Bi-Directional Attention Flow for Machine Comprehension (BiDAF)

ICLR 2017

2019.01.28

발표자 박성현

1 Introduction

Machine Comprehension

▶ Machine Comprehension

- 주어진 Context를 이해하고 주어진 질문에 답변하는 문제를 Machine Comprehension이라고 표현.

- 예시

Joe went to the kitchen. Fred went to the kitchen. Joe picked up the milk.
Joe travelled to the office. Joe left the milk. Joe went to the bathroom.
Where is the milk now? A: office
Where is Joe? A: bathroom
Where was Joe before the office? A: kitchen

- Joe가 어디에 있고 무엇을 하고 있는지에 대한 정보가 포함되어 있는 문장을 읽고, 질문에 답변하는 Task

1 Introduction

Motivation

▶ 기존의 Attention mechanism 특징

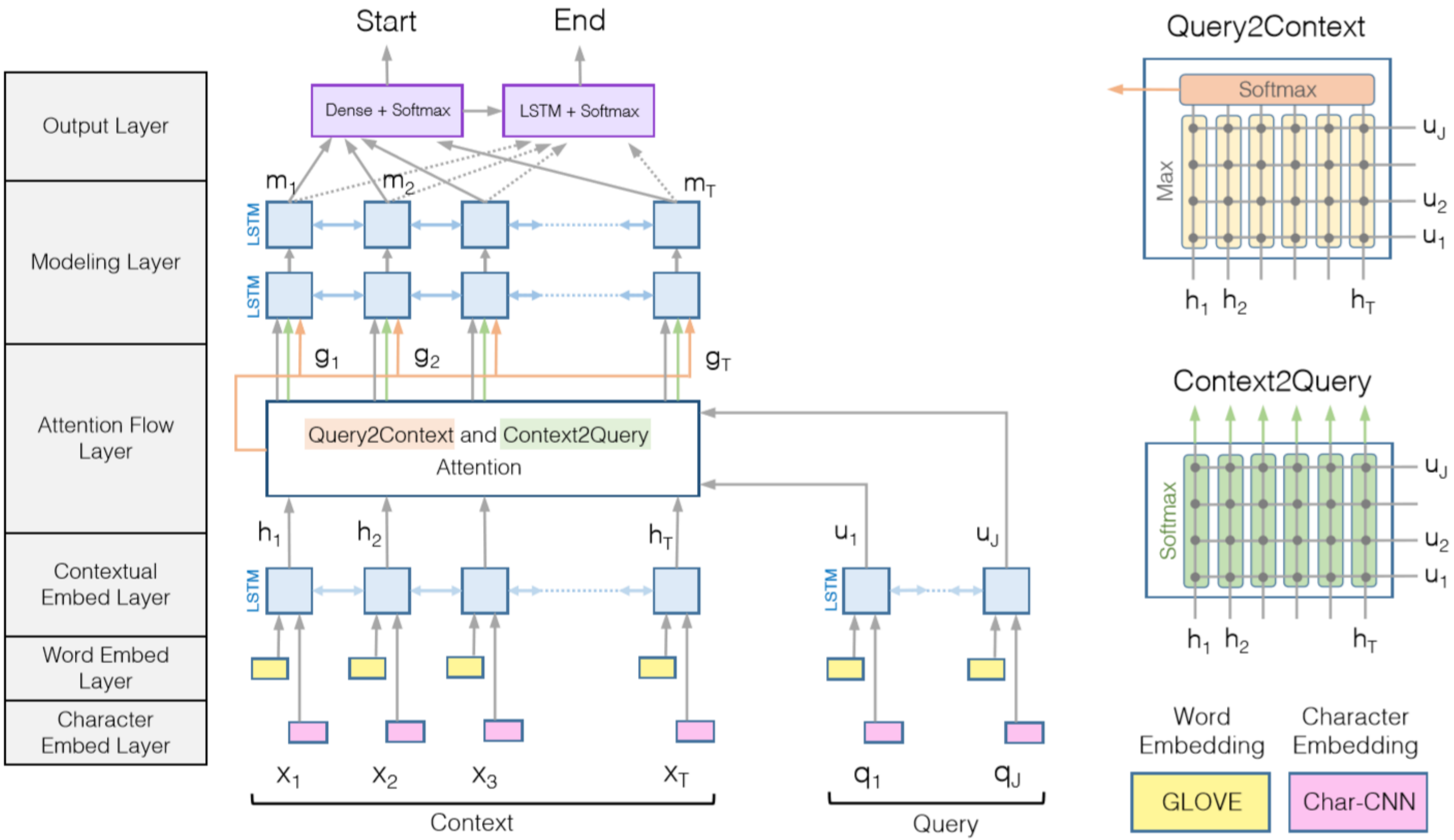
- **Attention weights** are often used to extract the most relevant information from the context for answering the question by **summarizing the context into a fixed-size vector**
- Attention weights are often **temporally dynamic**, whereby the attention weights at the current time step are a function of the attended vector at the previous time step
- They are **Uni-directional**, wherein the query attends on the context paragraph or the image

▶ BIDAF

- Character-level, word-level and contextual embedding
- Use **bi-directional attention flow**
- **Memory-less** attention mechanism

