论文阅读笔记

Step2

MF1833063, 史鹏, spwannasing@gmail.com

2019 年 3 月 23 日

1 Pointer Networks

我们引入了一种新的神经网络结构来学习输出序列的条件概率，其中的元素是与输入序列中的位置对应的离散序列。这类问题不能通过现在的序列对序列和NTM（Neural Turing Mechines）解决，因为在每个步骤的输出的目标数量取决于输入的长度，这是可变的。诸如排列可变长度序列和各种组合优化问题的问题都属于这个类型。我们的模型采用最近提出的神经注意机制解决了可变大小输出字典的问题。它不同于前面提出的，不是使用注意力机制/将编码器的隐层单元混合在每个解码阶段的上下文向量中，而使用注意力机制作为指针来选择输入序列的一个成员来作为输出序列。我们把这种体系结构叫做指针网络。Ptr-Nets不仅改进了序列到序列（seq-to-seq）的输入注意力机制，而且允许我们将模型泛化去处理可变字典。

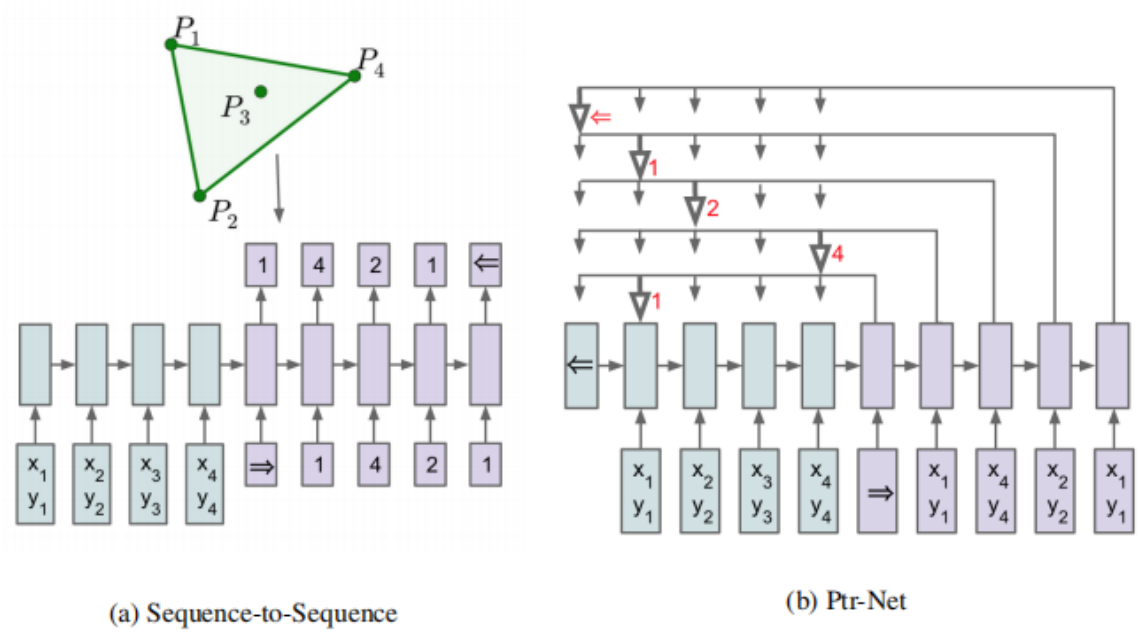


图 1: seq2seq与Prt-Net对比

1. seq2seq Model

$$p(C^P|P;\theta) = \prod_{i=1}^{m(P)} p(C_i|C_1,...,C_{i-1},P;\theta)$$

P是input向量，C^P是输出向量，问题就是需要预先设定输出的长度，当输出长度根据输入的不同发生变化的时候就不能很好的完成任务，因为要遍历所有的解空间开销太大了。

2. Content Based Input Attention

定义encoder和decoder的hidden state分别为$(e_1, ..., e_n)$、$(d_1, ..., d_{m(P)})$

$$u_j^i = v^T tanh(W_1 e_j + W_2 d_i)$$
$$a_j^i = softmax(u_j^i)$$
$$d_i' = \sum_{j=1}^{n} a_j^i e_j$$

然后把d_i,d_i'连接作为下一个step的隐藏层输入。

3. Ptr-Net

$$u_j^i = v^T tanh(W_1 e_j + W_2 d_i)$$
$$p(C_i|C_1, ..., C_{i-1}, P = softmax(u_i))$$

