BiDAF:
BI-DIRECTIONAL ATTENTION FLOW FOR MACHINE COMPREHENSION (2016)

Jang woojin

0. History of QA

- Question Answering
  - QA vs IR system
    - QA = Query (specific) → answer for specific question
      - Ex. 한국의 수도는? → 서울
    - IR = Query(general) → Document list
      - Ex. 다이어트에 좋은 음식은? → 다이어트에 좋은 음식과 관련된 문서, 영상 등이 리스트로 검색결과로 나옴.

  - 이 논문이 나올 때 상황
    - 2015년 쯤에 MRC dataset이 공개되면서 MRC에 관심이 많아짐
    - Attention이 등장하면서 MRC에도 적용해보고자 하는 시도가 이루어짐
    - Stanford attentive reader → BiDAF → Stanford attentive reader2 (DrQA)

0. History of QA

- Types of questions
  1) Factoid type questions [what, which, when , who, how]
     - ex. Q) which city has most population in KOREA? A) SEOUL
  2) List type questions [list of answers]
     - ex. Q) what are the cities in KOREA? A) seoul, busan …..
  3) Confirmation questions [yes or no]
     - ex. Q) Did WHO declare CORONA virus as pandemic? A) yes
  4) Casual questions [Why or How]
     - ex. Q) Why are u hungry? A) B/c I am on a diet. (description)
  5) Hypothetical questions [no specific answers]
     - Q) what would happen if you won the lottery? A) ☺
  6) Complex questions
     - what are the reasons of financial crisis ? A) …

A survey on type of question answering system

1. Abstract

- MC (Machine Comprehension == MRC)
  - answering a query about a given context paragraph
  - query 와 context 간에 복잡한 interaction을 이해해야함.

- Attention이 등장하면서 MC 분야에서도 적용되기 시작함
  - Uni-directional attention을 보통 썼다.(stanford attentive reader)
  - context 중에서 일부만 attention하여 그 정보를 fixed-size vector로 표현하는 방식

- Here, BiDAF : Bi-directional Attention Flow
  - multi-stage hierarchical process
  - to obtain a query-aware context representation without early summarization
  - Query to Context 뿐만 아니라 Context to Query도 같이 고려하는 구조를 통해 성능 향상
  - SQuAD 와 CNN daily mail에서 SOTA 달성함

2. Introduction

1. BiDAF
    1. hierarchical multi-stage architecture (6 layers)
    2. Context 세분화
    3. Char/ word/ contextual level embedding

2. 과거 attention 사용 모델과 BiDAF 차이점
    1. Fixed-sized vector로 표현하지 않고, 매 타임 스텝마다 vector를 구하고, 후속 layer에 정보가 전달되는 방식
       (not summarize the query and context into single feature vectors)
    2. Memory-less attention : 기존에는 Bahdanau attention → current time step에서 계산

    3. Bi-directional
       - 기존에는 query에서 context 방향으로만 계산
       - BiDAF는 query to context & context to query 양방향을 고려.
       → context와 query간에 충분한 정보 교환 가능하도록.

3. Model

- BiDAF 6개의 layer로 구성

1. Character layer:
   char-CNN
2. Word layer :
   pre-trained GLOVE model
3. Contextual layer:
   utilize contextual cues
4. Attention layer:
   produces a set of query-aware feature vectors for each word in the context.
5. Modeling layer:
   bi-LSTM
6. Output layer:
   find answer span.