2 Model

Bi-Directional Attention Flow Model

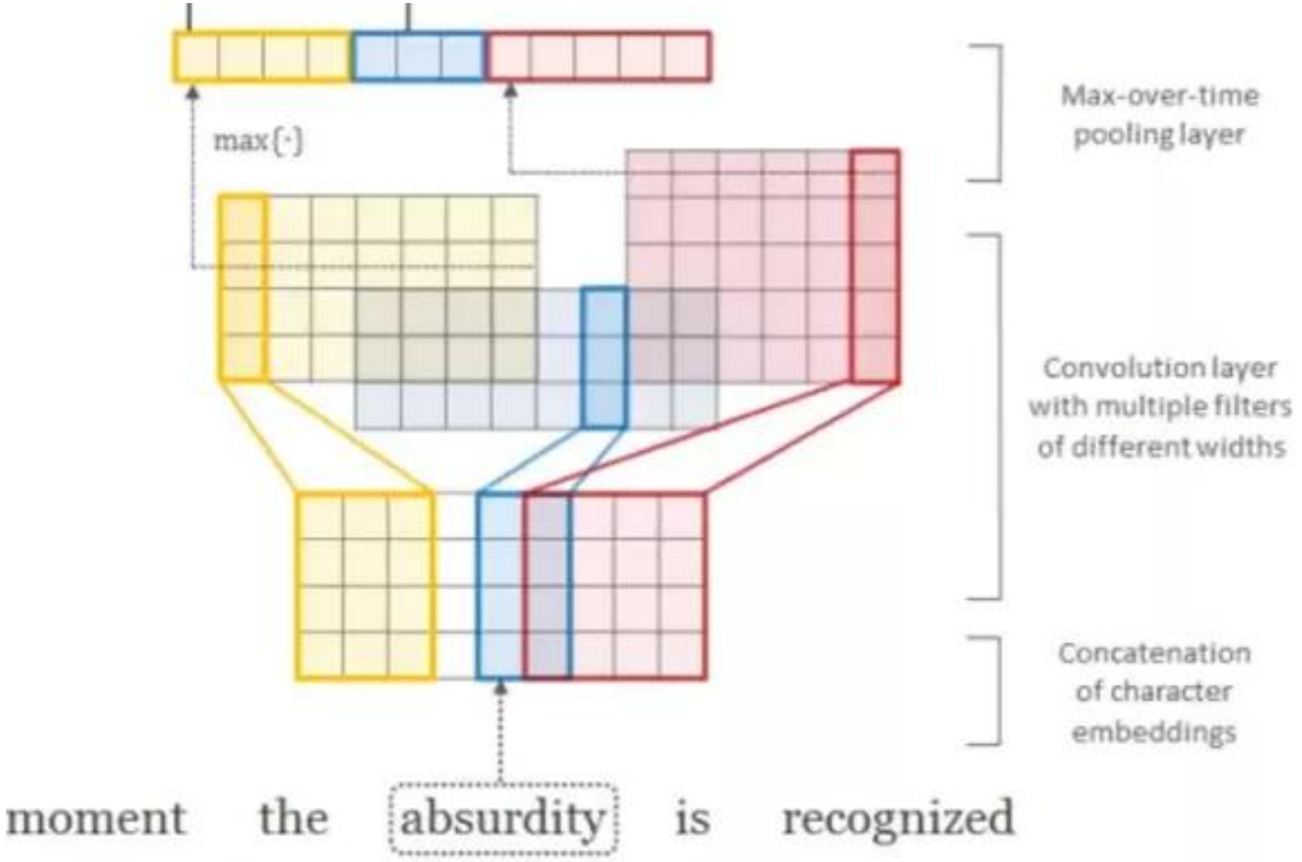


2 Model

Character Embedding Layer

▶ Char-CNN (by Kim)

- Mapping each word to a high-dimensional vector space
- Outputs of the CNN are max-pooled over the entire width to obtain fixed-size vector for each word
- Input : $\{x_1, \dots, x_T\}$ and $\{q_1, \dots, q_J\}$ represent the words in the input context paragraph and query



2 Model

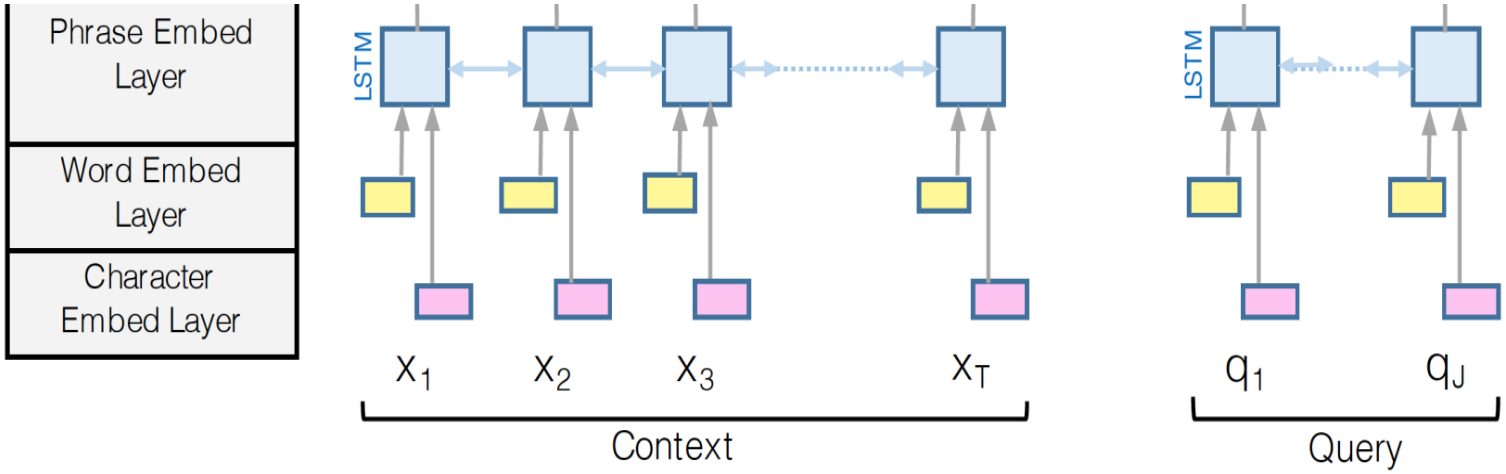
Word Embedding Layer

▶ GloVe (by Pennington)