2 Iterative Alternating Neural Attention for Machine Reading

和以往的模型不同，不是将query压缩成一个向量，而是部署了部署一种迭代交替的注意机制，允许对查询和文档进行细粒度的探索。

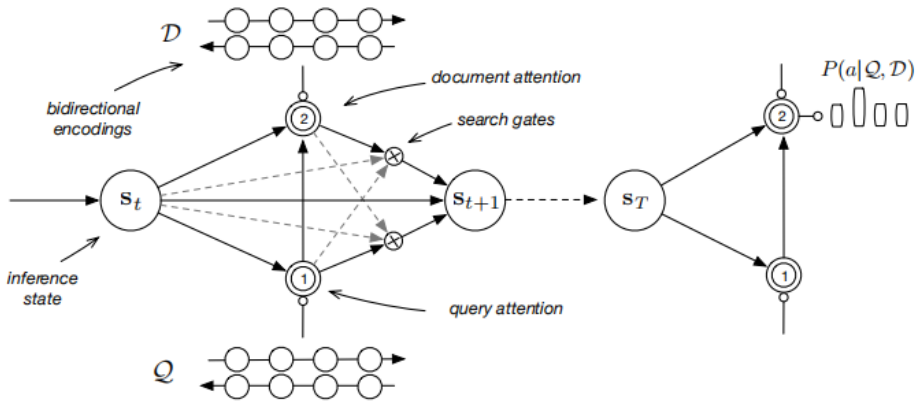


Figure 1: Our model first encodes the query and the document by means of bidirectional GRU networks. Then, it deploys an iterative inference mechanism that alternates between attending query encodings (1) and document encodings (2) given the query attended state. The results of the alternating attention is gated and fed back into the inference GRU. Even if the encodings are computed only once, the query representation is dynamic and changes throughout the inference process. After a fixed number of steps T, the weights of the document attention are used to estimate the probability of the answer $P(a|\mathcal{Q},\mathcal{D})$.

图 2: 网络结构图

整个推理过程由GRU实现，由以下三部分组成。

1. Bidirectional Encoding
   将document和query的word embedding送入Bi-RGU，$\widetilde{x}_i = [\overrightarrow{h}_i, \overleftarrow{h}_i]$ 分别表示为$\widetilde{q}_i$,$\widetilde{d}_i$

2. Iterative Alternating Attention
   递归网络迭代地执行交替搜索步骤，以收集可能对预测答案有用的信息。在每一个time step：

   [1] 对query encoding执行attentive read，得到一个q_t

   [2] 在给定q_t的情况下，抽取出d_t，代表document中与query相关的部分。

   两种抽取都是依据inference GRU S_{t-1}之前的隐藏状态：

$$q_{i,t} = \underset{t=1,\dots,|Q|}{softmax} \tilde{q_i}^T (A_q s_{t-1} + a_q)$$

$$q_t = \sum_i q_{i,t} \tilde{q_i}$$

$$d_{i,t} = \underset{t=1,\dots,|D|}{softmax} \tilde{d_i}^T (A_d [s_{t-1}, q_t] + a_d)$$

