Natural Language Processing

Lecture 13 : Question Answering

Master Degree in Computer Engineering
University of Padua
Lecturer : Giorgio Satta

Lecture partially based on material originally developed by :
Cristopher Manning, Stanford University

Question Answering

Aniket Deole from Unsplash

Question answering system **IBM Watson** won the TV game-show *Jeopardy!* in 2011, surpassing humans champions.

Hybrid system, pre-neural network; very complex architecture.


Google
who is the guitar player of led zeppelin?
All
News
Images
Videos
Maps
More
Tools
About 10,600,000 results (0.86 seconds)
Led Zeppelin / Guitarist
Jimmy Page
Led Zeppelin were an English rock band formed in London in 1968. The group consisted of vocalist Robert Plant, guitarist Jimmy Page, bassist/keyboardist John Paul Jones, and drummer John Bonham.
https://en.wikipedia.org › wiki › Led_Zeppelin
Led Zeppelin - Wikipedia

Question Answering

Question answering (QA) systems focus on **factoid questions**, that is, questions that can be answered with simple facts.

Example :
Where is the Louvre Museum located?
What is the average age of the onset of autism?

Non-factoid questions requires articulated answers.

Example :
How does a film qualify for an Academy Award?

Two major paradigms for factoid QA.

Text-based QA

- use efficient algorithms (information retrieval) to find relevant documents from text collection
- use reading comprehension algorithms on relevant documents to select span of text containing the answer

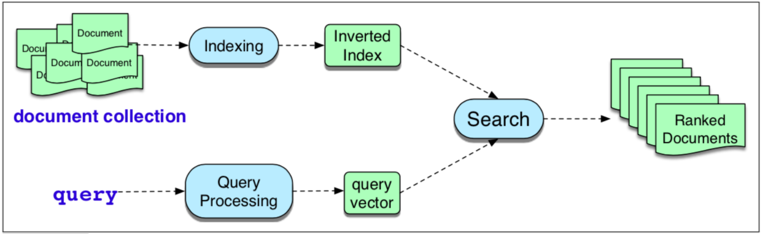
Knowledge-based QA

- produce a semantic representation of the query
- match semantic representation against fact database

Text-based QA

Umar Mukhtar from Pexels

Information retrieval



The first step in text-based QA uses information retrieval (IR) systems to map input queries to a set of documents from some collection, ordered by **relevance**.

IR uses vector space models, sparse or dense.

Machine reading

Machine reading

The second step in text-based QA is called span based **machine reading** (related to human reading comprehension)

- the input is a factoid question along with a passage that could contain the answer
- the output is the answer fragment, or else NULL if there is no answer in the passage

Example :
question: "How tall is Mt. Everest?"
passage: "Mount Everest, reaching 29,029 feet at its summit, is located in Nepal and Tibet ..."
output fragment: "29,029 feet"

Machine reading

Let $q = q_1, \ldots, q_N$ be a **query** and let $p = p_1, \ldots, p_M$ be a **passage**, where q_i and p_j are tokens.

Lengths are imbalanced: $N \sim 15$, $M \sim 100$.

A **span** is any fragment $p_i, \ldots, p_j$ of p, with i the start position and $j \geqslant i$ the end position.

The goal is to compute the probability $P_{\text{span}}(p_i, \ldots, p_j \mid q, p)$ that span $p_i, \ldots, p_j$ is the answer to q.

Machine reading

We present two simple **neural** approaches to span based machine reading, computing $P_{\text{span}}(p_i, \ldots, p_j \mid q, p)$ in two different ways

- on the basis of BERT-like, pre-treined contextual embeddings
- on the basis of RNN and attention-like mechanisms

Chronologically these have been proposed in the inverse order, second approach was conceived before transformers.

These approaches are not state-of-the-art (SoTA) but, when hyperparameters are properly tuned, they provide very good performance.

Using contextual embeddings

Jonathan Roger from Unsplash

Using contextual embeddings

To compute the probability $P_{\text{span}}(p_i, \ldots, p_j \mid q, p)$, we make the **simplifying** assumption that

$$P_{\text{span}}(p_i, \ldots, p_j \mid q, p) \quad = \quad P_{\text{start}}(i \mid q, p) \cdot P_{\text{end}}(j \mid q, p)$$

Independence assumption for start and end of span, when conditioned on both query and passage.

