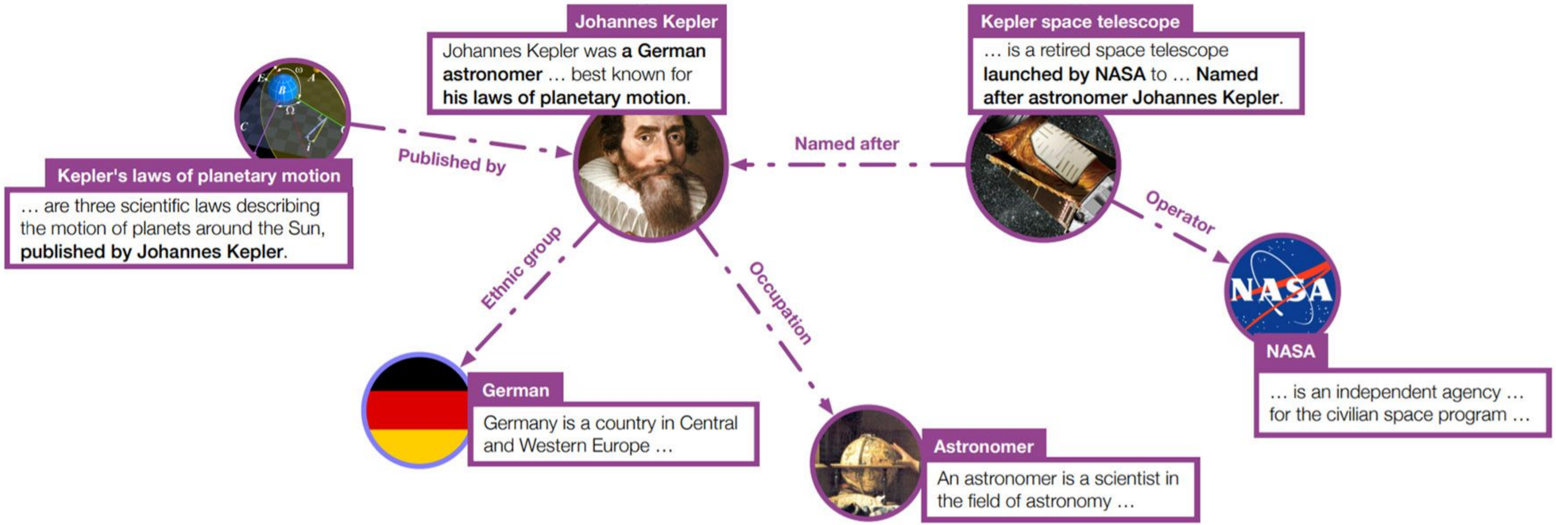
- generation with pre-trained T5 on Wikipedia

KBQA:
Knowledge Base Question Answering

Knowledge Base Question Answering

KBQA components:

- extraction of triplets that can contain the answer;
- finding the answer in the extracted triplets.



ерситет
оиус

SPARQL Requests for KBQA

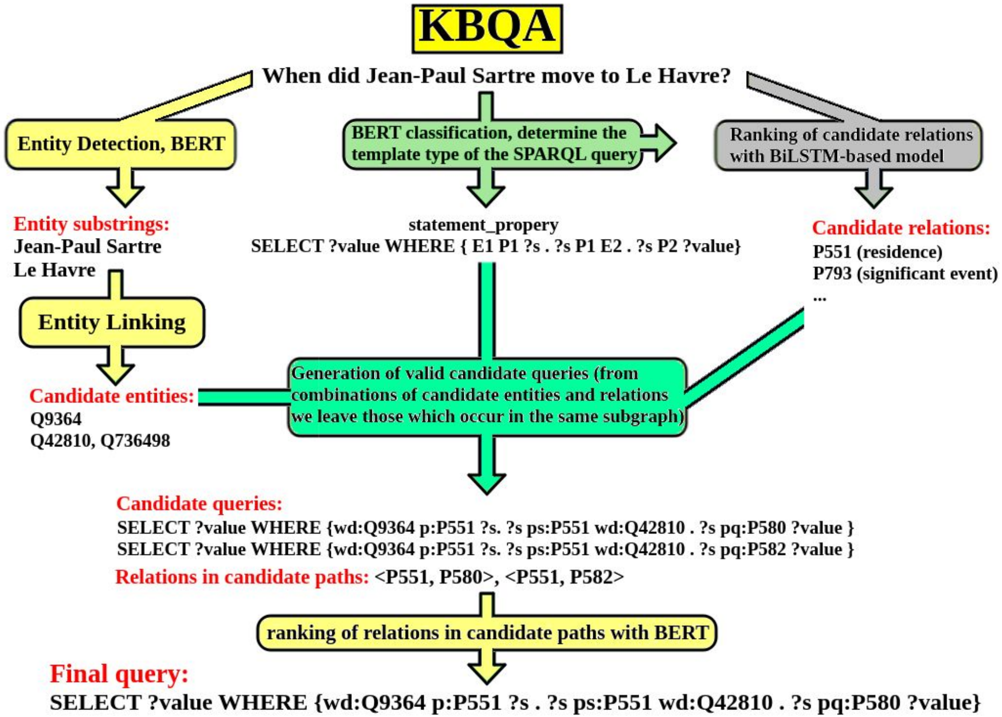
Some methods are based on SPARQL-requests response to which is the answer.