$$d_t = \sum_i d_{i,t} \tilde{d_i}$$

为了更新当前step的hidden state，inference GRU会根据以上两步收集到的信息来进行更新。

$$s_t = f([q_t, d_t], s_{t-1})$$

这里的f是GRU的更新函数。

3. Answer Prediction
   经过固定的时间步之后，根据最后一个step的$d_{i,T}$来预测答案的概率。

$$P(a|Q, D) = \sum_{i \in I(a,D)} d_{i,T}$$

here I(a,D) 是a出现在Document中的位置集合。训练的目标函数就是最大化P。

3 MACHINE COMPREHENSION USING MATCH-LSTM AND ANSWER POINTER

提出了一种end-to-end的神经网络结构，基于match-LSTM和Pointer Net。又根据Pointer的工作方式分为连续型和boundary型。

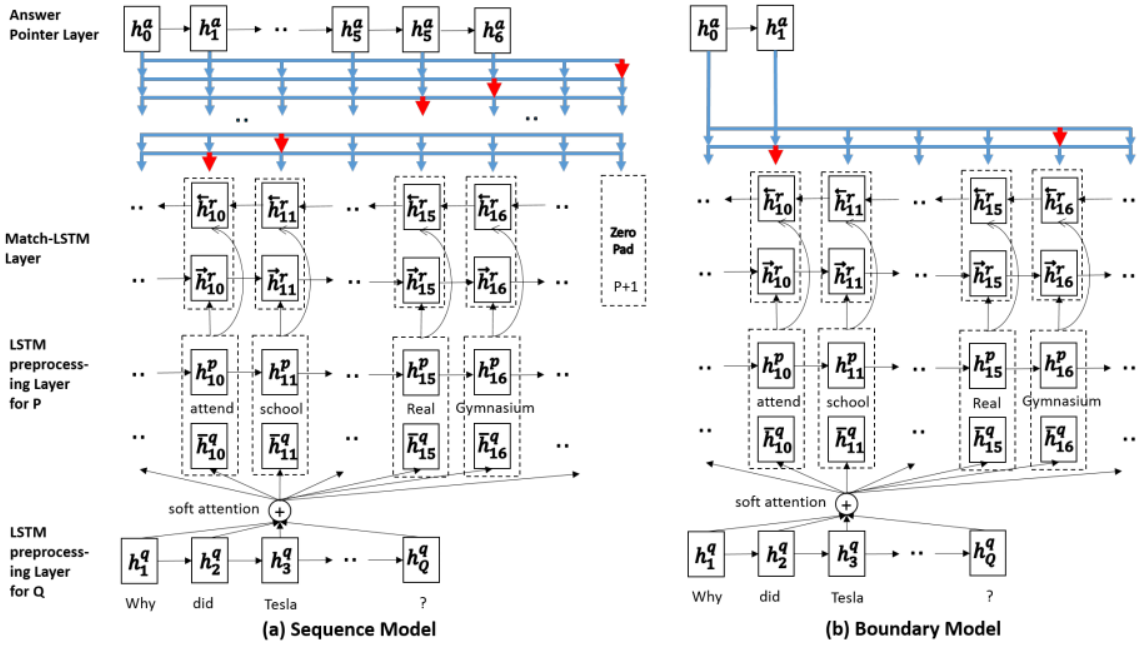


图 3: 两种结构对比

1. Match LSTM
   最初是解决文本蕴含问题提出的，在hypothesis的每一个位置，会通过attention机制来得到一个加权的premise向量。这个向量会被加到这个位置的hypothesis表示中，然后送入LSTM。这个过程叫做match-LSTM。

2. LSTM Preprocessing Layer

$$H^p = \overrightarrow{LSTM}(P)$$
$$H^q = \overrightarrow{LSTM}(Q)$$

3. Match-LSTM Layer
   将query看作premise，passage看作hypothesis。

$$\begin{aligned}\overrightarrow{\mathbf{G}}_i &= \tanh(\mathbf{W}^q\mathbf{H}^q + (\mathbf{W}^p\mathbf{h}_i^p + \mathbf{W}^r\overrightarrow{\mathbf{h}}_{i-1}^r + \mathbf{b}^p) \otimes \mathbf{e}_Q),\\ \overrightarrow{\alpha}_i &= \mathrm{softmax}(\mathbf{w}^r\overrightarrow{\mathbf{G}}_i + b \otimes \mathbf{e}_Q),\end{aligned}$$

$$\overrightarrow{z}_i = \begin{bmatrix} h_i^p \\ H^q\overrightarrow{\alpha^T}_i \end{bmatrix}$$

$$\overrightarrow{h^r}_i = \overrightarrow{LSTM}(\overrightarrow{z}_i, \overrightarrow{h^r}_{i-1})$$

同时还有另一个反向的LSTM类似以上结构，然后把两个LSTM的输出连接起来。

$$H^r = \begin{bmatrix} \overrightarrow{H^r} \\ \overleftarrow{H^r} \end{bmatrix}$$

4. Answer Pointer Layer

$$\begin{aligned} \mathbf{F}_k &= \tanh(\mathbf{V}\tilde{\mathbf{H}}^{\mathrm{r}} + (\mathbf{W}^{\mathrm{a}}\mathbf{h}^{\mathrm{a}}_{k-1} + \mathbf{b}^a) \otimes \mathbf{e}_{(P+1)}), \\ \beta_k &= \mathrm{softmax}(\mathbf{v}^\intercal \mathbf{F}_k + c \otimes \mathbf{e}_{(P+1)}), \end{aligned}$$

$$\mathbf{h}^{\mathrm{a}}_k = \overrightarrow{LSTM}(\tilde{\mathbf{H}}^{\mathrm{r}}\beta_k^\intercal, \mathbf{h}^{\mathrm{a}}_{k-1}).$$

$$p(\mathbf{a}|\mathbf{H}^{\mathrm{r}}) = \prod_k p(a_k|a_1, a_2, \ldots, a_{k-1}, \mathbf{H}^{\mathrm{r}}),$$

$$p(a_k = j|a_1, a_2, \ldots, a_{k-1}, \mathbf{H}^{\mathrm{r}}) = \beta_{k,j}.$$

$$-\sum_{n=1}^{N} \log p(\mathbf{a}_n|\mathbf{P}_n, \mathbf{Q}_n).$$

图 4: The Sequence Model

$$p(\mathbf{a}|\mathbf{H}^{\mathrm{r}}) = p(a_{\mathrm{s}}|\mathbf{H}^{\mathrm{r}})p(a_{\mathrm{e}}|a_{\mathrm{s}}, \mathbf{H}^{\mathrm{r}}).$$

