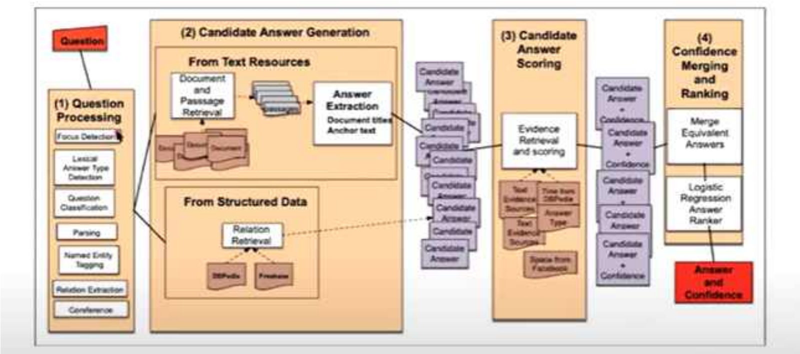
Lec 11. Question Answering(Danqu Chen)

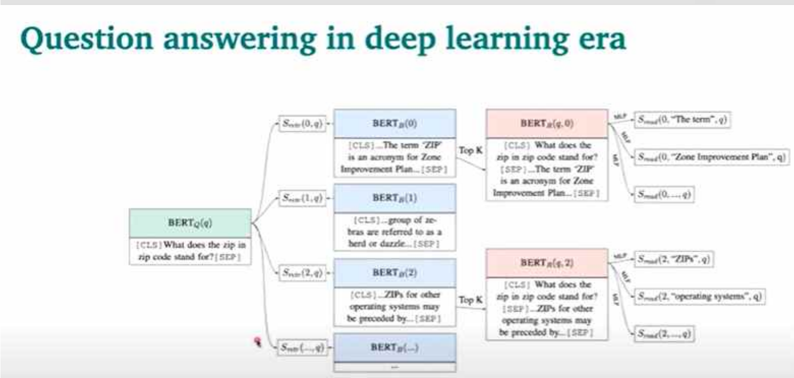
1. What is question answering?

- the goal: to build systems that automatically answer questions posed by human in a natural language.

- information source: A text passage, all Web documents, knowledge bases, images..
- Question type: Facoid vs. non-facttoid, open-domain vs. coled-domain, simple vs. compositional, ..
- Answer type: A short segment of text, a paragraph, a list, yes/no, ...

eg. IBM Watson beated Jeopardy champions





- unstructured text

2. Reading comprehension
: comprehen a passage of text and answer questions about its content(P,Q) ->A
-> an important testbed for evaluating how well computer systems sunderstand human language
-> Many other NLP tasks can be reduced tot a reading comprehension problem: information extraction, semantic role labeling

eg. SQuAD (compare the predicted answer to each gold answer)

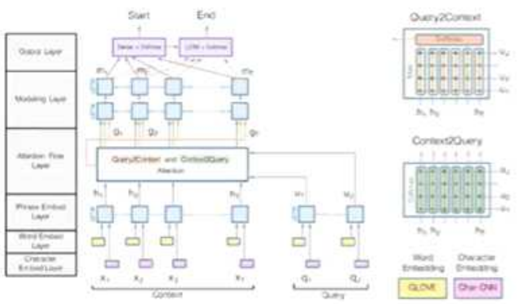
-Neural models for reading comprehension
-> How can we build a model to solve SQuAD?

- Problem formulation
  - Input: $C = (c_1, c_2, \ldots, c_N), Q = (q_1, q_2, \ldots, q_M), c_i, q_i \in V$
  - Output: $1 \leq$ start $\leq$ end $\leq N$

N~100, M ~15
answer is a span in the passage

=> A family of LSTM-based models with attention
=> Fine-tuning BERT-like models for reading comprehension

