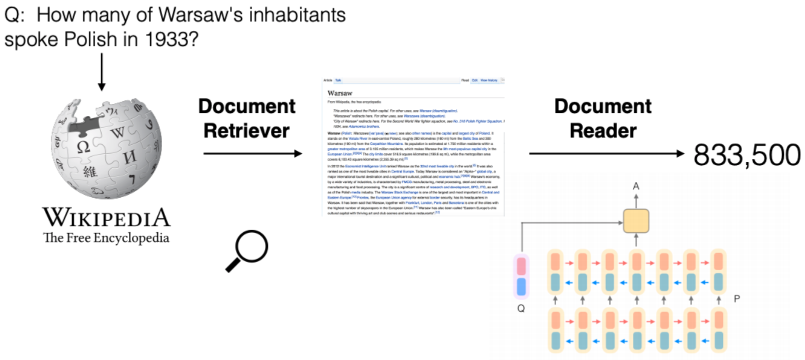
Question Answering

Natural Language Processing: Jordan Boyd-Graber
University of Maryland
DR. QA

Adapted from material from Danqi Chen

Overview of the Document Reader Question Answering



Good source code available!

Big idea

Super Bowl 50 was an American football game to determine the champion of the National Football League (NFL) for the 2015 season. The American Football Conference (AFC) champion Denver Broncos defeated the National Football Conference (NFC) champion Carolina Panthers 24–10 to earn their third Super Bowl title. The game was played on February 7, 2016, at Levi's Stadium in the San Francisco Bay Area at Santa Clara, California. As this was the 50th Super Bowl, the league emphasized the "golden anniversary" with various gold-themed initiatives, as well as temporarily suspending the tradition of naming each Super Bowl game with Roman numerals (under which the game would have been known as "Super Bowl L"), so that the logo could prominently feature the Arabic numerals 50.

Q: Which NFL team represented the AFC at Super Bowl 50?

A: Denver Broncos

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \quad (1)$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \quad (2)$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \tag{1}$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \tag{2}$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \quad (1)$$
$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \quad (2)$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \quad (1)$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \quad (2)$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \quad (1)$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \quad (2)$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

Start and End Probabilities

$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\} \quad (1)$$

$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\} \quad (2)$$

1. A vector representing our question
2. Vector representing each word in the query text
3. Parameter: here's the start/end of the answer

This is your objective function! Will backprop into each of these parameters.

Question Encoding

$$\vec{q} = \sum_j b_j \vec{q}_j \tag{3}$$

$$b_j = \frac{\exp\left\{\vec{w} \cdot q_j\right\}}{\sum_{j'} \exp\left\{w \cdot q_{j'}\right\}} \tag{4}$$

Question Encoding

$$\vec{q} = \sum_j b_j \vec{q}_j \tag{3}$$

$$b_j = \frac{\exp\left\{\vec{w} \cdot q_j\right\}}{\sum_{j'} \exp\left\{w \cdot q_{j'}\right\}} \tag{4}$$

Question vector is a weighted sum

Question Encoding

$$\vec{q} = \sum_j b_j \vec{q}_j \tag{3}$$

$$b_j = \frac{\exp\left\{\vec{w} \cdot q_j\right\}}{\sum_{j'} \exp\left\{w \cdot q_{j'}\right\}} \tag{4}$$

The weight is a scalar

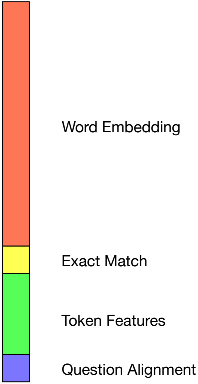
Question Encoding

$$\vec{q} = \sum_j b_j \vec{q}_j \tag{3}$$

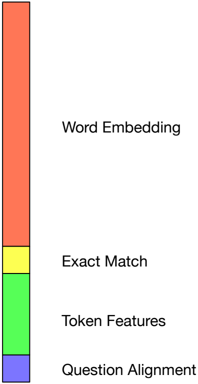
$$b_j = \frac{\exp\left\{\vec{w} \cdot q_j\right\}}{\sum_{j'} \exp\left\{w \cdot q_{j'}\right\}} \tag{4}$$