图 5: The Boundary Model

4 Gated Self-Matching Networks for Reading Comprehension and Question Answering

首先通过gated attention-based RNN来得到question-aware的passage表示。然后提出了使用self-matching来获得更加细粒度的表示。

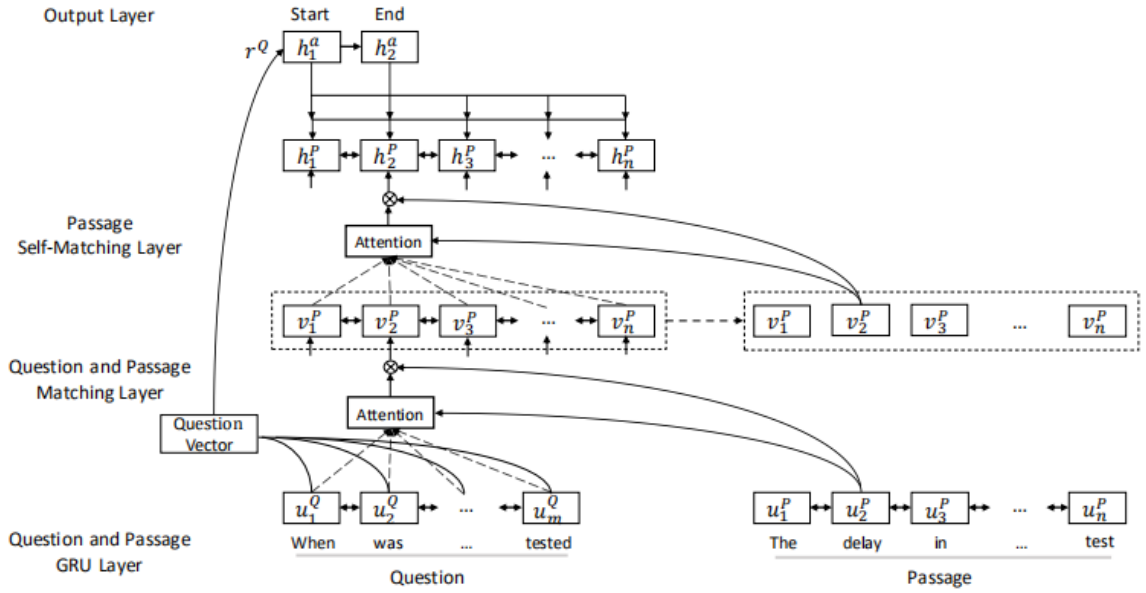


图 6: 结构图

1. Question and Passage Encoder
   使用了word-level和character-lever的embedding，然后送入BiRNN

$$u_t^Q = BiRNN_Q(u_{t-1}^Q, [e_t^Q, c_t^Q])$$
$$u_t^P = BiRNN_P(u_{t-1}^P, [e_t^P, c_t^P])$$

2. Gated Attention-based Recurrent Networks
   sentence-pair representation :

$$v_t^P = RNN(v_{t-1}^P, c_t)$$

   here $c_t = att(c^Q, [u_t^P, v_{t-1}^P])$

$$s_j^t = v^T tanh(W_u^Q u_j^Q + W_u^P u_t^P + W_v^P v_{t-1}^P)$$
$$a_i^t = \frac{s_i^t}{\sum_{j=1}^m s_j^t}$$
$$c_t = \sum_{i=1}^m a_i^t u_i^Q$$

   match-LSTM:

$$v_t^P = RNN(v_{t-1}^P, [u_t^P, c_t])$$

add another gate:

$$g_t = sigmoid(W_g[u_t^P, c_t])$$
$$[u_t^P, c_t]^* = g_t \odot [u_t^P, c_t]$$

然后就用这个代替原本的$[u_t^P, c_t]$,这就是gated attention-based Recurrent Network。这个gate有效的建模了只有passage中的部分与问题相关的这个现象。