Using contextual embeddings

Pre-trained encoder BERT used to encode question and passage separated by a [SEP] token.

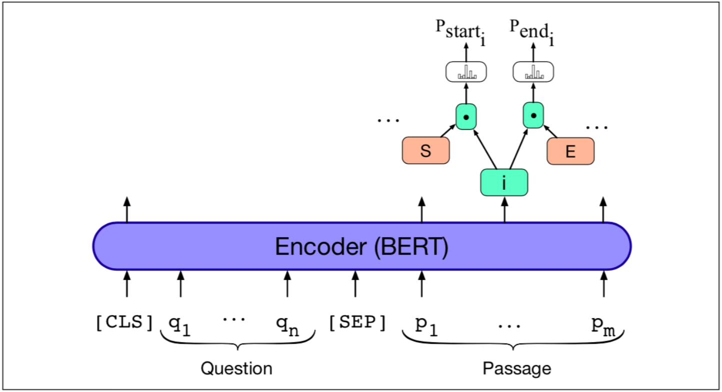
Let $\mathbf{e}(p_i)$ be the BERT embedding of token p_i within passage p.

During fine-tuning, **start vector** $\mathbf{S}$ is learned to estimate start probabilities for each position i, using dot product and softmax

$$P_{\text{start}}(i \mid q, p) \quad = \quad \frac{\exp(\mathbf{S} \cdot \mathbf{e}(p_i))}{\sum_j \exp(\mathbf{S} \cdot \mathbf{e}(p_j))}$$

Similarly, we learn **end vector** $\mathbf{E}$ to estimate $P_{\text{end}}(j \mid q, p)$.

Using contextual embeddings



Using contextual embeddings

At **inference** time, the score of a candidate span from position i to j is

$$\mathbf{S} \cdot \mathbf{e}(p_i) + \mathbf{E} \cdot \mathbf{e}(p_j)$$

The model prediction is the highest scoring span with $j \geqslant i$.

For each **training** instance, compute the negative sum of the log-likelihoods of the correct start position i^* and the correct end position j^*

$$L = -\log P_{\text{start}}(i^* \mid q, p) - \log P_{\text{end}}(j^* \mid q, p)$$

Averaging for all instances provides the fine-tuning **loss**.

Using contextual embeddings

Many datasets contain **negative examples**, that is, (q, p) pairs in which the answer to q is not in the passage p.

Negative examples are conventionally treated as having start and end indices pointing to the [CLS] special token.

Pratical issues

For many datasets the annotated documents/passages have length M larger than the maximum 512 input tokens BERT allows.

In such cases slide over the entire document a window of size 512 minus N (the question length) minus the number of special tokens. Use a **stride** of $\Delta = 128$ tokens.

This produces several instances in the training set for the given document, most of which are negative examples. If number of negative examples is disproportionately large, then you need to **balance** the training set by downsampling.

Stanford attentive reader

Eisenstein §17.5.2

Aleksandar Pasaric from Pexels

Stanford attentive reader

Stanford attentive reader uses RNN combined with an attention-like mechanism.

Assume query $q = q_1, \ldots, q_N$ and passage $p = p_1, \ldots, p_M$, where q_t and $p_{t'}$ are tokens.

Let $\mathbf{e}(q_t), \mathbf{e}(p_{t'})$ be static (non-contextual) embeddings associated with tokens q_t and $p_{t'}$, respectively.

We use **bidirectional** LSTM to encode individual tokens from query and passage.

Query and passage encoded independently.

We start with monodirectional embeddings

$$
\begin{aligned}
\overrightarrow{\mathbf{h}}_t^{(q)} &= \mathrm{LSTM}(\overrightarrow{\mathbf{h}}_{t-1}^{(q)}, \mathbf{e}(q_t)) \\
\overleftarrow{\mathbf{h}}_t^{(q)} &= \mathrm{LSTM}(\overleftarrow{\mathbf{h}}_{t+1}^{(q)}, \mathbf{e}(q_t)) \\
\overrightarrow{\mathbf{h}}_t^{(p)} &= \mathrm{LSTM}(\overrightarrow{\mathbf{h}}_{t-1}^{(p)}, \mathbf{e}(p_t)) \\
\overleftarrow{\mathbf{h}}_t^{(p)} &= \mathrm{LSTM}(\overleftarrow{\mathbf{h}}_{t+1}^{(p)}, \mathbf{e}(p_t))
\end{aligned}
$$