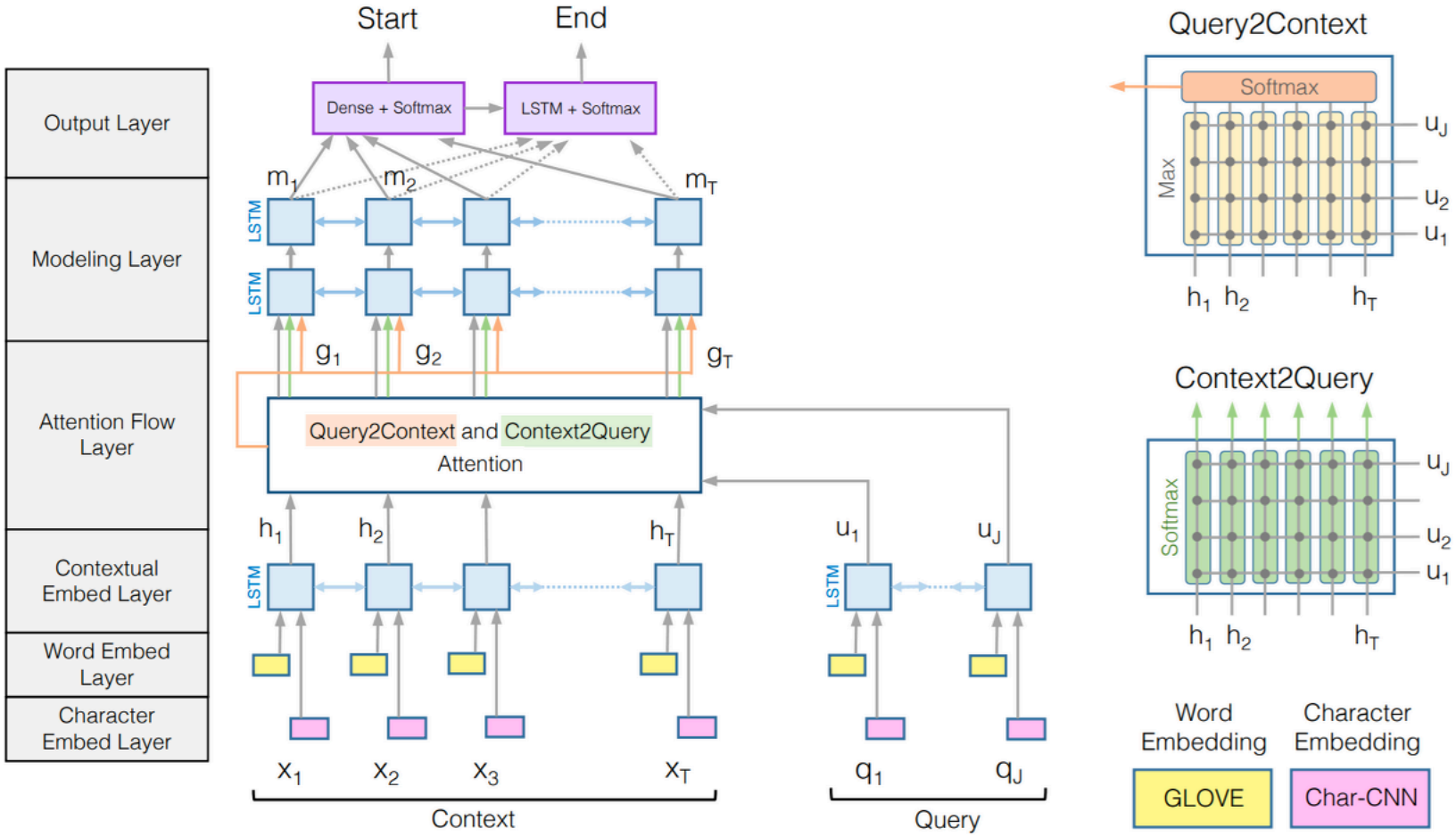


Figure 1: BiDirectional Attention Flow Model *(best viewed in color)*

3. Model

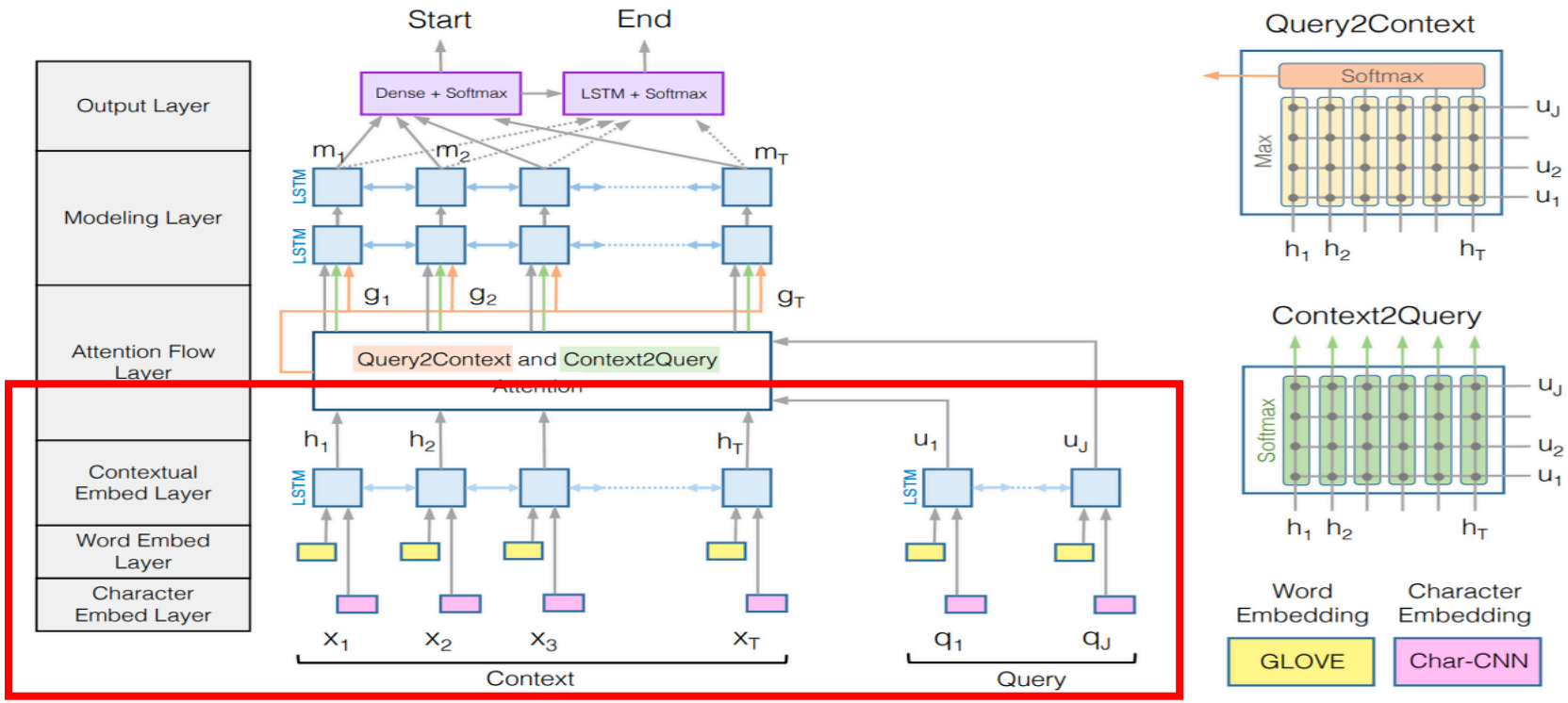


Figure 1: BiDirectional Attention Flow Model *(best viewed in color)*

1. Concatenate [character vector , word vector] (both context and query)
2. 2-layer Highway Network
3. Contextual layer → bi-LSTM 결과는forward + backward를 concat

→ 3번째 contextual layer까지는 context와 query 각각을 계산하여 attention flow layer 입력으로 들어감.

