Models

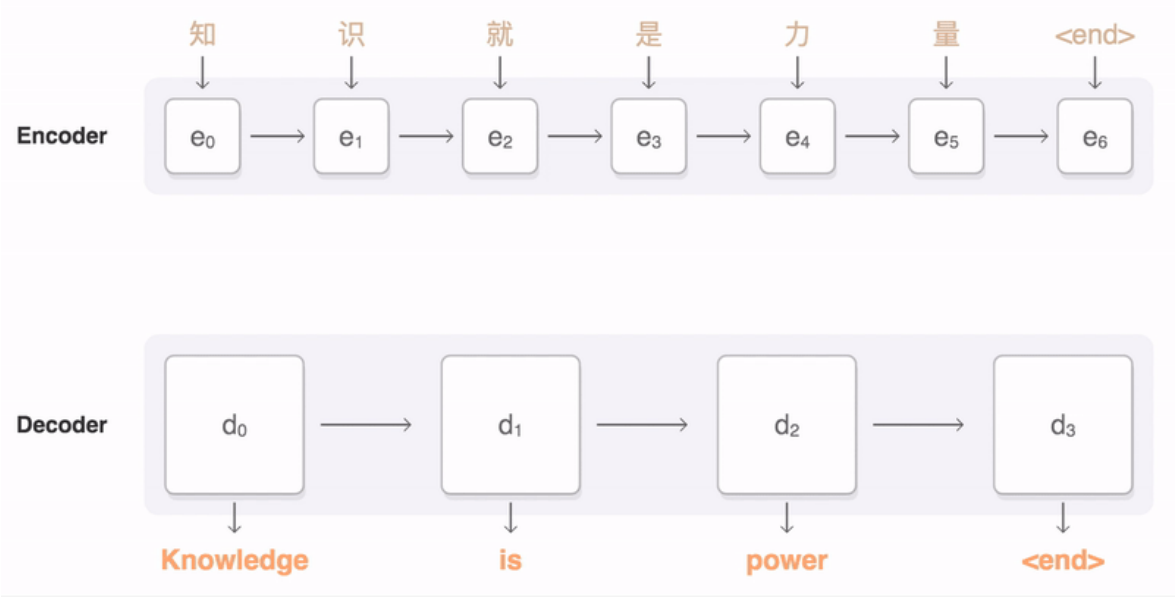
主要介绍机器阅读理解的深度神经网络模型。

Some Keywords Need to Know!!!

阅读理解模型常用知识储备。

Attention

- Attention过程



- 计算注意力主要分三个步骤：
    1. 计算query和每个key(Encoder中的项)之间的相似性$f_c(q,k_i)$以获得注意力分配权重。其中相似函数有点积、拼接、检测器等。
    2. 然后使用softmax函数来正则化这些权重。
    3. 最后将这些权重与相应的value(NLP任务key=value)一起加权并获得最终的值。

$$A(q,(k,v)) \rightarrow_{\substack{maps\ as \\ output}} \rightarrow \Sigma_{\substack{k \\ i=1}} f_c(q,k_i)v_i, q \in Q, k \in K, v \in V$$

- self Attention
- 三类模型
    - Attention Reader
        - 通过动态attention机制从文本中提取相关信息，再依据该信

息给出预测结果
    - Attention-Sum Reader
        - 只计算一次attention weights，然后直接喂给输出层做最后的预测，也就是利用attention机制直接获取文本中各位置作为答案的概率，和pointer network类似思想，效果依赖于对query的表示
    - Multi-hop Attention
        - 计算多次attention

Pointer Network

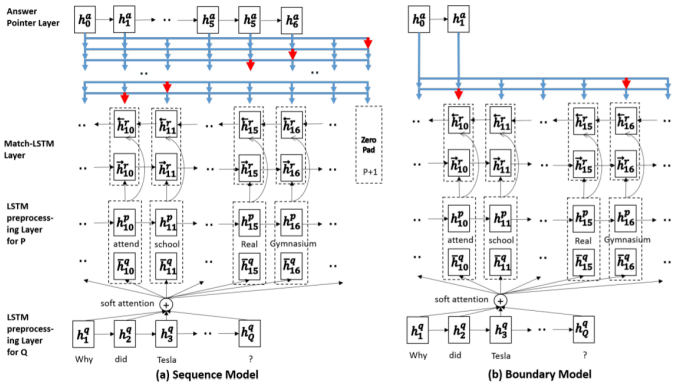
用来学习输出序列的条件概率，其中的元素是与输入序列中的位置对应的离散序列。采用注意力机制解决可变大小输出字典的问题，将注意力机制作为指针来选择输入序列的一成员来作为输出序列。