Steps:

- Determine a template for SPARQL-request
- Entity linking
- Determine relations between entities and the answer
- Fill the template of the SPARQL-request with entities and relations
- Make a request – get the answer

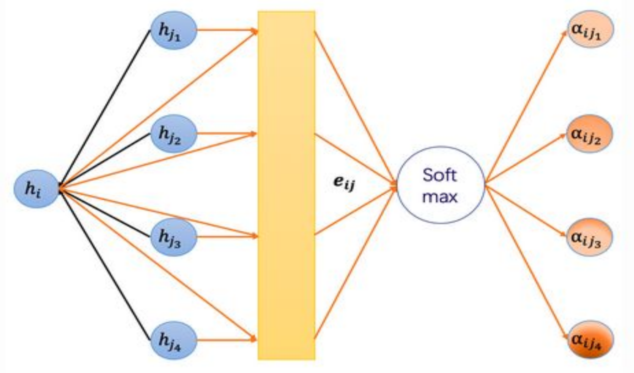
- 1-hop — SPARQL-запрос с одним триплетом
  Пример: «Кто написал роман "Зима тревоги нашей"?»
  SELECT ?answer WHERE wd:Q2063426wdt:P50 ?answer
- multi-constraint — SPARQL-запрос, где сущность-ответ входит в несколько триплетов
  Пример: «На какой реке стоит Минск?»
  SELECT ?answer WHERE wd:Q2280 wdt:P206 ?answer . ?answer wdt:P31 wd:Q4022
- qualifier-constraint — SPARQL-запрос содержит уточняющее отношение
  Пример: «Какой футболист в 2004 году получил "Золотой мяч"?»
  SELECT ?answer WHERE wd:Q166177 p:P1346 [ps:P1346 ?answer; pq:P585 "2004-01-01T00:00:00+00:00"8sd:dateTime] .
- count — в SPARQL-запросе используется для подсчета количества сущностей
  Пример: «Сколько существует законов Ньютона?»
  SELECT (COUNT(?laws) as ?answer) WHERE wd:Q38433 wdt:P527 ?laws
- ranking — сущности-ответы сортируются по убыванию или возрастанию определенного параметра с помощью оператора ORDER в SPARQL-запросе
  Пример: «В каком городе находится самая высокая в мире телебашня?»
  SELECT ?answer WHERE ?en p:P2048/psn:P2048/wikibase:quantityAmount ?h . ?en wdt:P31 wd:Q11166728 . ?en wdt:P131* ?answer . ?answer wdt:P31 wd:Q515 .

KBQA from DeepPavlov



Evseev, D. A., & Arkhipov, M. Y. (2020). Sparql query generation for complex question answering with bert and bilstm-based model. In Computational Linguistics and Intellectual Technologies (pp. 270-282).

LASAGNE (muLti-task semAntic parSing with trAnsformer and Graph atteNtion nEtworks)



$$z_i^{(l)} = W^{(l)} h_i^{(l)}, \tag{1}$$

$$e_{ij}^{(l)} = \text{LeakyReLU}(\vec{a}^{(l)^T}(z_i^{(l)} \| z_j^{(l)})), \tag{2}$$

$$\alpha_{ij}^{(l)} = \frac{\exp(e_{ij}^{(l)})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik}^{(l)})}, \tag{3}$$

$$h_i^{(l+1)} = \sigma\left(\sum_{j \in \mathcal{N}(i)} \alpha_{ij}^{(l)} z_j^{(l)}\right), \tag{4}$$