3. Self-Matching Attention

$$h_t^P = RNN(h_{t-1}^P, [v_t^P, c_t])$$

here $c_t = att(v^P, v_t^P)$

$$s_j^t = v^T tanh(W_v^P v_j^P + W_v^{\tilde{P}} v_t^P)$$
$$a_i^t = \frac{s_i^t}{\sum_{j=1}^n s_j^t}$$
$$c_t = \sum_{i=1}^m a_i^t v_i^P$$

这里也使用了同上的additional gate。

4. Output Layer

$$s_j^t = \mathrm{v}^{\mathrm{T}} \tanh(W_h^P h_j^P + W_{\mathrm{h}}^a h_{t-1}^a)$$
$$a_i^t = \exp(s_i^t) / \Sigma_{j=1}^n \exp(s_j^t)$$
$$p^t = \arg\max(a_1^t, \ldots, a_n^t)$$

$$c_t = \Sigma_{i=1}^n a_i^t h_i^P$$
$$h_t^a = \mathrm{RNN}(h_{t-1}^a, c_t)$$

$$s_j = \mathrm{v}^{\mathrm{T}} \tanh(W_u^Q u_j^Q + W_{\mathrm{v}}^Q V_r^Q)$$
$$a_i = \exp(s_i) / \Sigma_{j=1}^m \exp(s_j)$$
$$r^Q = \Sigma_{i=1}^m a_i u_i^Q$$

5 Attention-over-Attention Neural Networks for Reading Comprehension

提出了一种简单但是有效的模型，在document-level Attention上又放置了一层Attention。另外一个贡献是提出了N-best-reranking策略。