3. Model

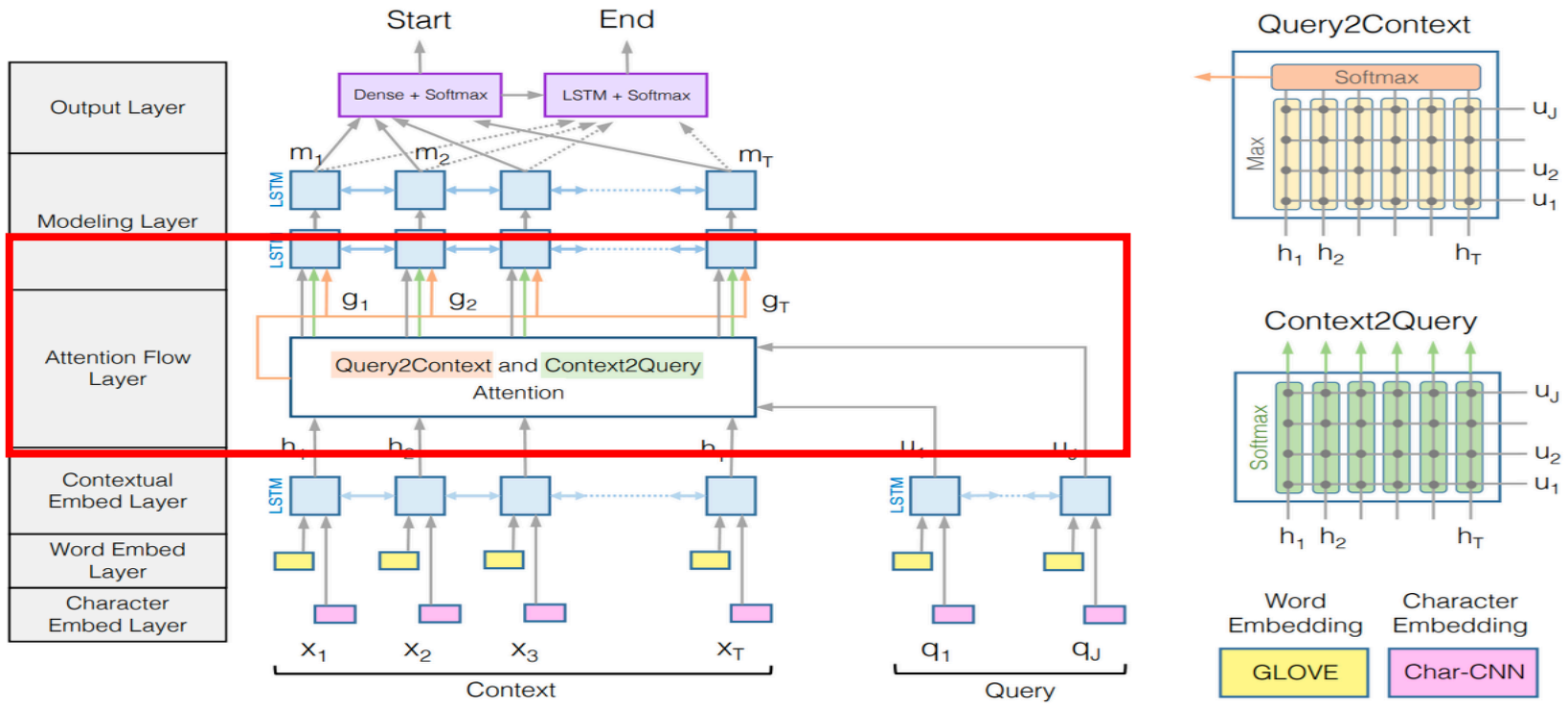
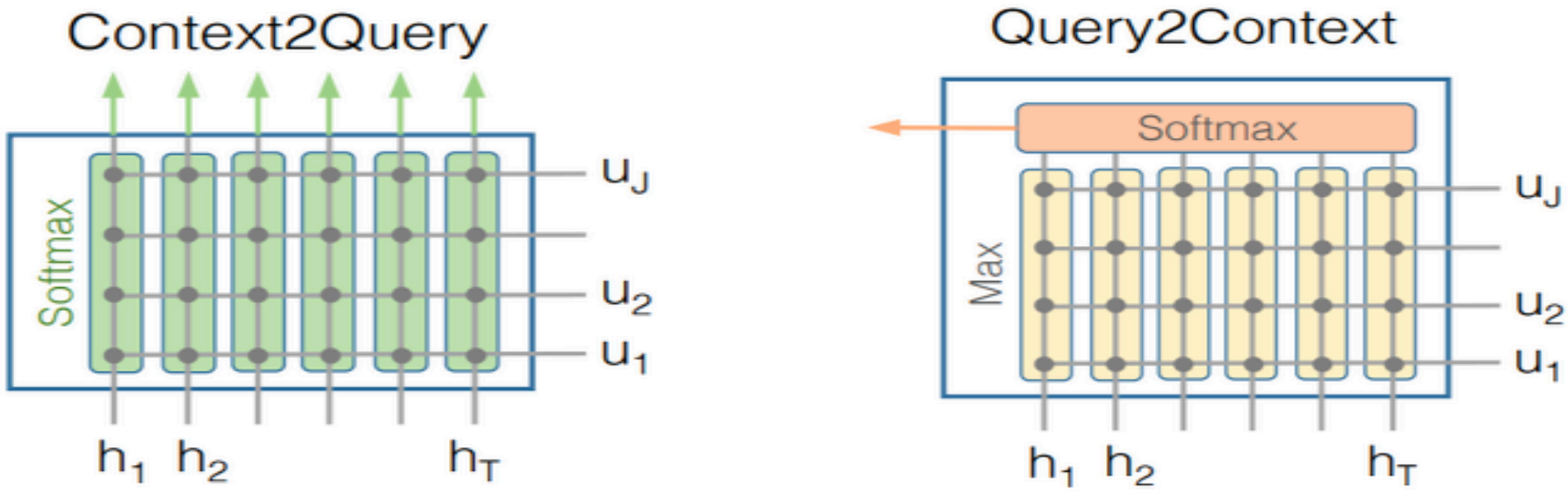