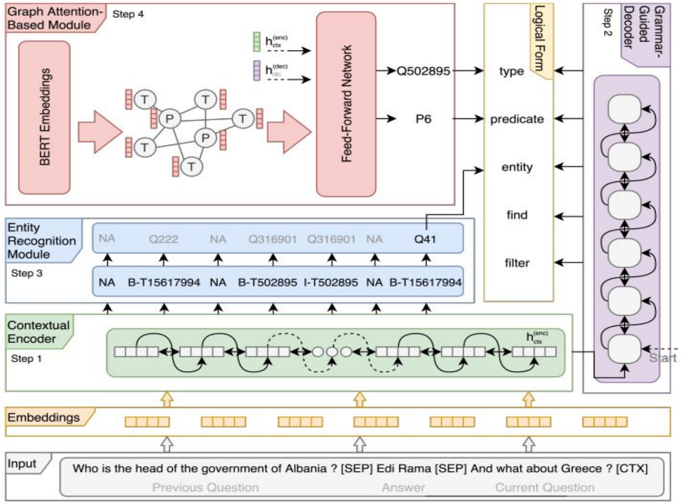
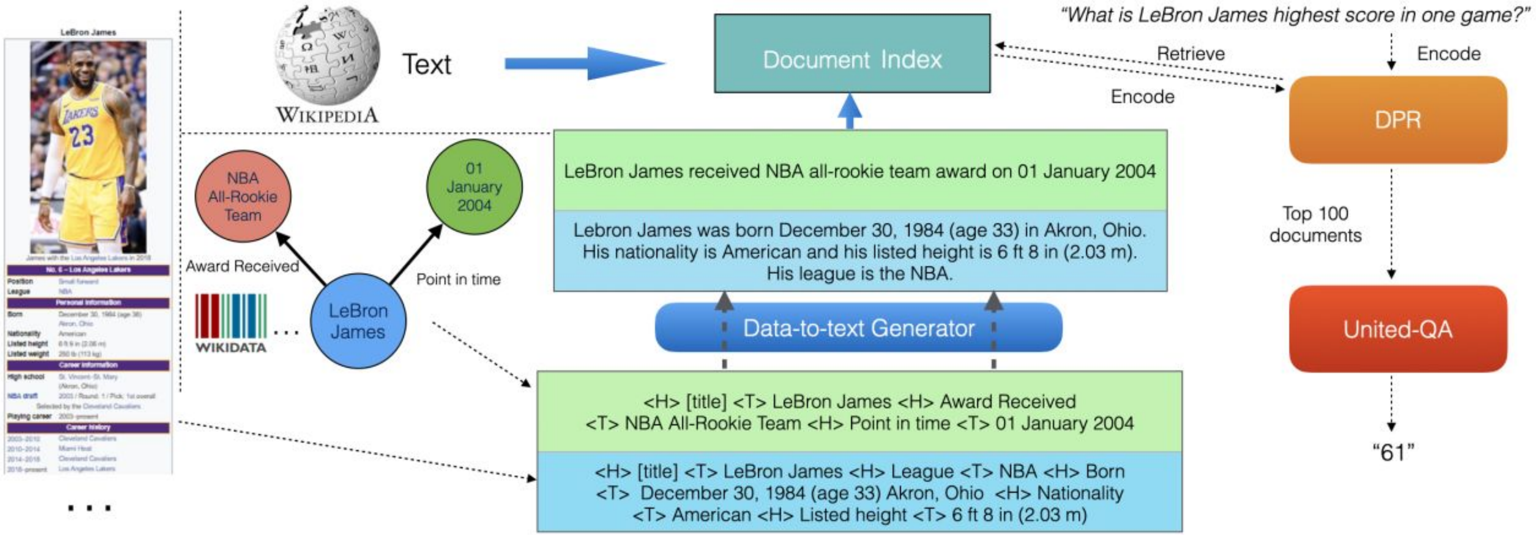


Figure 2: LASAGNE (Multi-task Semantic Parsing with Transformer and Graph Attention Networks) architecture. It consists of three modules: 1) A semantic parsing-based transformer model, containing a contextual encoder and a grammar guided decoder using the grammar defined in Table 1. 2) An entity recognition module, which identifies all the entities in the context, together with their types, linking them to the knowledge graph. It filters them based on the context and permutes them, in case of more than one required entity. Finally, 3) a graph attention-based module that uses a GAT network initialised with BERT embeddings to incorporate and exploit correlations between (entity) types and predicates. The resulting node embeddings, together with the context hidden state (h_{ctx}) and decoder hidden state (d_h), are used to score the nodes and predict the corresponding type and predicate.