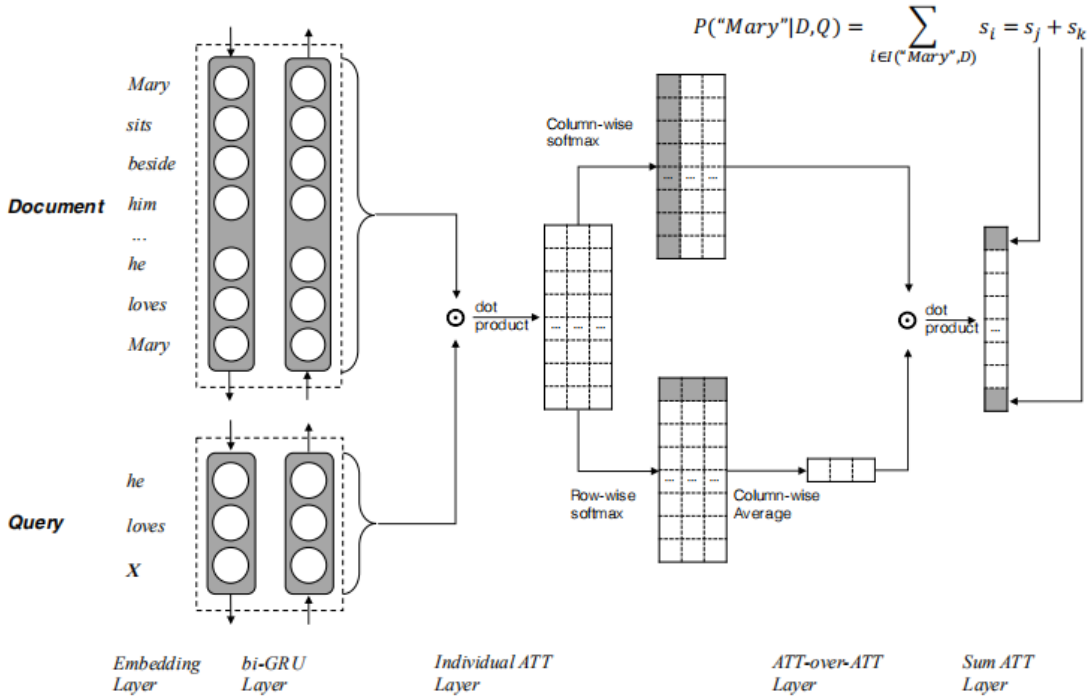


图 7: 网络结构图

1. Attention-over-Attention

$$M(i,j) = h_{doc}(i)^T h_{query}(j)$$

然后分别对行和列进行softmax运算。

$$\alpha(t) = softmax(M(1,t), ..., M(|D|,t))$$

$$\alpha = [\alpha(1), ..., \alpha(|Q|)]$$

$$\beta(t) = softmax(M(t,1), ..., M(t,|Q|))$$

$$\beta = \frac{1}{n}\sum_{t=1} |D|\beta(t)$$

$$s = \alpha^T \beta$$

$$P(w|D,Q) = \sum_{i \in I(w,D)} s_i$$

训练目标：最大化：$\mathcal{L} = \sum_i log(p(x)), x \in \mathcal{A}$

2. N-best-Reranking
   不是选出分数最高的作为答案，而是在解码的过程中生成最好的N个候选答案列表。将N个候选答案填入句子，用语言模型来对句子进行打分，重新选出得分最高的答案。

6 BI-DIRECTIONAL ATTENTION FLOW FOR MACHINE COMPREHENSION

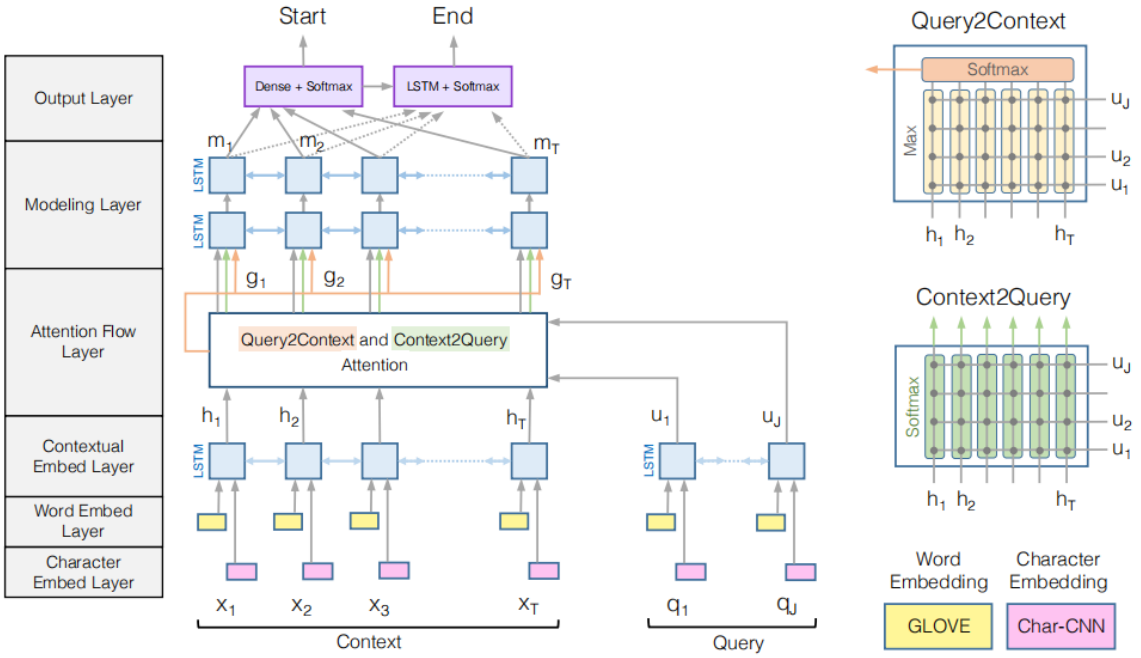


图 8: BiDirectional Attention Flow Model

1. Attention Flow Layer

H和U分别为经过BiLSTM的context和query。$S \in \mathbb{R}^{T \times J}$计算H和U的相似度。

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

这里选择$\alpha(h, u) = w_{(S)}^T[h; u; h \circ u]$