- Mapping each word to a high-dimensional vector space
- Use pre-trained word vectors to obtain the fixed word embedding of each word
- Input : $\{x_1, \ldots , x_T\}$ and $\{q_1, \ldots , q_J\}$ represent the words in the input context paragraph and query

▶ Highway Network

- The concatenation of character and word embedding vectors is passed to a two-layer Highway Network
- The outputs of the Highway Network are two sequences of d-dimensional vectors
- Two matrices : $X \in R^{d \times T}$ for the context / $Q \in R^{d \times J}$ for the query

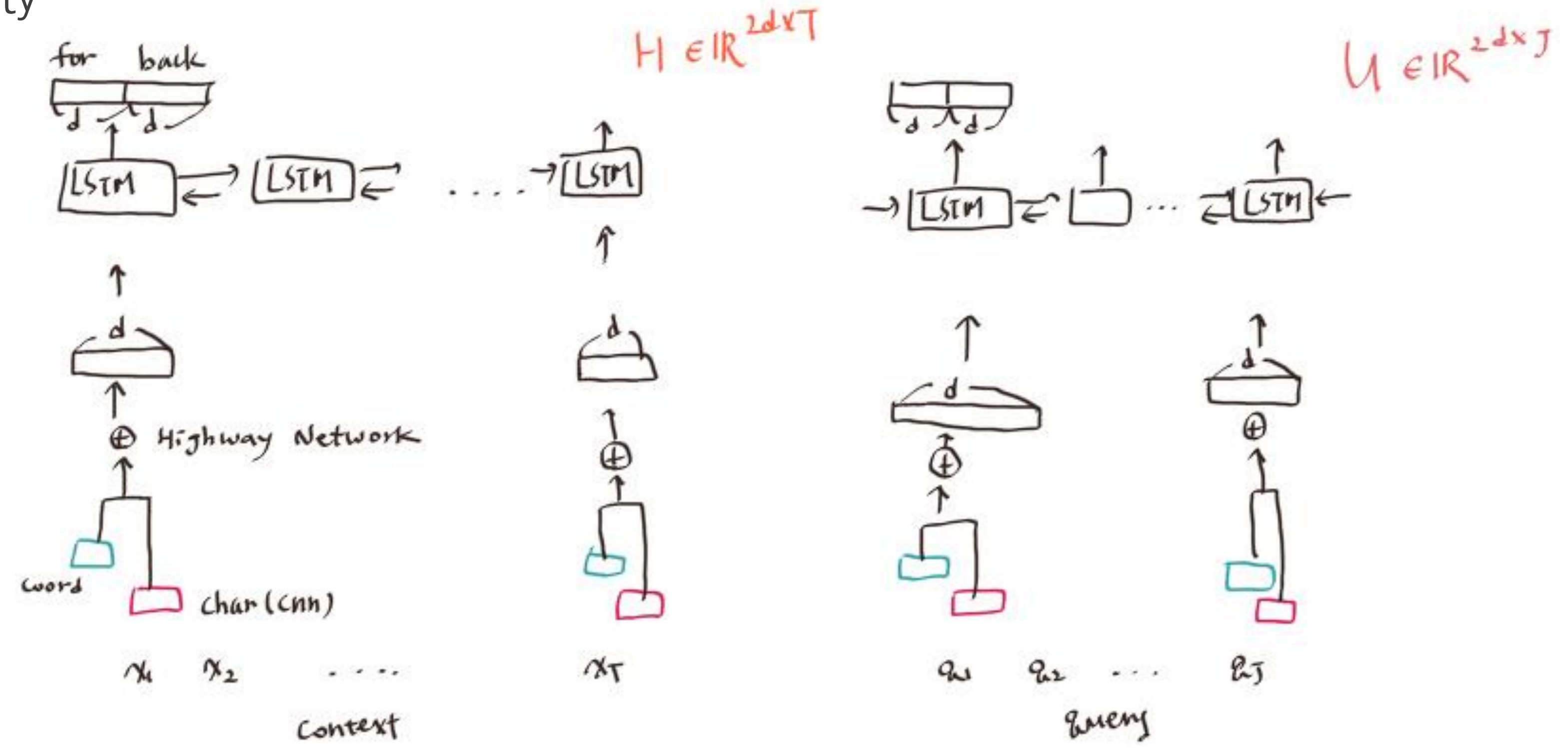


2 Model

Contextual Embedding Layer

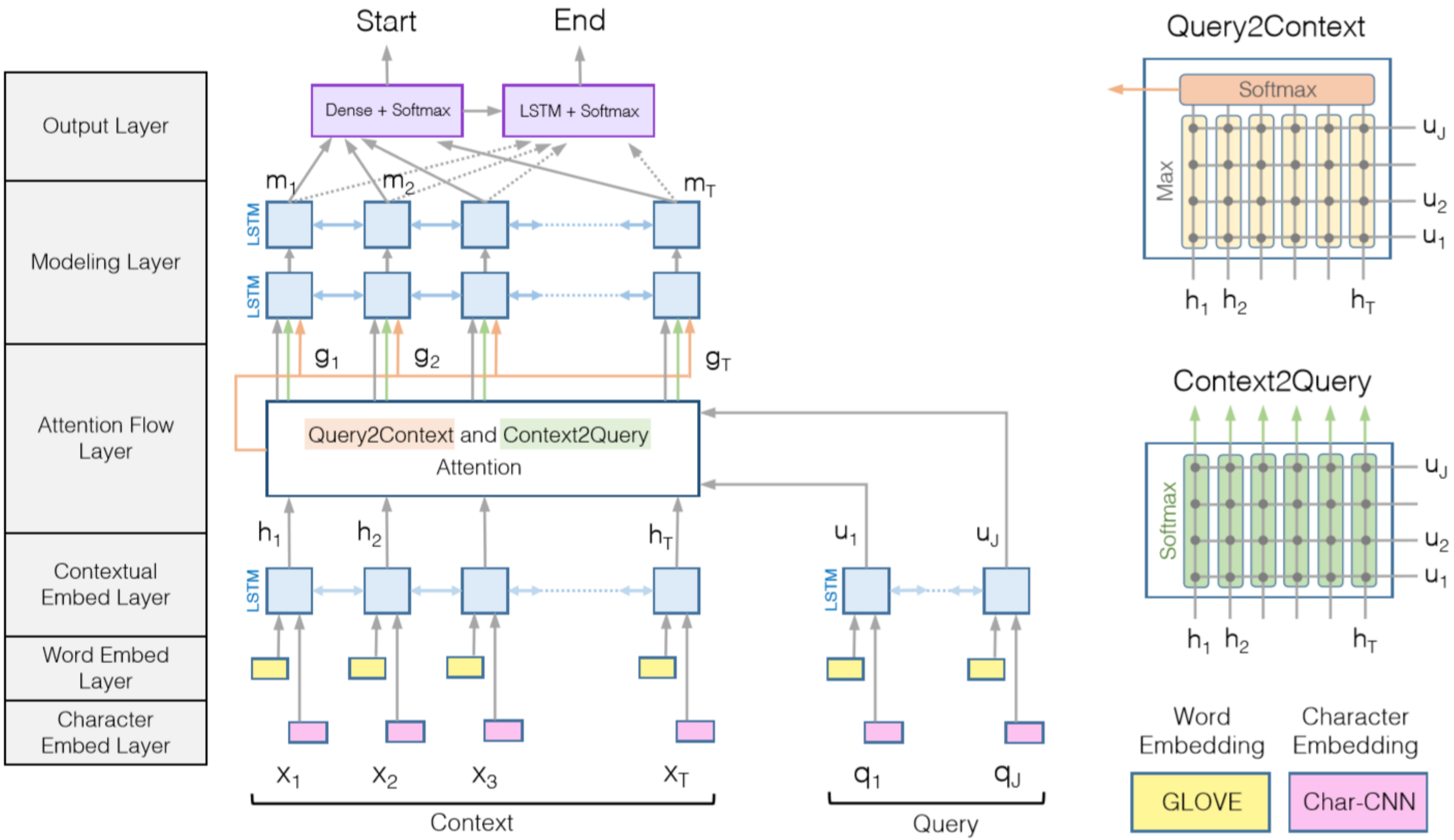
▶ Contextual Embedding Layer

- Use LSTM on top of the embeddings provided by the previous layers
- Concatenate the outputs of the two LSTMs
- Outputs : $H \in R^{2d \times T}$ from the **context word vectors** $\boldsymbol{X}$ / $U \in R^{2d \times J}$ from **query word vectors** $\boldsymbol{Q}$
- First three layers of the model are computing features from the query and context at different levels of granularity



2 Model

Bi-Directional Attention Flow Model



2 Model

Attention Flow Layer

▶ Attention Flow Layer

- Not used to summarize the query and context into single feature vectors
- Instead, attention vector at each time step, along with the embeddings from previous layers, are allowed to flow through to the subsequent modeling layer