LSTM-based vs BERT models



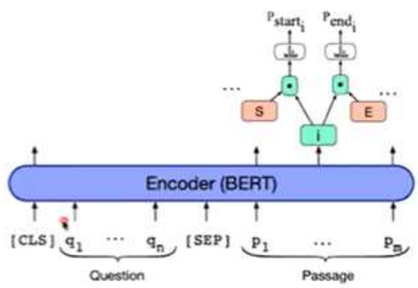
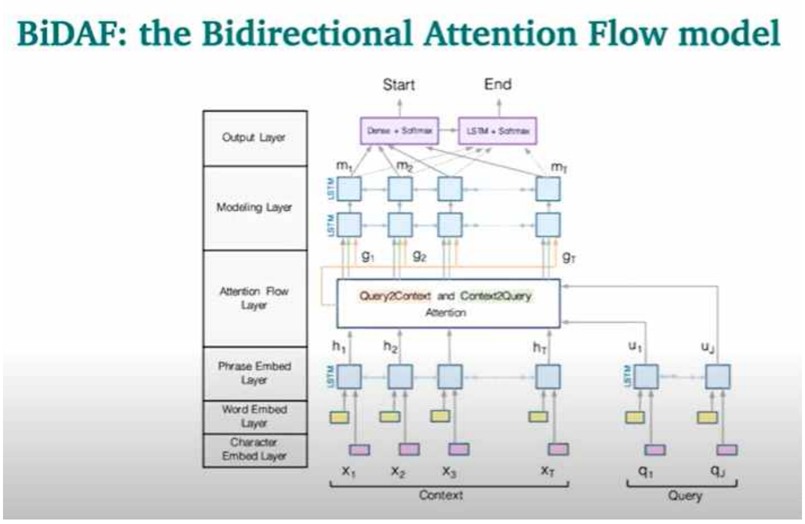
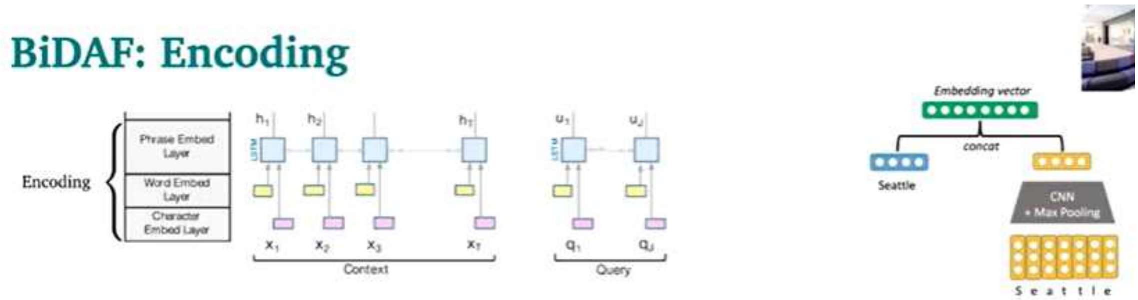


Image credit: (Seo et al, 2017)

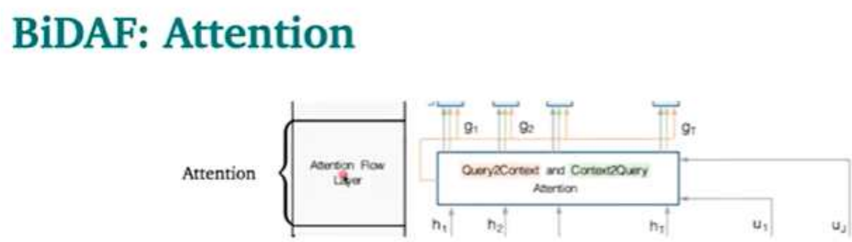
Image credit: J & M, edition 3

- which words in the source are most relevant to the current target word?
- don't need an autoregressive decoder to generate the target sentence word-by-word. intead, we just need to train two classifiers to predict the start and end positions of the answer.





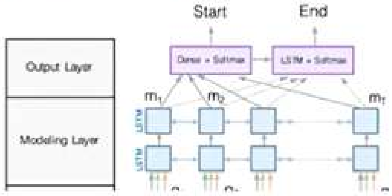
- Use a concatenation of word embeddings(GloVe) and charcter embedding(CNNs over character embeddings) for each word in context and query.
- Use two bidirectional LSTMs separately to produce contexttual embeddings for both context and query.