[1] Context-to-query Attention

$$a_t = softmax(S_{t:})$$

$$\tilde{U}_{:t} = \sum_j a_{tj} U_{:j}$$

[2] Query-to-context Attention

$$b = softmax(max_{col}(S))$$

$$\tilde{h} = \sum_t b_t H_{:t}$$

将其在列方向重复T遍组成$\tilde{H}$。

$$G_{:t} = \beta(H_{:t}, \tilde{U}_{:t}, \tilde{H}_{:t})$$

2. Modeling Layer
   将G送入LSTM，得到矩阵$M \in \mathbb{R}^{2d \times T}$

3. Output Layer

$$p^1 = softmax(w_{p^1}^T [G, M])$$

   将M送入另一个Bi-LSTM，得到M^2

$$p^2 = softmax(w_{p^2}^T [G, M^2])$$

4. Training

$$L(\theta) = -\frac{1}{N} \sum_i^N log(p_{y_i^1}^1) + log(p_{y_i^2}^2)$$

7 Gated-Attention Readers for Text Comprehension

提出了一种multi-hop结构，结合一种新的注意力机制。在GA Reader的迭代中，逐渐细化embedding的上下文表示。

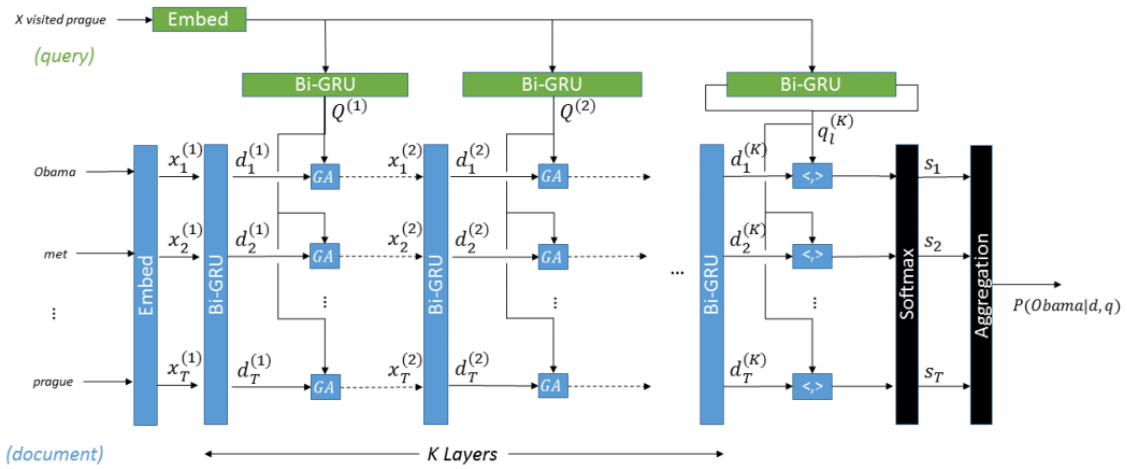


图 9: 网络结构

1. Multi-Hop Architecture

$$D^{(k)} = \overleftrightarrow{GRU}_D^{(k)}(X^{(k-1)})$$

$$Q^{(k)} = \overleftrightarrow{GRU}_Q^{(k)}(Y)$$

$$X^{(k)} = GA(D^{(k)}, Q^{(k)})$$