- This reduces the information loss caused by early summarization

$$\mathbf{S}_{tj} = \alpha(\mathbf{H}_{:t}, \mathbf{U}_{:j}) \in \mathbb{R}$$

S_{tj} : the similarity between t-th context word and j-th query word

α : a trainable scalar function that encodes the similarity between its two input vectors

$H_{:t}$: t-th column vector of H

$U_{:j}$: j-th column vector of U

$\alpha(h,u) = w_{(s)}^{T}[h; u; h \circ u]$

2 Model

Attention Flow Layer

▶ Context-to-query Attention

- 어떤 Query의 단어가 각각의 Context 단어와 가장 연관되어 있는지를 알아냄
- $U \in R^{2d \times J}$ from **query word vectors** $\boldsymbol{Q}$
- Attention weight : $a_t = softmax(S_{t:}) \in R^J$
- Each attended query vector : $\widetilde{U}_{:t} = \sum_j a_{tj} U_{:j}$ ($\widetilde{U}$: 2d-by-T matrix)
- t번째 Context에 연관이 높은 Query word vector에 가중치 부여

▶ Query-to-context Attention

- 어떤 Context의 단어들이 Query의 단어와 가장 유사성을 가져서 쿼리에 답변하기 위해 중요한지 알아냄
- $H \in R^{2d \times T}$ from the **context word vectors** $\boldsymbol{X}$
- Attention weight : $b = softmax(max_{col}(S)) \in R^T$
- Each attended context vector : $\tilde{h} = \sum_t b_t H_{:t} \in R^{2d}$
- Query의 단어들과 연관이 높은 Context word vector에 가중치 부여

- Contextual embeddings and the attention vectors are combined together to yield G

$$\mathbf{G}_{:t} = \boldsymbol{\beta}(\mathbf{H}_{:t}, \tilde{\mathbf{U}}_{:t}, \tilde{\mathbf{H}}_{:t}) \in \mathbb{R}^{d_{\mathbf{G}}}$$