Stanford attentive reader

We concatenate monodirectional embeddings to encode individual **passage tokens**

$$\mathbf{h}_t^{(p)} = [\overrightarrow{\mathbf{h}}_t^{(p)}; \overleftarrow{\mathbf{h}}_t^{(p)}]$$

We pick up boundary query embeddings to encode the **entire query** q

$$\mathbf{u}^{(q)} = [\overrightarrow{\mathbf{h}}_N^{(q)}; \overleftarrow{\mathbf{h}}_0^{(q)}]$$

Stanford attentive reader

We derive an attention distribution by computing a vector of **bilinear products** and by applying softmax

$$\begin{aligned}\tilde{\alpha}_i &= (\mathbf{u}^{(q)})^\top \mathbf{W}\,\mathbf{h}_i^{(p)} \\ \boldsymbol{\alpha} &= \text{softmax}(\tilde{\boldsymbol{\alpha}})\end{aligned}$$

where $\mathbf{W}$ is a learned matrix.

We then use attention to combine passage tokens, and compute an output vector

$$\mathbf{o} = \sum_{i=1}^{M} \alpha_i \mathbf{h}_i^{(p)}$$

Keep in mind that $\mathbf{o}$ is also a function of query q.

Stanford attentive reader

Each candidate answer c is encoded as a vector $\mathbf{x}_c$, using some function of the vectors $\mathbf{h}_i^{(p)}$ in the corresponding span.

For instance, combination of the span boundary vectors.

Finally, the score of each candidate c is computed as the inner product (similarity)

$$\hat{c} \quad = \quad \underset{c}{\operatorname{argmax}} \ \mathbf{o} \cdot \mathbf{x}_c$$

This architecture can be trained **end-to-end** from a loss based on the log-likelihood of the correct answers.

Practical issues

Practical issues

Performance of the two previous models can be **improved** using any of the following ideas.

Encode q by a weighted (learnable) combination of all query states, not just the boundary states.

Use more than one layer when implementing BiLSTM.

Use some similarity score between entire q and p.

Several proposals for similarity in the literature.

Practical issues

Concatenate vector representation of each token in passage with the following additional **features**

- POS and NER tags, 1-hot encoded
- term frequency (unigram probability)
- does p_t appear in q? exact match, uncased match, lemma

Word embeddings should implicitly represent this information, but in practice adding explicit features helps.

Retrieval-augmented generation

©Pixabay

Retrieval-augmented generation

Recall that using LLM and prompting, we can recast the task of question answering as word prediction.

Example :

```
Q: Who wrote the book "The Origin of Species"?
A:
```

LLMs have an enormous amount of knowledge encoded in their parameters. However, LLMs

- may lead to hallucination
- may not be up-to-date with their knowledge
- do not provide textual evidence to support their answer
- are unable to answer questions from proprietary data

Retrieval-augmented generation

A standard reader algorithm is to generate from a large language model, conditioned on the retrieved passages.

Example :

```
retrieved passage 1
...
retrieved passage n
Based on these texts, answer the following question:
Q: Who wrote the book "The Origin of Species"?
A:
```

This method is known as **retrieval-augmented generation**, or RAG for short.

Retrieval-augmented generation

More formally, we **reduce** the Q/A problem to the problem of computing the following probability.

Assume a query q and let $R(q)$ be the set of retrieved passages based on q. Then (symbol ';' denotes string concatenation)

$$
\begin{aligned}
& P(x_1, \ldots, x_n) \\
& \quad = \prod_{i=1}^{n} P(x_i \mid R(q) \,;\, \texttt{prompt} \,;\, \texttt{[Q:]} \,;\, q \,;\, \texttt{[A:]} \,;\, x_{<i})
\end{aligned}
$$

Details of prompt engineering also have to be worked out, like deciding whether to demarcate passages with [SEP] tokens.