- Context-to-query attention: For each context word, choose the most relevant words from the query words.
- Query-to-context attention: choose the context words that are most relevant to one of query words.
- 1. compute a similarity score for every pair of (c,q):
- 2. context-to-query attention(which question words are more relevant to c):
- 3. Query-to-attention(which context words are relevant to some question words):

The final output is
$\mathbf{g}_i = [\mathbf{c}_i; \mathbf{a}_i; \mathbf{c}_i \odot \mathbf{a}_i; \mathbf{c}_i \odot \mathbf{b}_i] \in \mathbb{R}^{8H}$

BiDAF: Modeling and output layers



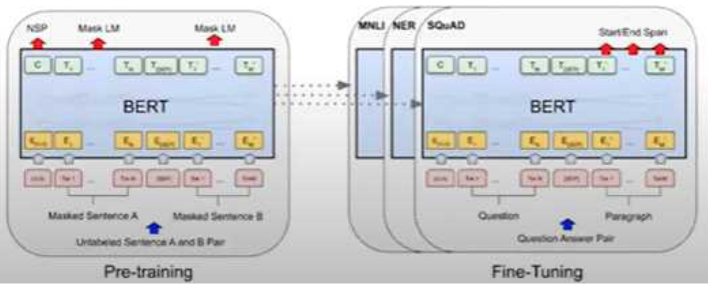
Modeling layer: pass $\mathbf{g}_i$ to another two layers of **bi-directional** LSTMs.

- Attention layer is modeling interactions between query and context
- Modeling layer is modeling interactions within context words

$$\mathbf{m}_i = \text{BiLSTM}(\mathbf{g}_i) \in \mathbb{R}^{2H}$$

BERT for reading comprehension

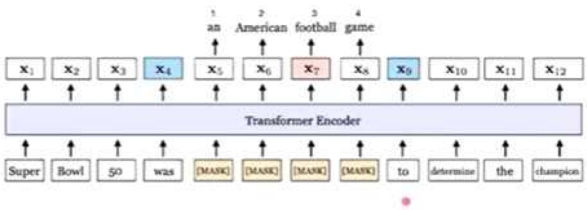
- BERT is a deep bidirectional Transformer encoder pre-trained on large amounts of text (Wikipedia + BooksCorpus)
- BERT is pre-trained on two training objectives:
  - Masked language model (MLM)
  - Next sentence prediction (NSP)
- $\text{BERT}_{\text{base}}$ has 12 layers and 110M parameters, $\text{BERT}_{\text{large}}$ has 24 layers and 330M parameters



Can we design better pre-training objectives?

The answer is yes!

$$\mathcal{L}(\text{football}) = \mathcal{L}_{\text{MLM}}(\text{football}) + \mathcal{L}_{\text{SBO}}(\text{football})$$
$$= -\log P(\text{football} \mid \mathbf{x}_7) - \log P(\text{football} \mid \mathbf{x}_4, \mathbf{x}_9, \mathbf{p}_3)$$



Two ideas:

1) masking contiguous spans of words instead of 15% random words

2) using the two end points of span to predict all the masked words in between = compressing the information of a span into its two endpoints

$$\mathbf{y}_i = f(\mathbf{x}_{s-1}, \mathbf{x}_{e+1}, \mathbf{p}_{i-s+1})$$

- How to answer questions over a single passage of text

3. Open-domain (textual) question answering
- How to answer questions over a large collect of documents

Retriever-reader framework

- Input: a large collection of documents $\mathcal{D} = D_1, D_2, \ldots, D_N$ and Q
- Output: an answer string A

- Retriever: $f(\mathcal{D}, Q) \longrightarrow P_1, \ldots, P_K$ K is pre-defined (e.g., 100)
- Reader: $g(Q, \{P_1, \ldots, P_K\}) \longrightarrow A$ A reading comprehension problem!

Large language models can do open-domain QA well

- … without an explicit retriever stage