Figure 1: BiDirectional Attention Flow Model *(best viewed in color)*

- Attention Flow Layer
  - Use Shared similarity matrix for attention (S ∈ $R^{T\ X\ J}$)
  - H: the output of contextual layers from Context ($h_t = H$의 t번째)
  - U: the output of contextual layers from Query(u_t = U의 t번째)

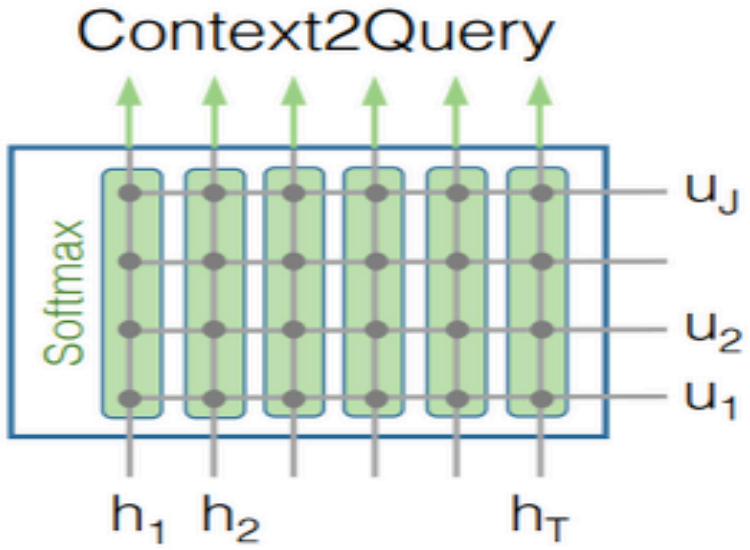
3. Model

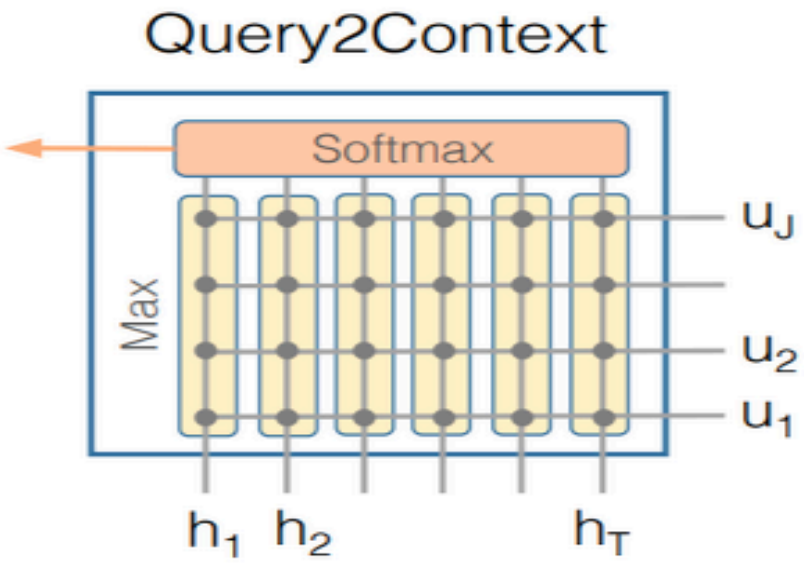


- Attention Flow Layer
  - Use Shared similarity matrix for attention (S ∈ $R^{T\,X\,J}$)
  - Context2Query: which query words are most relevant to each context word
  - Query2Context : which context words have the closest similarity to one of the query words
  - Context는 전체 document가 입력으로 들어가는 형태
  - T 번째 context word 와 J 번째 query word간 유사도 계산

3. Model

- Use Shared similarity matrix for attention (S ∈ $R^{T\ X\ J}$)