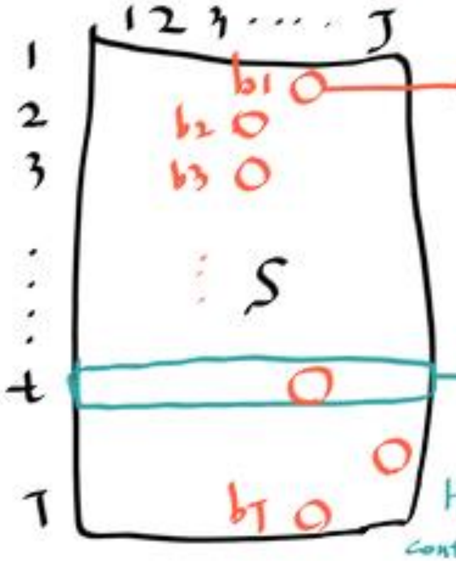
2 Model

Attention Flow Layer

Similarity matrix　context　Query

$$S_{tj} = \alpha(H_{:t}, U_{:j}) \in \mathbb{R}$$

where $\alpha(h, u) = W_{(S)}^{T} [\underbrace{h; u; h \circ u}_{\text{concatenate}}]$　$W_{(S)} \in \mathbb{R}^{6d}$, each of h, u, $h \circ u$: $2d$



query-to-context
context-to-query

→ 각 row에서 최대값 추출

$\vec{b} \in \mathbb{R}^t$

같은것을 T번 (복제)

$$\left[\cdots \middle| \sum_j b_j \cdot H^{(j)} \middle| \cdots \right] \overset{Q2C}{=} \tilde{H}_{2d \times T}$$

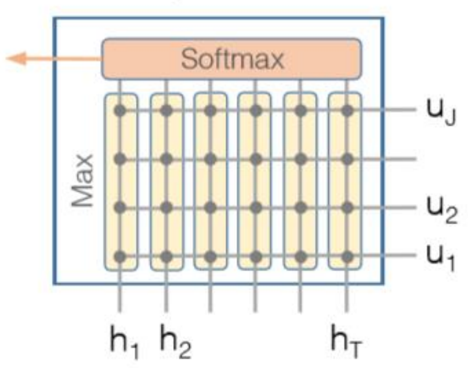
쿼리의 입장에서 컨텍스트 중 가장 중요한 단어들의 가중 합

$$\rightarrow \text{softmax}(S_t) = \vec{\alpha_t}$$

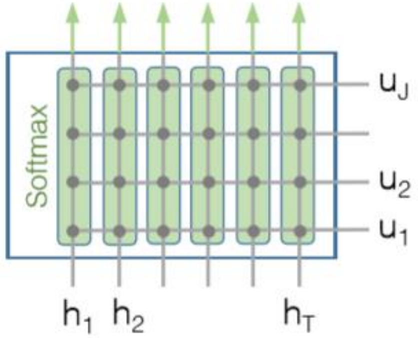
$$H_{\text{context}} = \left[h^1 \middle| \cdots h^t \cdots \middle| h^T \right] \overset{C2Q}{\Rightarrow} \left[\cdots \middle| \sum \vec{\alpha_{tj}} \cdot U^{(j)} \middle| \cdots \right] = \tilde{U}_{2d \times T}$$

이 단어와 가장 비슷한 쿼리

Query2Context



Context2Query

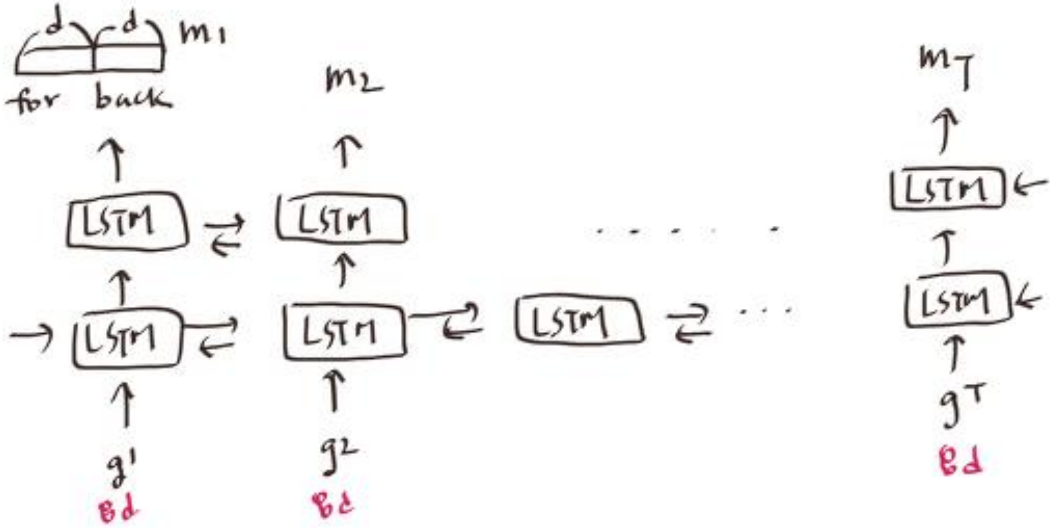


2 Model

Modeling Layer

▶ Modeling Layer

- Input : G (the query-aware representations of context words)
- Use two layers of bi-directional LSTM with the output size of d for each direction
- Output : a matrix $M \in R^{2d \times T}$
- Context가 Query를 알고 있다고 생각하고, Context word 사이의 상호작용을 찾아내는 Layer



$$G^{(t)} = [\underbrace{H^{(t)}}_{2d} ; \underbrace{\tilde{u}^{(t)}}_{2d} ; \underbrace{H \circ u^{(t)}}_{2d} ; \underbrace{H^{(t)} \circ \tilde{H}^{(t)}}_{2d}] \in \mathbb{R}^{8d}$$

t번째 시점

원래 컨텍스트

C2Q 결과

Q2C 결과

query-aware representation

2 Model

Output Layer

▶ Output Layer

- Query에 답변하는 단계로 application-specific하게 구현함

$$p^1_{start} = \mathrm{softmax}\left(\underset{10d}{W^T_{(p^1)}}\,[\underset{8d}{G};\underset{2d}{M}]\right)$$

$$p^2_{end} = \mathrm{softmax}\left(W^T_{(p^2)}\,[G;M^2]\right)$$

→ pass M to another bi-LSTM

Training

$$L(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\left[\log(p^1_{y^1_i}) + \log(p^2_{y^2_i})\right]$$

→ p^1의 y^1_i번째 성분.