Kacupaj, E., Plepi, J., Singh, K., Thakkar, H., Lehmann, J., & Maleshkova, M. (2021). Conversational question answering over knowledge graphs with transformer and graph attention networks. arXiv preprint arXiv:2104.01569.

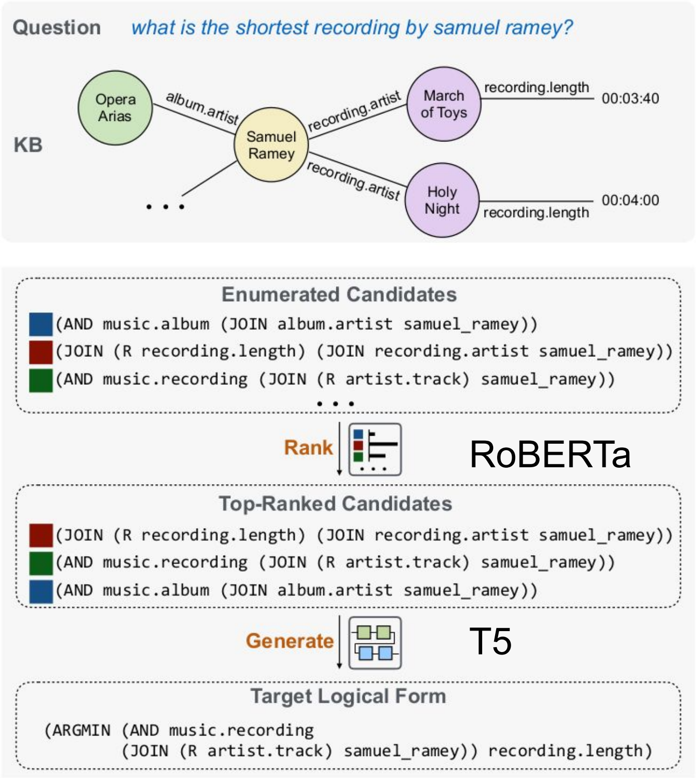
Using both Text and KB for QA

Main idea is to transfer triplets to text to perform retrieve in Faiss-index for both text paragraphs and triplets-based paragraphs.



Ma, K., Cheng, H., Liu, X., Nyberg, E., & Gao, J. (2022, May). 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) (pp. 1605-1620).

RNG-KBQA (Rank-and-Generate)



RoBERTa

T5

Ye, X., Yavuz, S., Hashimoto, K., Zhou, Y., & Xiong, C. (2021). RnG-KBQA: Generation Augmented Iterative Ranking for Knowledge Base Question Answering. arXiv preprint arXiv:2109.08678.

RNG-KBQA (Rank-and-Generate)

- Entity Linking for entities from the question to Freebase
- Compose possible requests to KB.
- Ranking possible requests with RoBERTa.
- Final request generation with T5.

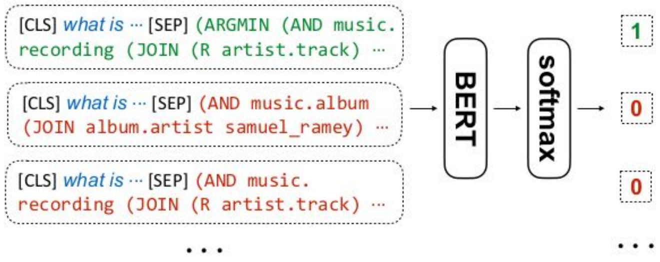


Figure 2: The ranker that learns from the contrast between the ground truth and negative candidates.

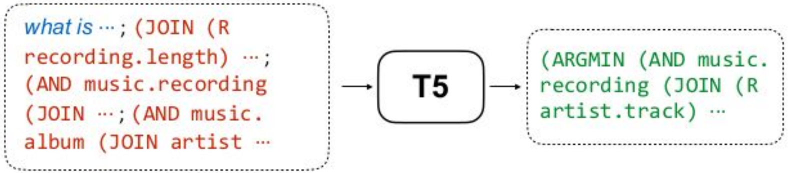


Figure 3: The generation model conditioned on question and top-ranked candidates returned by the ranker.