A focus parameter learns how to focus on particular words in the question

Paragraph Encoding

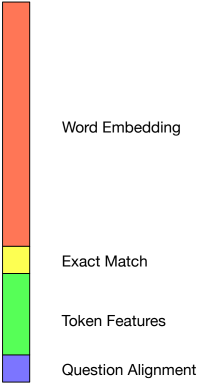


Paragraph Encoding



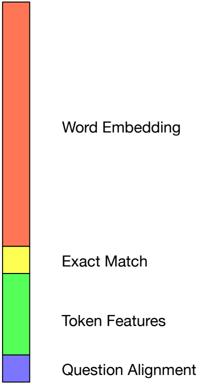
pre-trained word embedding

Paragraph Encoding



Part of speech, NER tags, normalized term frequency

Paragraph Encoding

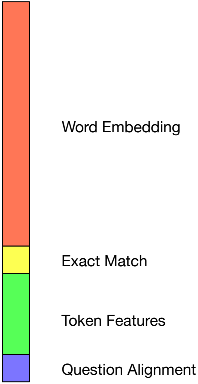


Who is the leader of the US

Donald Trump is the president of the United States

$$a_{i,j} = \frac{\exp\left\{\vec{E}(p_i) \cdot E(q_j)\right\}}{\sum_{j'} E(p_i) \cdot E(q_{j'})} \tag{5}$$

Paragraph Encoding



Who is the leader of the US

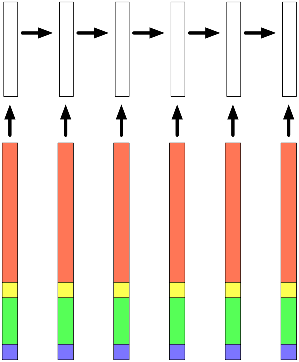
Donald Trump is the president of the United States

$$a_{i,j} = \frac{\exp\left\{\vec{E}(p_i) \cdot E(q_j)\right\}}{\sum_{j'} E(p_i) \cdot E(q_{j'})} \tag{5}$$

Paragraph Encoding

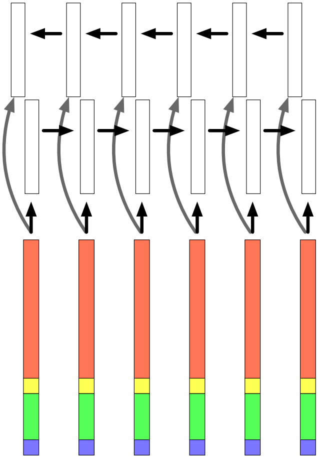
Create learned representations

Paragraph Encoding



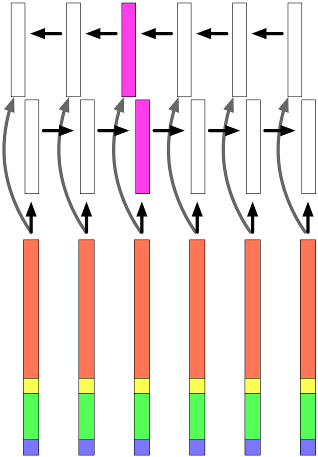
LSTM: encode contextual effects

Paragraph Encoding



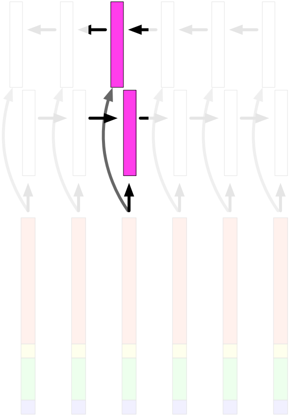
Add a backwards direction as well (bi-directional LSTM)

Paragraph Encoding



Use the concatenation of these two hidden layers as the representation of the word

Paragraph Encoding



$$P_{\text{start}}(i) \propto \exp\{\vec{p}_i W_s \vec{q}\}$$
$$P_{\text{end}}(i) \propto \exp\{\vec{p}_i W_e \vec{q}\}$$

Implementation

- Trained on passages
- Backprop through all layers
- Look at code

```
# RNN document encoder
self.doc_rnn = layers.StackedBRNN(
    input_size=doc_input_size,
    hidden_size=args.hidden_size,
    num_layers=args.doc_layers,
    dropout_rate=args.dropout_rnn,
    dropout_output=args.dropout_rnn_output,
    concat_layers=args.concat_rnn_layers,
    rnn_type=self.RNN_TYPES[args.rnn_type],
    padding=args.rnn_padding,
)

# RNN question encoder
self.question_rnn = layers.StackedBRNN(
    input_size=args.embedding_dim,
    hidden_size=args.hidden_size,
    num_layers=args.question_layers,
    dropout_rate=args.dropout_rnn,
    dropout_output=args.dropout_rnn_output,
    concat_layers=args.concat_rnn_layers,
    rnn_type=self.RNN_TYPES[args.rnn_type],
    padding=args.rnn_padding,
)
```

`https://github.com/facebookresearch/DrQA/`

More complicated models