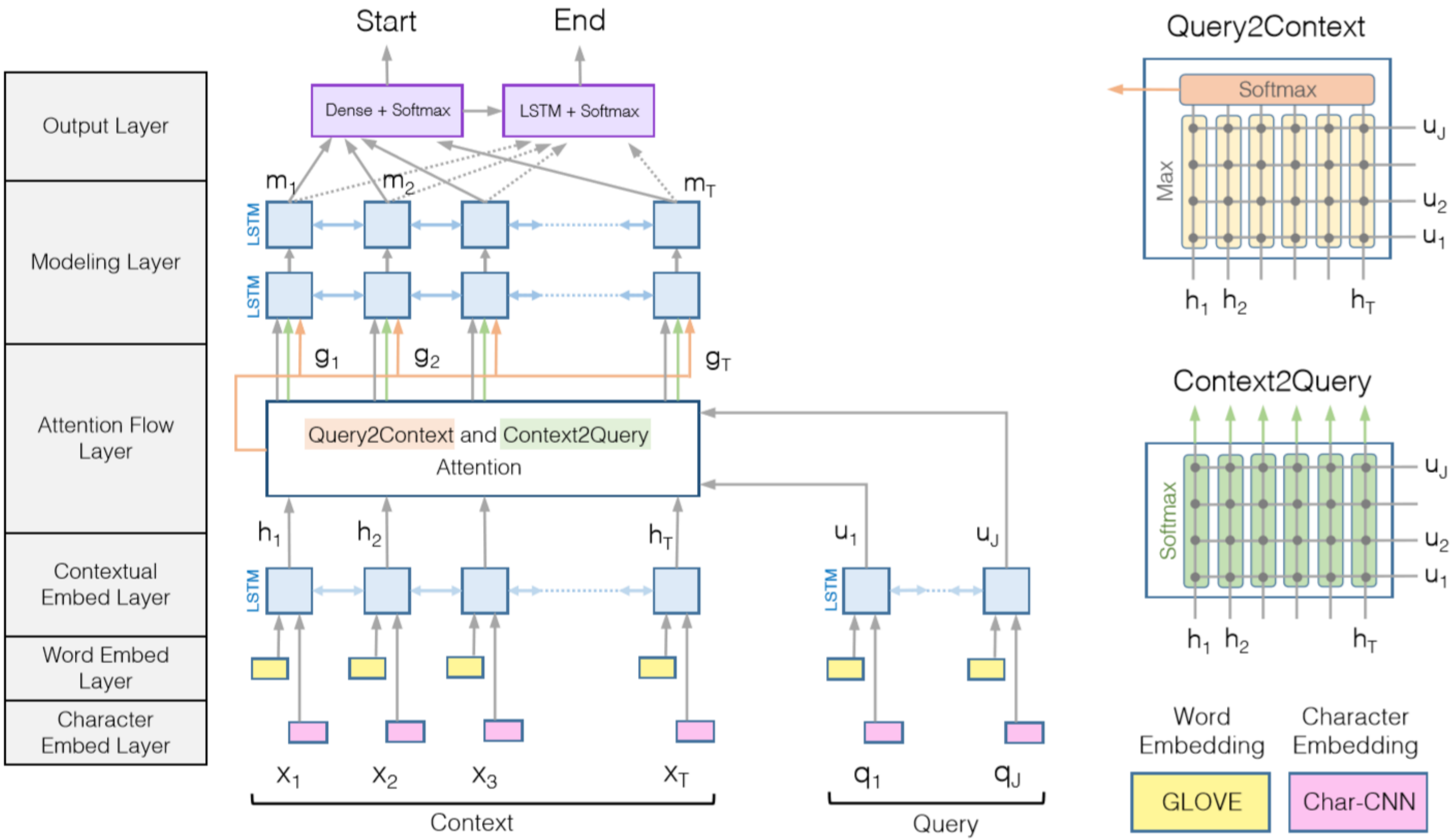
예측 start 값이 실제 타겟 y^1_i과 같을 확률.

Test

$$\max_{\substack{(k,l)\\ k \le l}} \left[p^1_k \times p^2_l\right] = (k^*, l^*)$$ 를 선택

2 Model

Bi-Directional Attention Flow Model



3 Experiments

Question Answering

▶ Dataset

- SQuAD : a MC dataset on a large set of Wikipedia articles with more than 100,000 questions

	Single Model EM	Single Model F1	Ensemble EM	Ensemble F1
Logistic Regression Baseline[a]	40.4	51.0	-	-
Dynamic Chunk Reader[b]	62.5	71.0	-	-
Fine-Grained Gating[c]	62.5	73.3	-	-
Match-LSTM[d]	64.7	73.7	67.9	77.0
Multi-Perspective Matching[e]	65.5	75.1	68.2	77.2
Dynamic Coattention Networks[f]	66.2	75.9	71.6	80.4
R-Net[g]	**68.4**	**77.5**	72.1	79.7
BiDAF (Ours)	68.0	77.3	**73.3**	**81.1**