Research papers

Alfons Morales on Unsplash

Title: A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task
Authors: Danqi Chen, Jason Bolton, Christopher Manning
Conference: ACL 2016
Content: This paper reports a thorough examination of the reading comprehension task. It is shown that simple, carefully designed systems can obtain important performance.

https://aclanthology.org/P16-1223/

Title: Bidirectional Attention Flow for Machine Comprehension
Authors: Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, Hannaneh Hajishirzi
Conference: ICLR 2017
Content: Machine comprehension requires modeling complex interactions between the context and the query. In this paper we introduce the Bi-Directional Attention Flow (BIDAF) network, a multi-stage hierarchical process that represents the context at different levels of granularity and uses bi-directional attention flow mechanism to obtain a query-aware context representation.

https://arxiv.org/abs/1611.01603

Miscellanea

Datasets

Several datasets for machine reading, containing tuples of the form (question, passage, answer).

http://nlpprogress.com/english/question_answering.html

SQuAD: Stanford question answering dataset consists of passages from Wikipedia and associated questions whose answers are spans from the passage.

150K questions, including negative examples.

Natural Question: dataset of real, anonymized queries to the Google search engine linked to a Wikipedia article that may or may not contain the answer.

300K questions.

Example : Sample from SQuAD 2.0 dataset.

Beyoncé Giselle Knowles-Carter (born September 4, 1981) is an American singer, songwriter, record producer and actress. Born and raised in Houston, Texas, she performed in various singing and dancing competitions as a child, and rose to fame in the late 1990s as lead singer of R&B girl-group Destiny's Child. Managed by her father, Mathew Knowles, the group became one of the world's best-selling girl groups of all time. Their hiatus saw the release of Beyoncé's debut album, Dangerously in Love (2003), which established her as a solo artist worldwide, earned five Grammy Awards and featured the Billboard Hot 100 number-one singles "Crazy in Love" and "Baby Boy".
Q: "In what city and state did Beyoncé grow up?" A: "Houston, Texas"
Q: "What areas did Beyoncé compete in when she was growing up?" A: "singing and dancing"
Q: "When did Beyoncé release Dangerously in Love?" A: "2003"

Evaluation

Machine reading systems often evaluated using two metrics.

Ignoring punctuation and articles like a, an, the.

Exact match: percentage of predicted answers that match the gold answer exactly.

For each question, treat prediction and gold as a bag of tokens. Then compute

- **precision** as the ratio of the number of shared words to the total number of words in the prediction
- **recall** as the ratio of the number of shared words to the total number of words in the ground truth
- **F1 score** as the harmonic mean of precision and recall

and return average F1 over all questions.

SQuAD leaderboard

Rank	Model	EM	F1
	Human Performance *Stanford University* (Rajpurkar & Jia et al. '18)	86.831	89.452
1 Jun 04, 2021	IE-Net (ensemble) *RICOH_SRCB_DML*	**90.939**	**93.214**
2 Feb 21, 2021	FPNet (ensemble) *Ant Service Intelligence Team*	90.871	93.183
3 May 16, 2021	IE-NetV2 (ensemble) *RICOH_SRCB_DML*	90.860	93.100
4 Apr 06, 2020	SA-Net on Albert (ensemble) *QIANXIN*	90.724	93.011

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

Last accessed: July 2021.

Answer sentence selection

Answer sentence selection (AS2) is a task related to machine reading.

Given a question and a **document**, choose the sentence in the document that contains the answer fragment.

No fragment selection is needed.

Using the document as **context** helps identifying the sentence with the correct answer.

Answer fragment may contain a pronoun that needs to be resolved in the context of the document.

Knowledge-based QA

©Pixels

Knowledge-based QA

Text-based QA uses textual information over the web (unstructured).

Knowledge-based QA answers a natural language question by mapping it to a query over some structured knowledge repository.

Knowledge-based QA

Two main approaches to knowledge-based QA.

Graph-based QA: models the knowledge base as a graph, with entities as nodes and relations as edges.

Example : Google knowledge graph.

QA by semantic parsing: maps queries to logical formulas, and queries a fact database.

Example : "What states border Texas?" is translated into lambda expression $\lambda x.\text{state}(x) \wedge \text{borders}(x, \text{texas})$ and later mapped to SQL query.

Entity linking

Both approaches to knowledge-based QA require algorithms for entity linking.

Entity linking is the task of associating a mention in text with the representation of some real-world entity in an ontology.

This is a way of having a canonical representation for entities.

The most common ontology for factoid question answering is Wikipedia, in which case the task is called **wikification**.

Entity linking

Entity linking is done in (roughly) two stages

- mention detection
- mention disambiguation

Two approaches to entity linking

- classical approaches based on dictionaries and network structure
- modern approaches based on neural networks