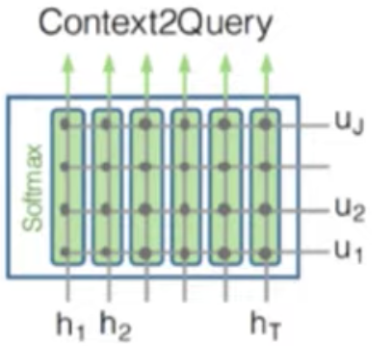


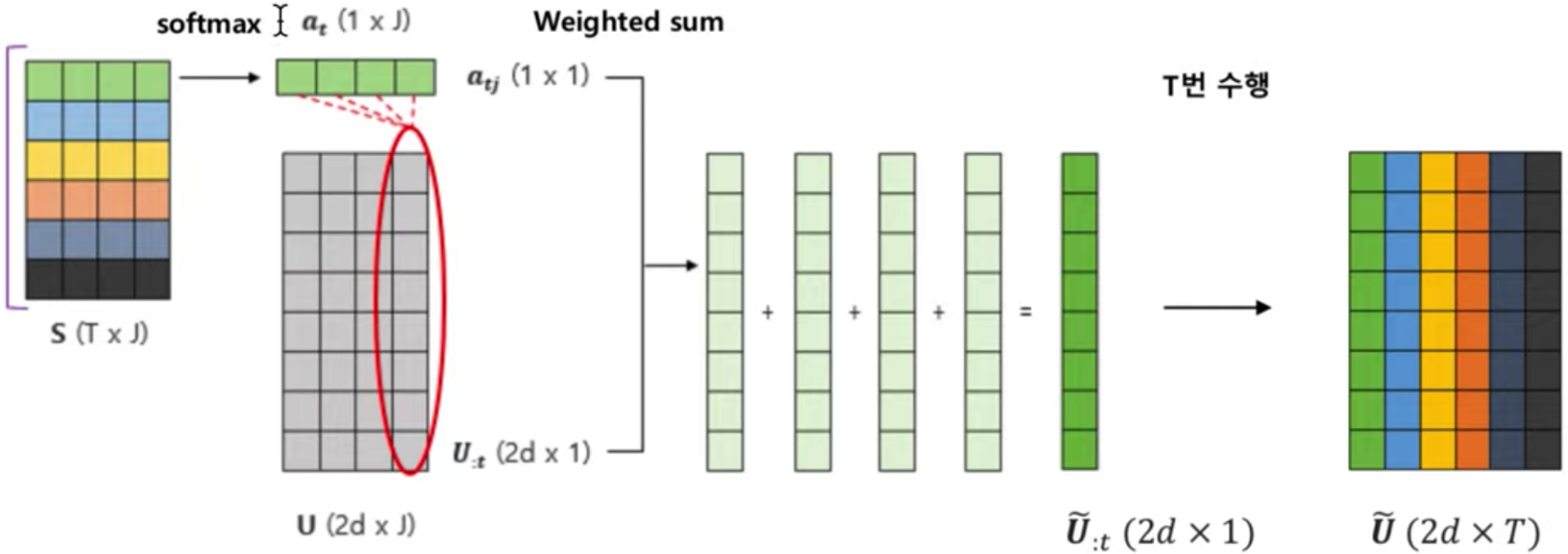


- $S_{tj} = \alpha\left(H_{:t}, U_{:j}\right) \in \mathbb{R}$
- $\alpha(h,u) = w_{(S)}^{\top}[h; u; h \circ u]$, where $w_{(S)} \in \mathbb{R}^{6d}$

3. Model

- Context2Query：which query words are most relevant to each context word

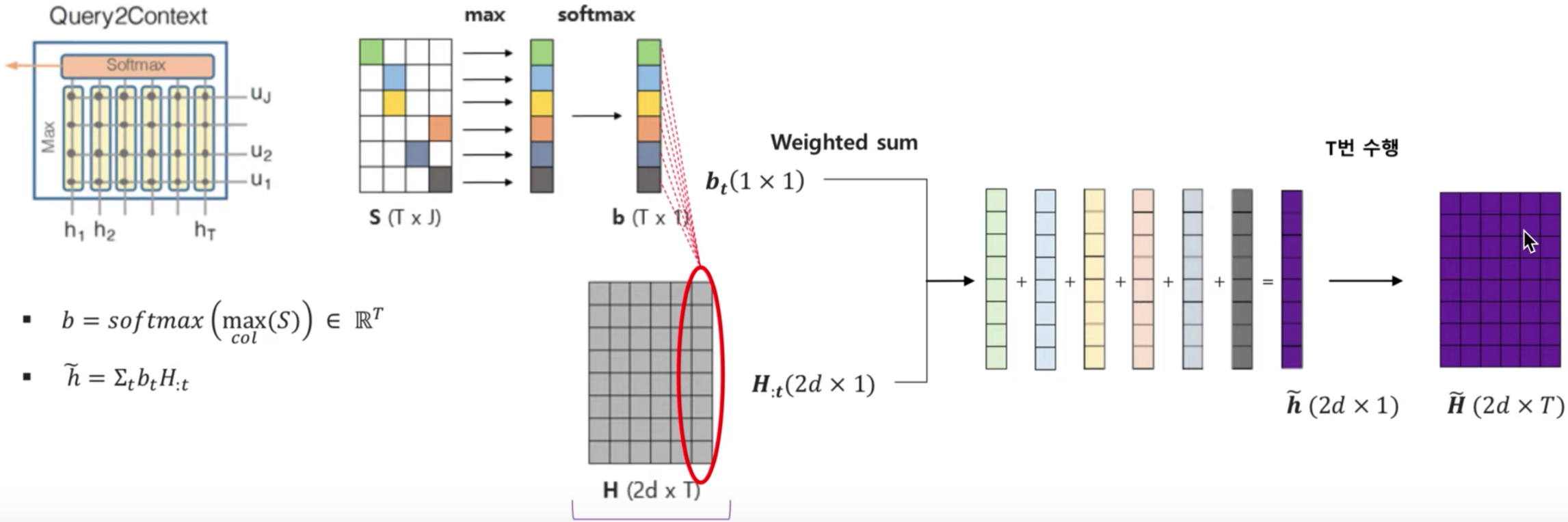




- $a_t = softmax(S_{t:}) \in \mathbb{R}^J$
- $\widetilde{U_{:t}} = \Sigma_j a_{tj} U_{:j}$

3. Model

- Query2Context : which context words have the closest similarity to one of the query words



- $b = softmax\left(\max_{col}(S)\right) \in \mathbb{R}^T$
- $\tilde{h} = \Sigma_t b_t H_{:t}$

$\tilde{h}$ indicates the weighted sum of the most important words in the context with respect to the query.