Dialog Utterances Generation with Facts from KBs

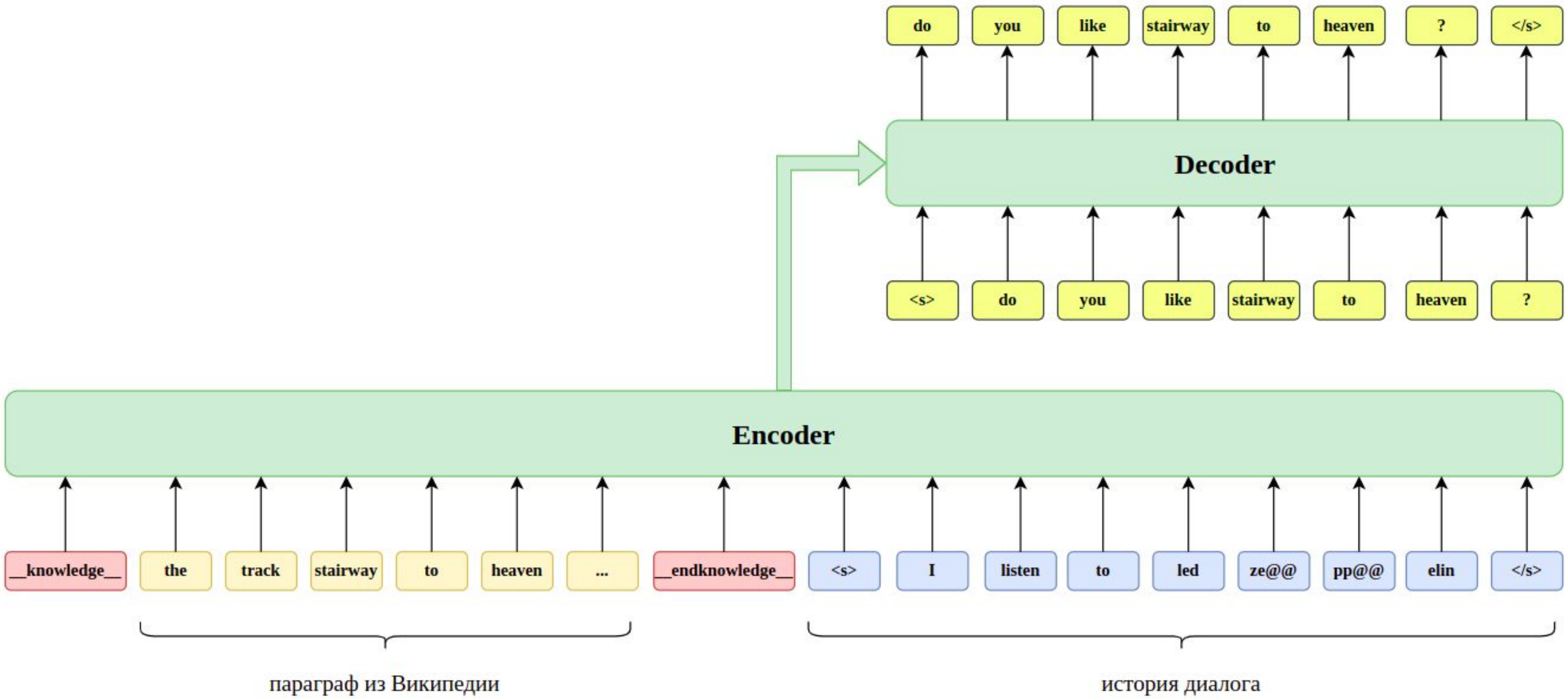
Based on texts:

- Retrieve paragraphs from the KB, selection of the most relevant.
- Generative model gets a dialog context and top-N paragraphs, and generates a response.

Based on graph:

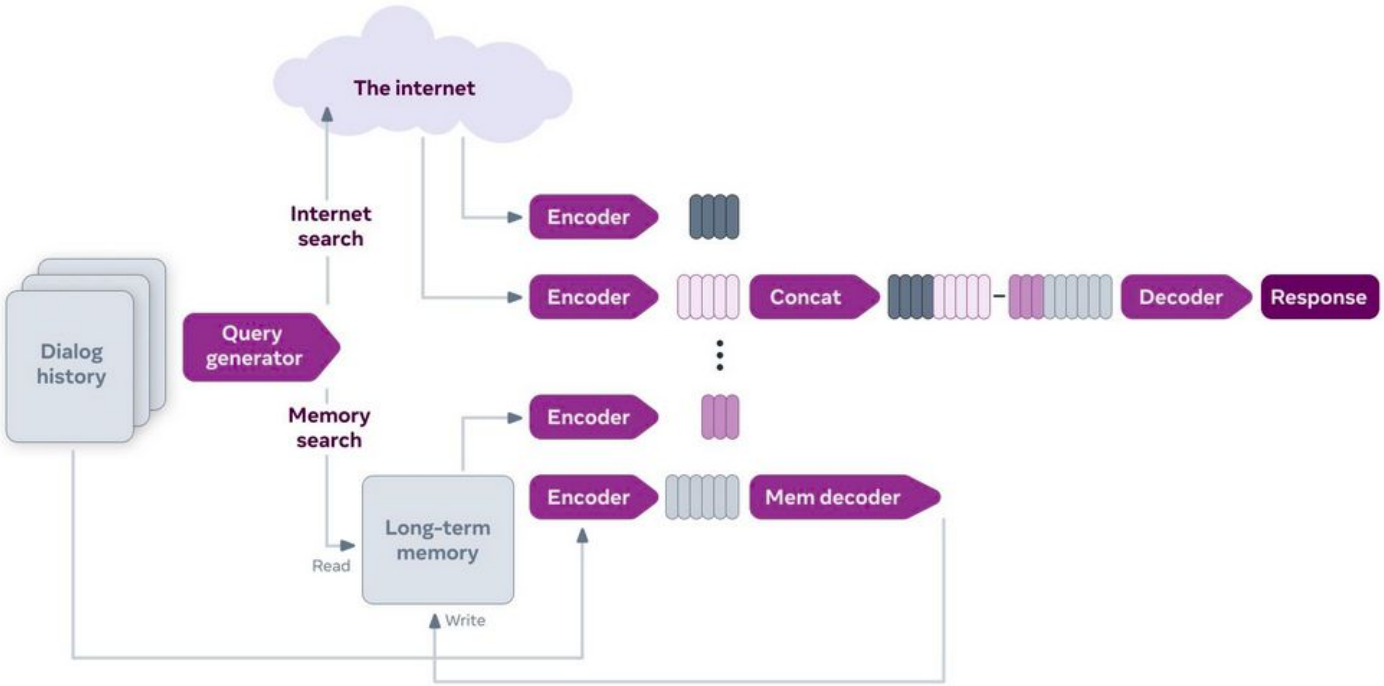
- Entity Linking for entities from the utterance.
- Extracting triplets from KB, finding the most relevant.
- Generative model gets a dialog context and top-N triplets, and generates a response.

BlenderBot – Knowledge-grounding Generation



Roller, S., Dinan, E., Goyal, N., Ju, D., Williamson, M., Liu, Y., ... & Weston, J. (2020). Recipes for building an open-domain chatbot. arXiv preprint arXiv:2004.13637.