(a) Results on the SQuAD test set

	EM	F1
No char embedding	65.0	75.4
No word embedding	55.5	66.8
No C2Q attention	57.2	67.7
No Q2C attention	63.6	73.7
Dynamic attention	63.5	73.6
BiDAF (single)	67.7	77.3
BiDAF (ensemble)	72.6	80.7

(b) Ablations on the SQuAD dev set

3 Experiments

Question Answering

Layer	Query	Closest words in the Context using cosine similarity
Word	When	when, When, After, after, He, he, But, but, before, Before
Contextual	When	When, when, 1945, 1991, 1971, 1967, 1990, 1972, 1965, 1953
Word	Where	Where, where, It, IT, it, they, They, that, That, city
Contextual	Where	where, Where, Rotterdam, area, Nearby, location, outside, Area, across, locations
Word	Who	Who, who, He, he, had, have, she, She, They, they
Contextual	Who	who, whose, whom, Guiscard, person, John, Thomas, families, Elway, Louis
Word	city	City, city, town, Town, Capital, capital, district, cities, province, Downtown
Contextual	city	city, City, Angeles, Paris, Prague, Chicago, Port, Pittsburgh, London, Manhattan
Word	January	July, December, June, October, January, September, February, April, November, March
Contextual	January	January, March, December, August, December, July, July, July, March, December
Word	Seahawks	Seahawks, Broncos, 49ers, Ravens, Chargers, Steelers, quarterback, Vikings, Colts, NFL
Contextual	Seahawks	Seahawks, Broncos, Panthers, Vikings, Packers, Ravens, Patriots, Falcons, Steelers, Chargers
Word	date	date, dates, until, Until, June, July, Year, year, December, deadline
Contextual	date	date, dates, December, July, January, October, June, November, March, February

Table 2: Closest context words to a given query word, using a cosine similarity metric computed in the Word Embedding feature space and the Phrase Embedding feature space.

3 Experiments

Question Answering

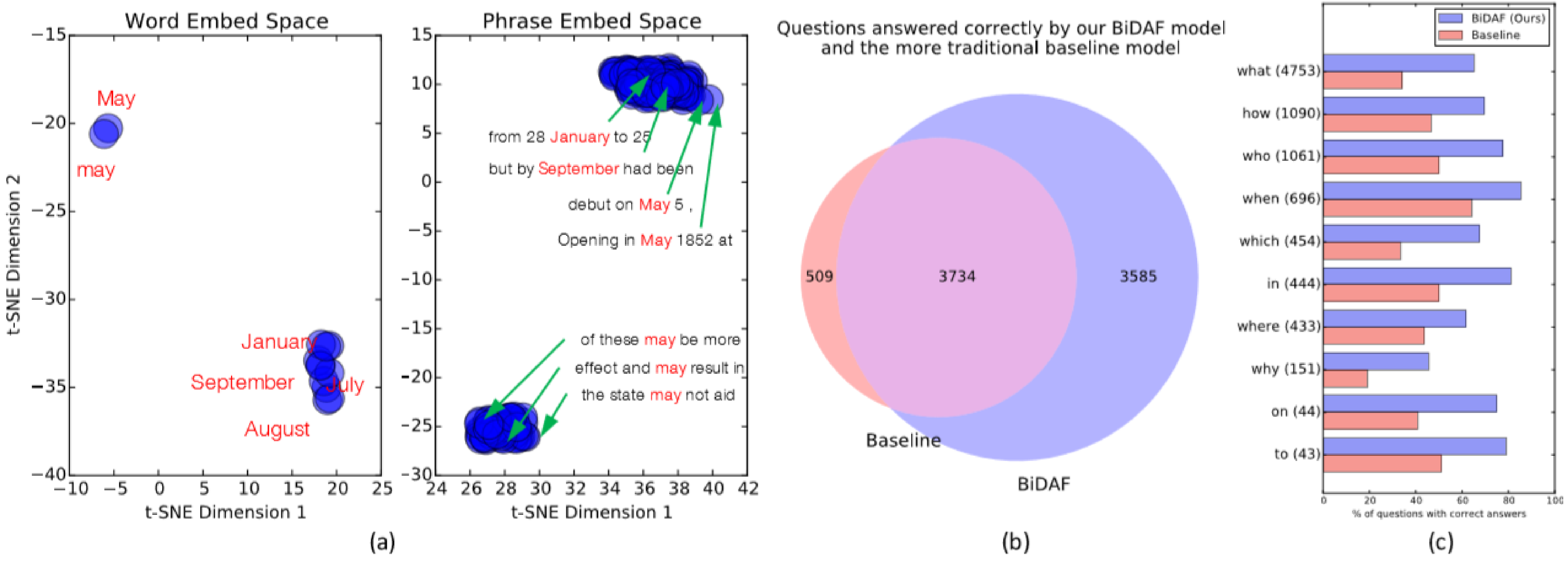


Figure 2: (a) t-SNE visualizations of the *months* names embedded in the two feature spaces. The contextual embedding layer is able to distinguish the two usages of the word *May* using context from the surrounding text. (b) Venn diagram of the questions answered correctly by our model and the *more traditional* baseline (Rajpurkar et al., 2016). (c) Correctly answered questions broken down by the 10 most frequent first words in the question.