2. Gated-Attention Module

$$\alpha_i = softmax(Q^T d_i)$$

$$\tilde{q}_i = Q\alpha_i$$

$$x_i = d \odot \tilde{q}_i$$

3. Answer Prediction

$$s = softmax((q_l^{(K)})^T D^{(K)})$$

$$Pr(c|d,q) \propto \sum s_i$$

8 A Constituent-Centric Neural Architecture for Reading Comprehension

提出了一种Tree LSTM以及Chain-of-trees LSTM。将解析树与LSTM相结合。

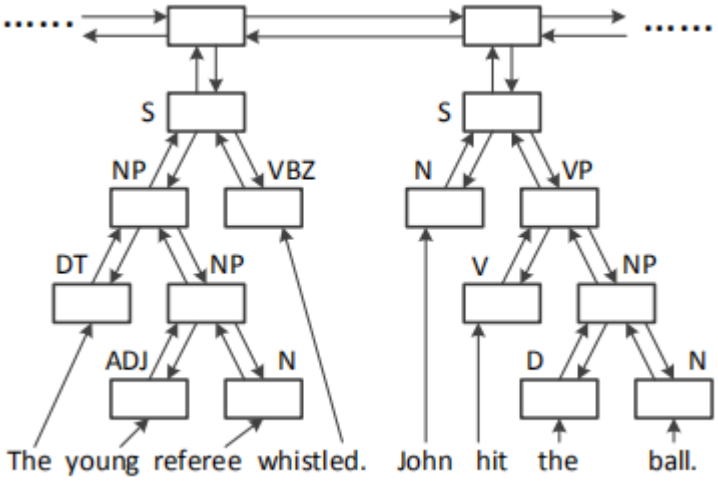


图 10: Chain-of-trees LSTM

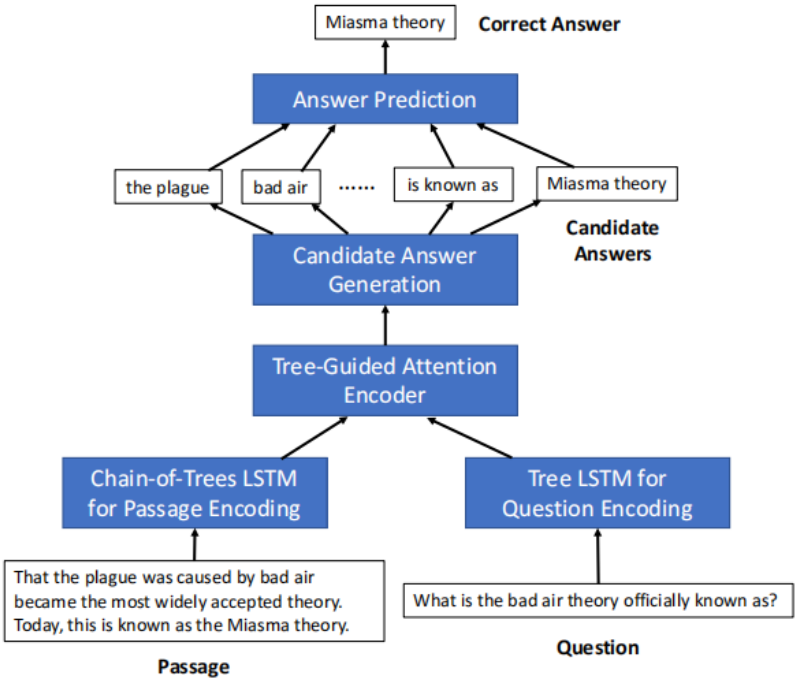
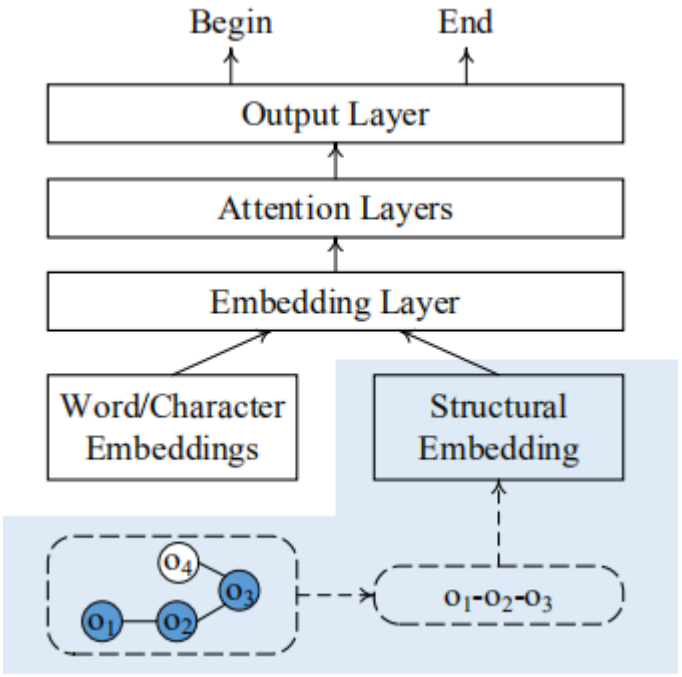
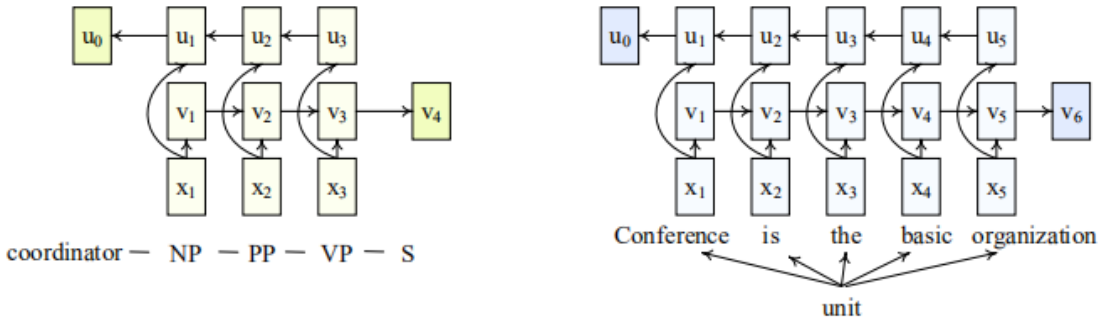


图 11: Constituent-centric neural network

9 Structural Embedding of Syntactic Trees for Machine Comprehension

提出了SEST以及SEDT，利用句子的结构信息并将其embedding。





(a) A SECT example (b) A SEDT example

10 Accurate Supervised and Semi-Supervised Machine Reading for Long

将文档切分为一个个小的，重叠的window，然后通过RNN来并行的编码。提出了Sliding-Window Encoder Attentive Reader（SWEAR）模型。另外还提出了一个半监督学习的版本。