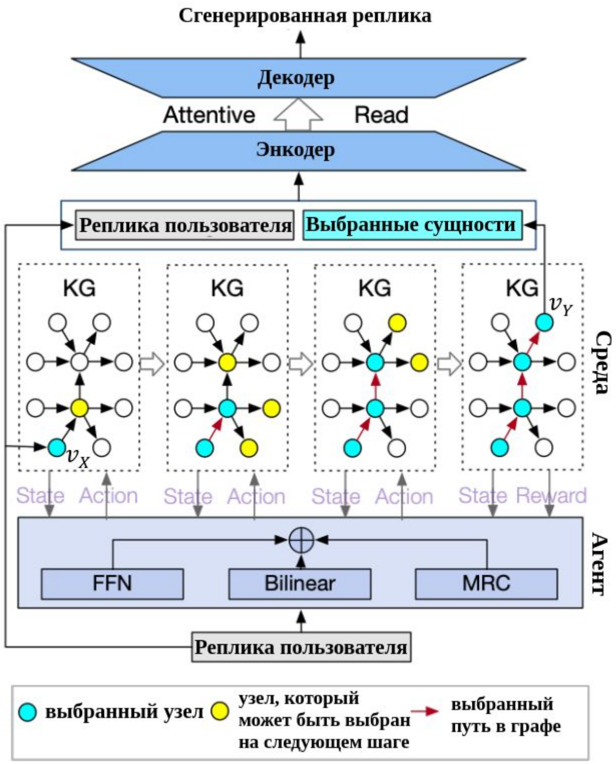
BlenderBot2.0 – Adding Search Results



https://parl.ai/projects/blenderbot2/

Dialog Utterances Generation based on KGs

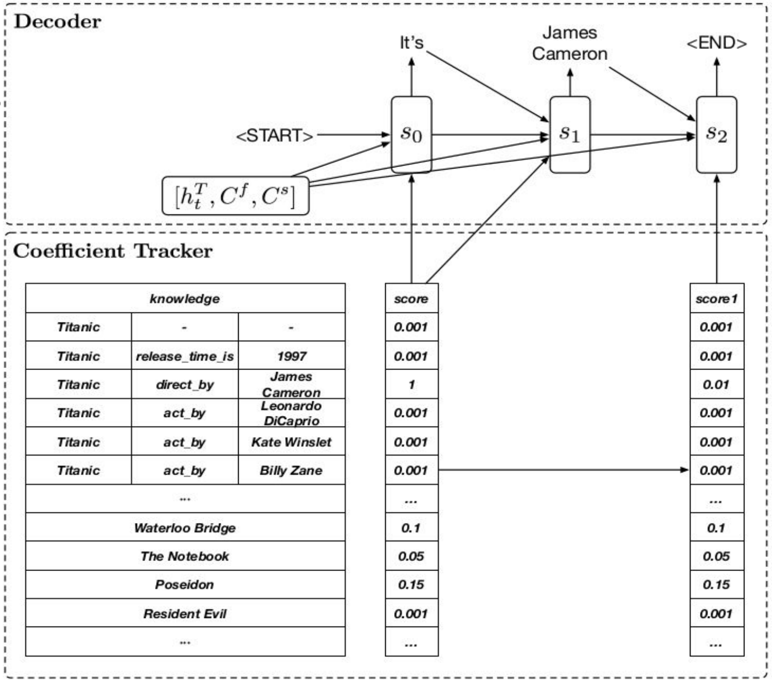
- Entity linking for the initial entity
- On every step, we go over the edges on the graph to determine the next entity
- Dialog context is given to LSTM which hidden depends on the previous hidden, current entity and previous relation
- The next edge depends on the hidden state, embedding of the dialog history, embedding of the possible entity.



Liu, Z., Niu, Z. Y., Wu, H., & Wang, H. (2019). Knowledge aware conversation generation with explainable reasoning over augmented graphs. arXiv preprint arXiv:1903.10245.

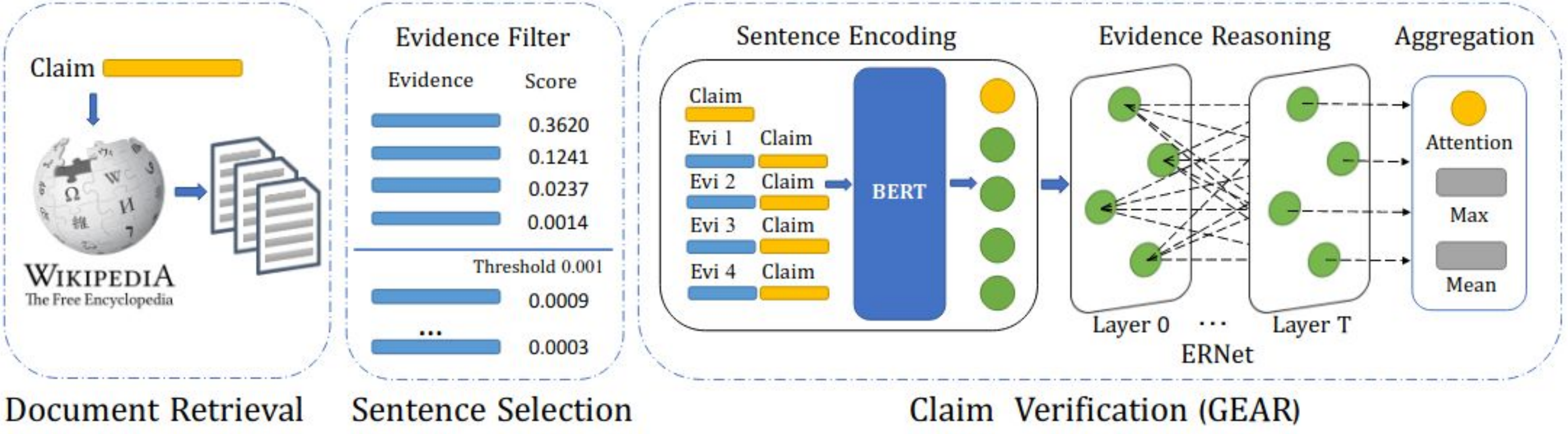
Dialog Utterances Generation based on KGs

- Possible triplets are vectorized to ***h*** by averaging embeddings of entities and relation
- Possible triplets are ranked by the relevance to the dialog history
- Computing weighted sum of the triplets embeddings with ranking coefficients ***C^f***
- Computing the relevance of the entity for the response utterance
- Computing weighted sum of the entities embeddings ***C^s***



Liu, S., Chen, H., Ren, Z., Feng, Y., Liu, Q., & Yin, D. (2018, July). Knowledge diffusion for neural dialogue generation. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics.