3 Experiments

Question Answering

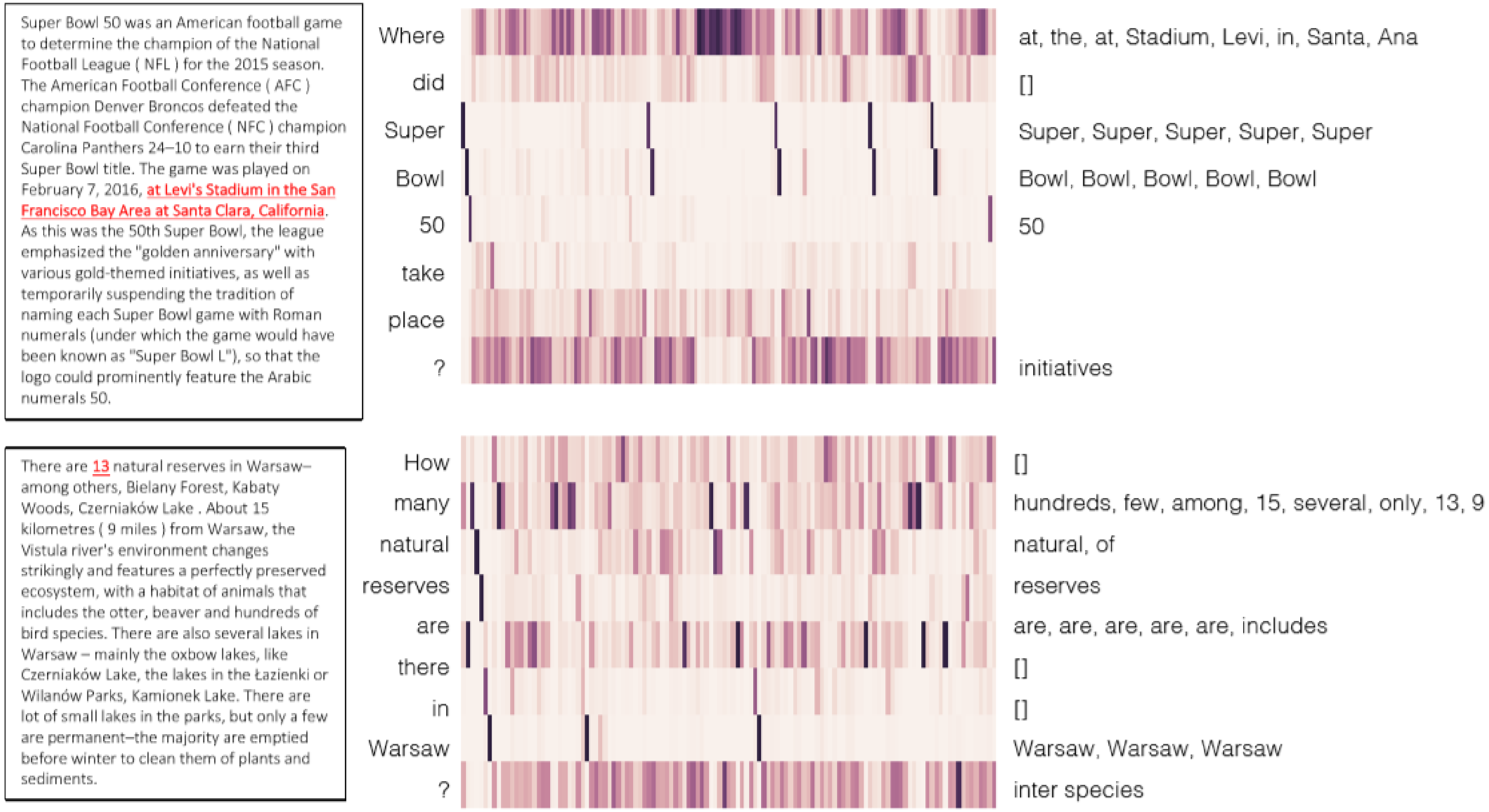


Figure 3: Attention matrices for question-context tuples. The left palette shows the context paragraph (correct answer in red and underlined), the middle palette shows the attention matrix (each row is a question word, each column is a context word), and the right palette shows the top attention points for each question word, above a threshold.

3 Experiments

Cloze Test

▶ Dataset

- CNN/Daily Mail datasets : a massive Cloze-style comprehension dataset
- Each example has a news article and an incomplete sentence extracted from the human-written summary of the article
- Predict the correct missing word

3 Experiments

Cloze Test

	CNN		DailyMail	
	val	test	val	test
Attentive Reader (Hermann et al., 2015)	61.6	63.0	70.5	69.0
MemNN (Hill et al., 2016)	63.4	6.8	-	-
AS Reader (Kadlec et al., 2016)	68.6	69.5	75.0	73.9
DER Network (Kobayashi et al., 2016)	71.3	72.9	-	-
Iterative Attention (Sordoni et al., 2016)	72.6	73.3	-	-
EpiReader (Trischler et al., 2016)	73.4	74.0	-	-
Stanford AR (Chen et al., 2016)	73.8	73.6	77.6	76.6
GAReader (Dhingra et al., 2016)	73.0	73.8	76.7	75.7
AoA Reader (Cui et al., 2016)	73.1	74.4	-	-
ReasoNet (Shen et al., 2016)	72.9	74.7	77.6	76.6
BiDAF (Ours)	**76.3**	**76.9**	**80.3**	**79.6**
MemNN* (Hill et al., 2016)	66.2	69.4	-	-
ASReader* (Kadlec et al., 2016)	73.9	75.4	78.7	77.7
Iterative Attention* (Sordoni et al., 2016)	74.5	75.7	-	-
GA Reader* (Dhingra et al., 2016)	76.4	77.4	79.1	78.1
Stanford AR* (Chen et al., 2016)	77.2	77.6	80.2	79.2

Table 3: Results on CNN/DailyMail datasets. We also include the results of previous ensemble methods (marked with *) for completeness.