3. Model

- Contextual layer + Attention vectors

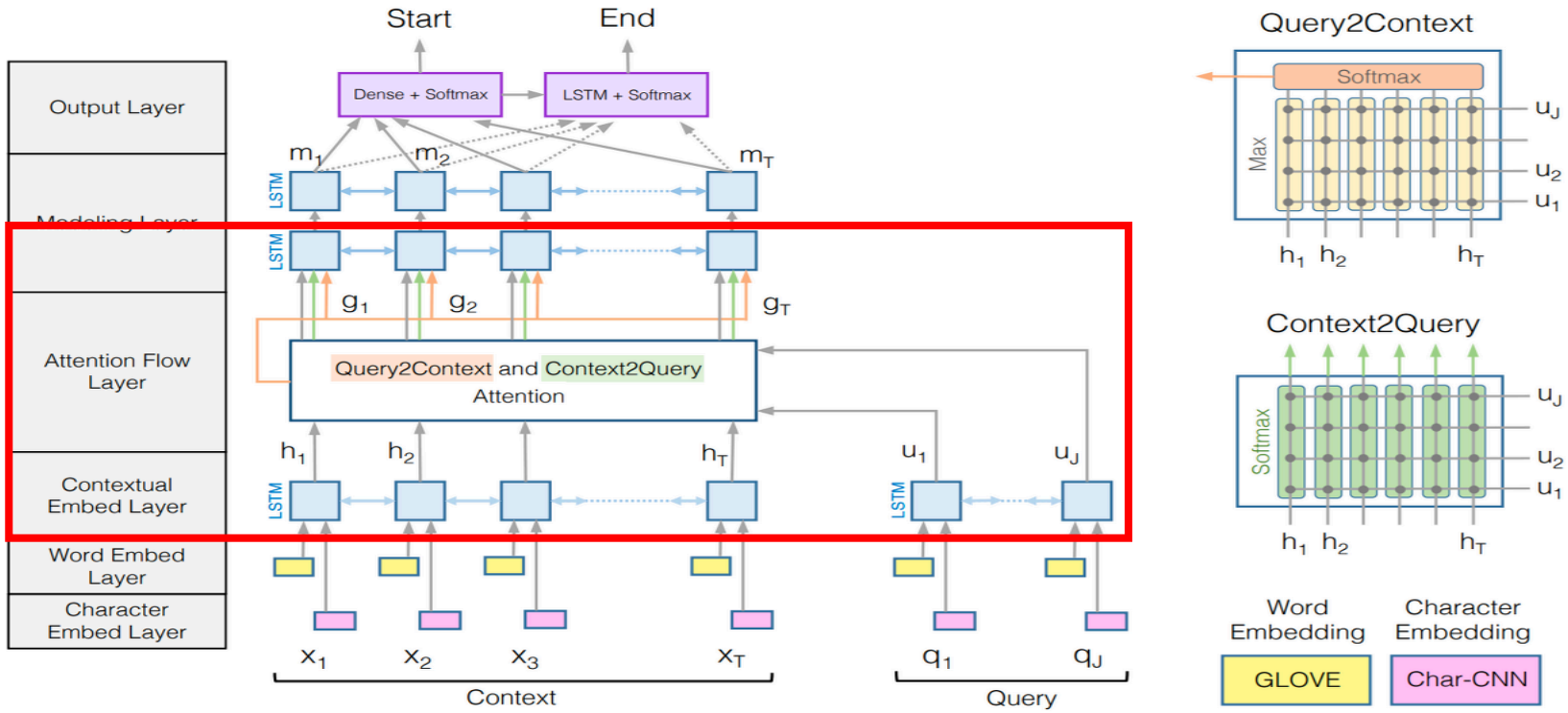


Figure 1: BiDirectional Attention Flow Model *(best viewed in color)*

- $G_{:t} = \beta(H_{:t}, \widetilde{U}_{:t}, \widetilde{H}_{:t}) \in \mathbb{R}^T$
- $\beta(h, \tilde{u}, \tilde{h}) = [h; \tilde{u}, h \circ \tilde{u}; h \circ \tilde{h}] \in \mathbb{R}^{8d \times T}$

3. Model

- Modeling layer

Input: G / output : M (2d x T)