Fact Checking



Zhou, J., Han, X., Yang, C., Liu, Z., Wang, L., Li, C., & Sun, M. (2019). GEAR: Graph-based evidence aggregating and reasoning for fact verification. arXiv preprint arXiv:1908.01843.

Fact Checking

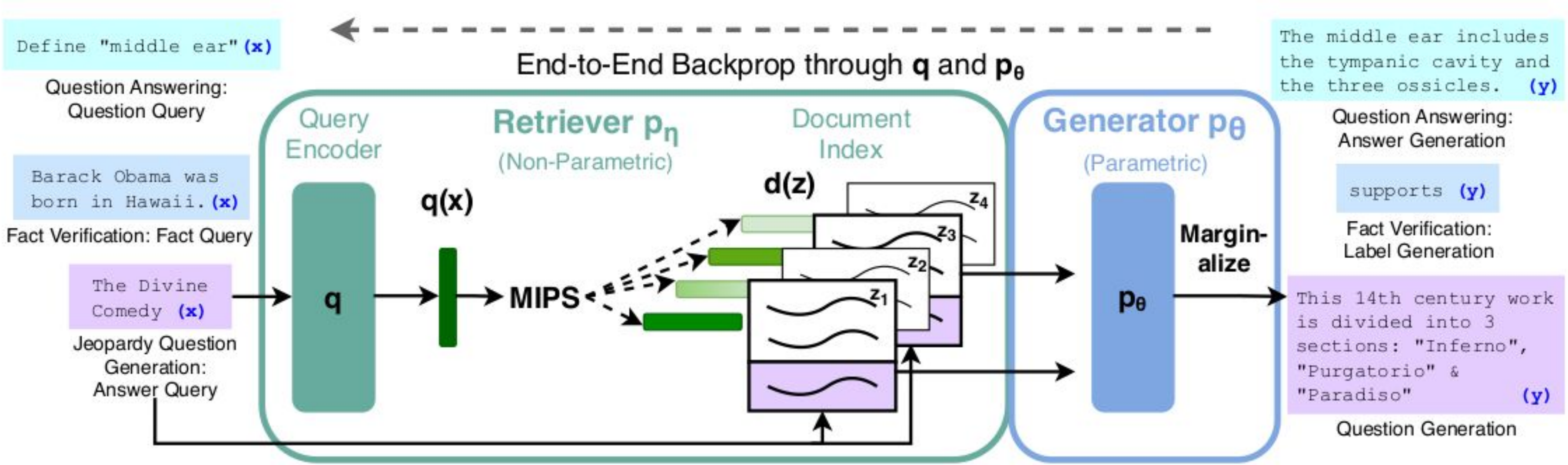


Figure 1: Overview of our approach. We combine a pre-trained retriever (*Query Encoder* + *Document Index*) with a pre-trained seq2seq model (*Generator*) and fine-tune end-to-end. For query x, we use Maximum Inner Product Search (MIPS) to find the top-K documents z_i. For final prediction y, we treat z as a latent variable and marginalize over seq2seq predictions given different documents.

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in Neural Information Processing Systems, 33, 9459-9474.

KBQA Summary

- Re-formulating questions to SPARQL-requests
- Transfer triplets to text to utilize as paragraphs
- Composing and ranking possible requests
- Knowledge-grounding generation of dialogue utterances
- Fact Checking with KBs

Summary

- Factual information is constantly updated, so KBs are necessary to story and utilize actual information instead of re-training constantly generative model.
- Entity Linking is an important task for QA.
- QA can rely on text paragraphs, KBs and both.
- QA can be solved as a span detection and a generation tasks.
- KBs can be used for dialogue response generation.
- KBs can be used for fact checking task.

Artificial Intelligence
Research Institute

airi.net

airi_research_institute

AIRI Institute

AIRI Institute

AIRI_inst

artificial-intelligence-research-institute