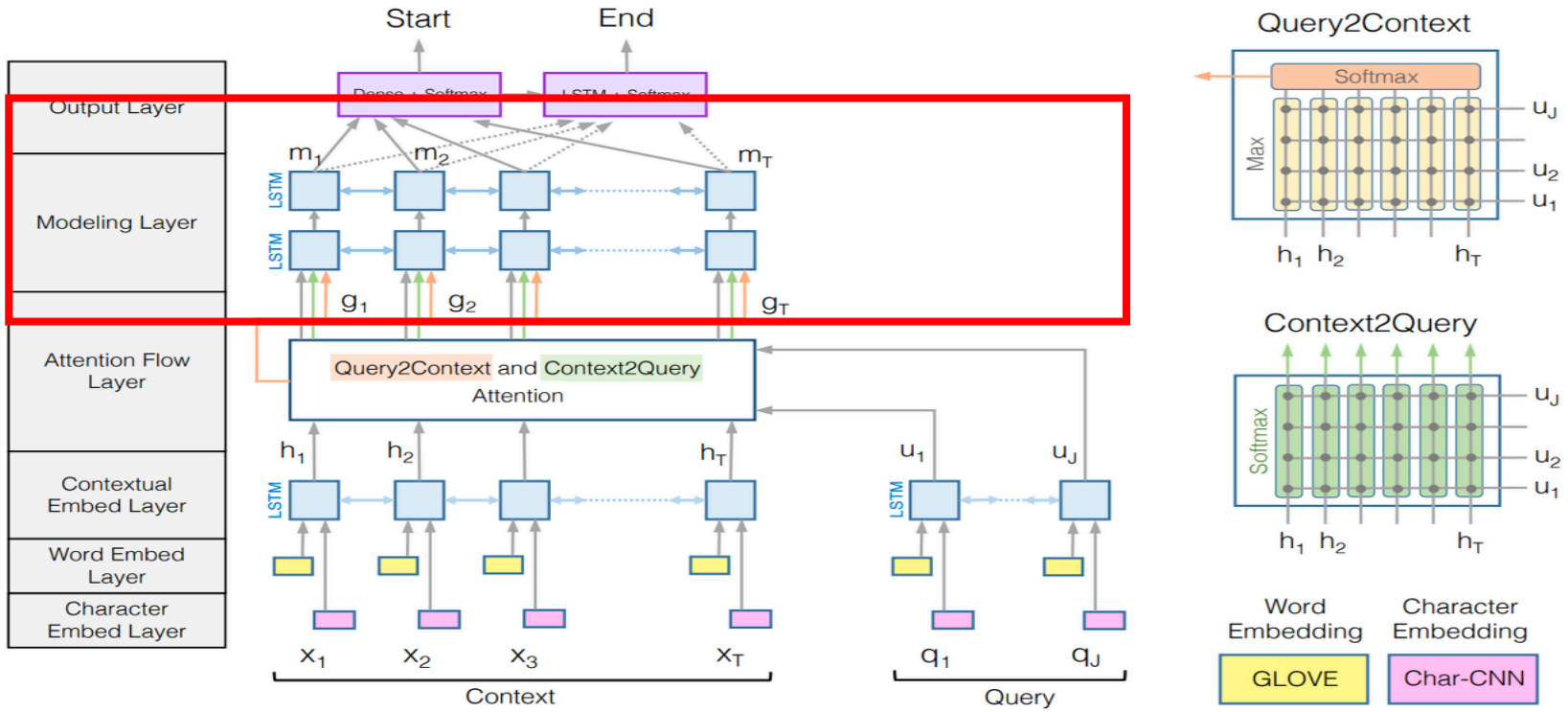


Figure 1: BiDirectional Attention Flow Model *(best viewed in color)*

- 2 layer bi-directional LSTM
- Query 와 context 간에 interaction 학습

3. Model

- Output layer

Input: M / output : predict answer span

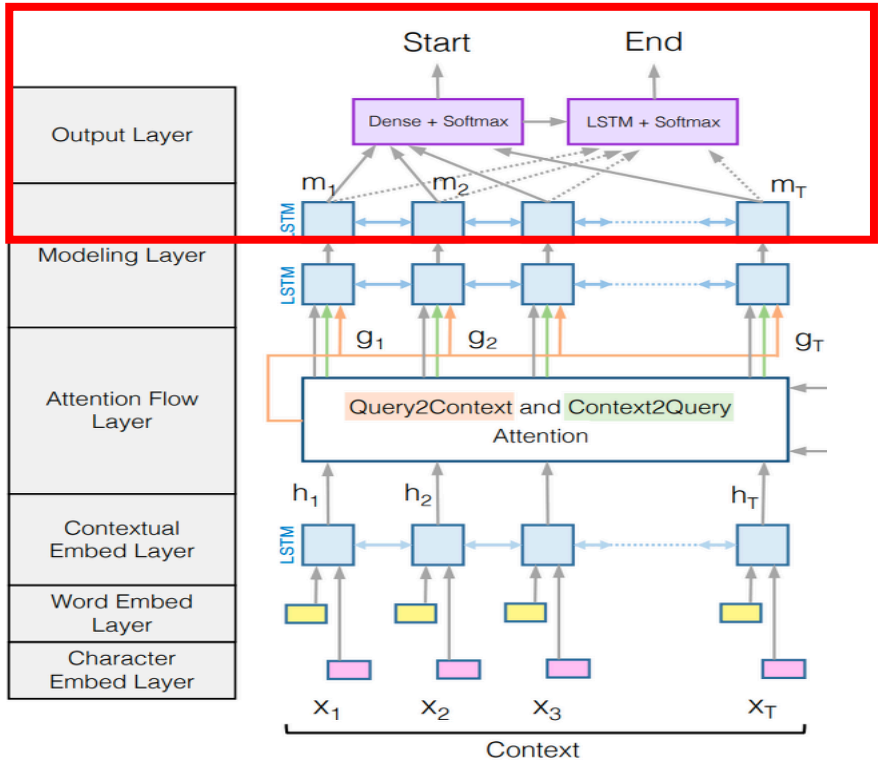


Figure 1: BiDirectional Attention Fl

- Predict start indices

$$\mathbf{p}^1 = \text{softmax}(\mathbf{w}_{(\mathbf{p}^1)}^\top [\mathbf{G}; \mathbf{M}]),$$

- Predict end indices

$$\mathbf{p}^2 = \text{softmax}(\mathbf{w}_{(\mathbf{p}^2)}^\top [\mathbf{G}; \mathbf{M}^2])$$

- Task에 따라 output layer만 변경해주면 여러가지 specific task를 해결 가능.

4. Question Answering Experiments

	Single Model		Ensemble	
	EM	F1	EM	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

4. Question Answering Experiments

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.

4. Question Answering Experiments

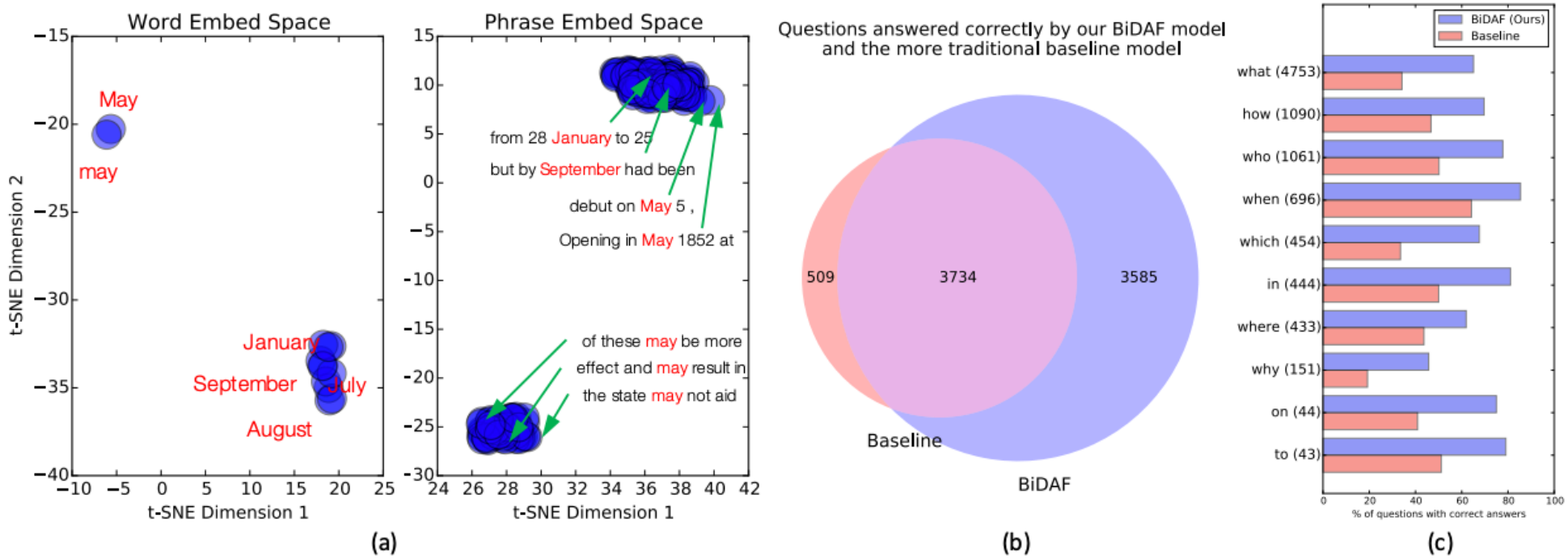


Figure 2: (a) t-SNE visualizations of the *months* names embedded in the two feature spaces. The contextual embedding layer is able to distinguish the two usages of the word *May* using context from the surrounding text. (b) Venn diagram of the questions answered correctly by our model and the *more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

4. Question Answering Experiments

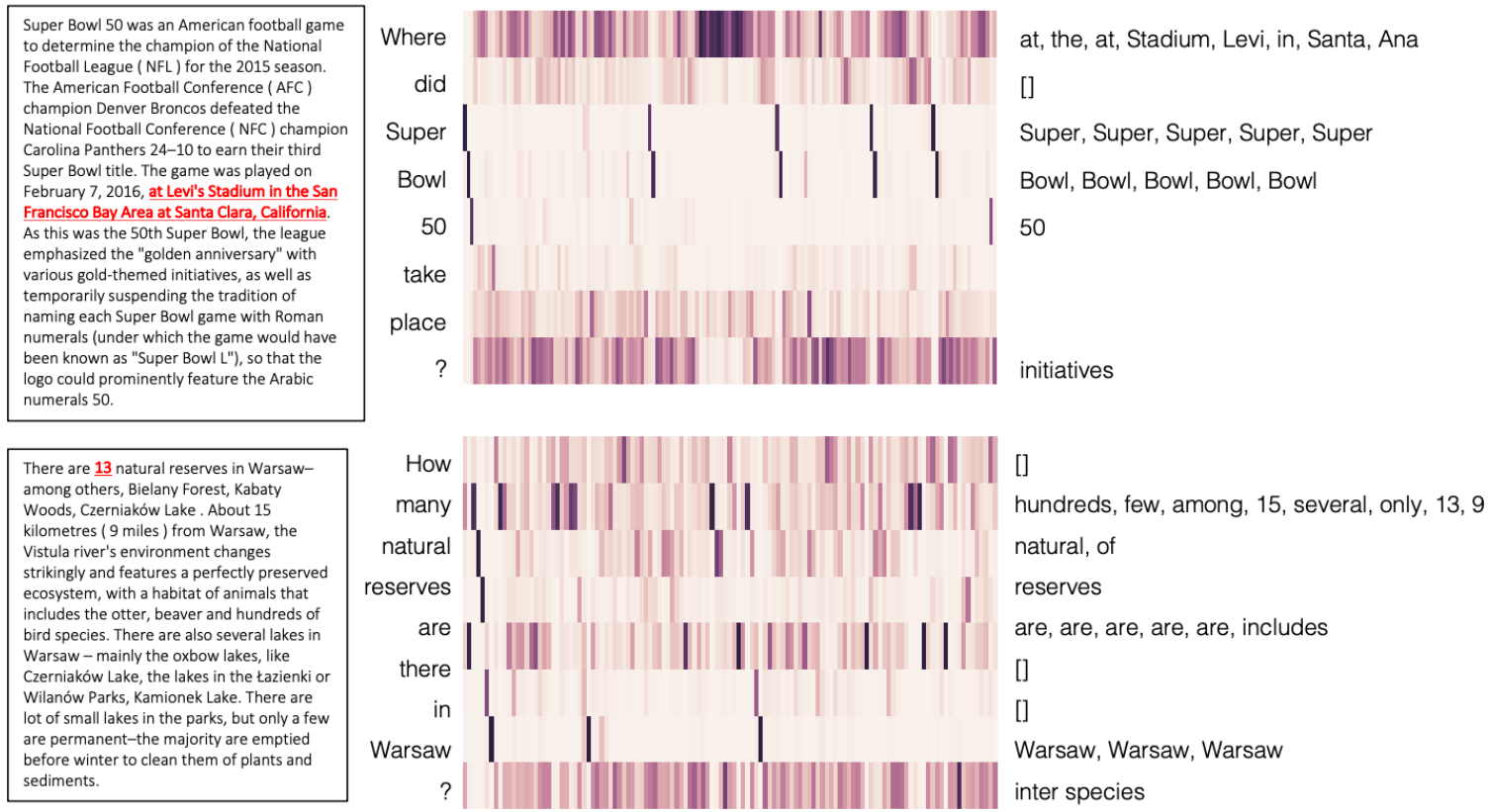


Figure 3: Attention matrices for question-context tuples. The left palette shows the context paragraph (correct answer in red and underlined), the middle palette shows the attention matrix (each row is a question word, each column is a context word), and the right palette shows the top attention points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

5. Cloze Test Experiments

	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.