简言之就是用 attention 的机制，预测目标序列在原文中的位置，每一步解码的概率，就是 attention score 经过 softmax 之后的概率分布。

Main Models

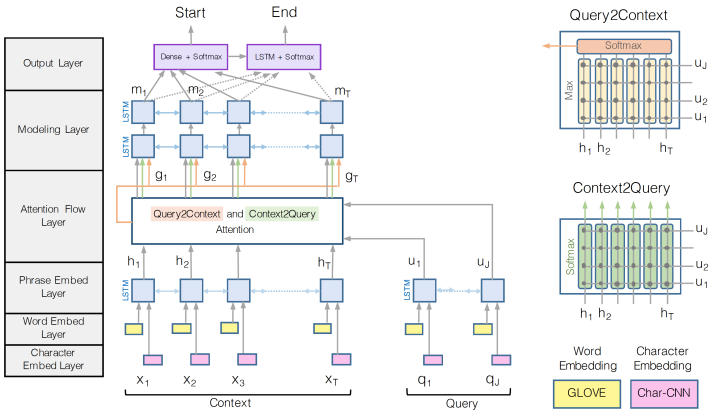
- 传统模型(参考论文的模型介绍、背景部分)
    - 基于手工设计的语法
    - 基于检测谓词参数三元组的信息提取方法
    - 作为关系数据库进行查询
    - (缺乏大规模训练数据集)
- 监督机器学习模型
    - 对问题、篇章分别进行词法、句法分析，针对分析结果进行特征提取：
    - 基于特征采用诸如 LR、CRF 等模型进行答案边界预测；
    - 采用梯度下降类算法在训练集上进行优化，拟合数据分布。
    - 基于特征的逻辑回归(SQuAD数据集baseline)
- 神经网络模型解决机器阅读理解的开端
    - Teaching Machines to Read and Comprehend
        - Attentive Reader
        - Impatient Reader
- 经典模式：End2End， Embed 层，Encode 层，Interaction 层和 Answer 层
    - Embed 层负责将原文和问题中的 tokens 映射为向量表示；

    - Encode 层主要使用 RNN 来对原文和问题进行编码，这样编码后每个 token 的向量表示就蕴含了上下文的语义信息；
    - Interaction 层是大多数研究工作聚焦的重点，该层主要负责捕捉问题和原文之间的交互关系，并输出编码了问题语义信息的原文表示，即 query-aware 的原文表示；
    - Answer 层则基于 query-aware 的原文表示来预测答案范围。
- 近年机器阅读理解发展
    - Attentive Reader
    - Match-LSTM



(1) encoder第一部分，Passage(P)和Question(Q)分别经过Bi-LSTM(LSTM preprocessing layer)得到HP和Q，通过soft attention实现Q和P得第一次融合。
(2) encoder第二部分，Match-LSTM Layer，通过attention机制，得到P中每个词i关于Q里每个词的权重a_i，然后将a_i作用到HQ上，得到新的向量。可以认为这个向量是P融合了Q之后的结果，再跟HQ拼接在一起，实现Q和P的第二次融合。
需要注意的是，这里的attention得到的是P里每个词对Q里每个词的两两的权重(而不是对Q整个序列的权重)
输入到下一层的包括：attention vector，上一层的hidden state，上一时刻的hidden state
(3) decoder，将答案预测看做一个序列问题，用pointer network去选择答案。
图中包含两种方式：其一是按顺序预测整个答案序列；其二是预测答案边界。后者效果更好，几乎成为了后续抽取式模型的标配(包括Bert fine tuning)。

    - BiDAF



主要是Context-to-query (C2Q) attention 和 Query-to-context (Q2C) attention。

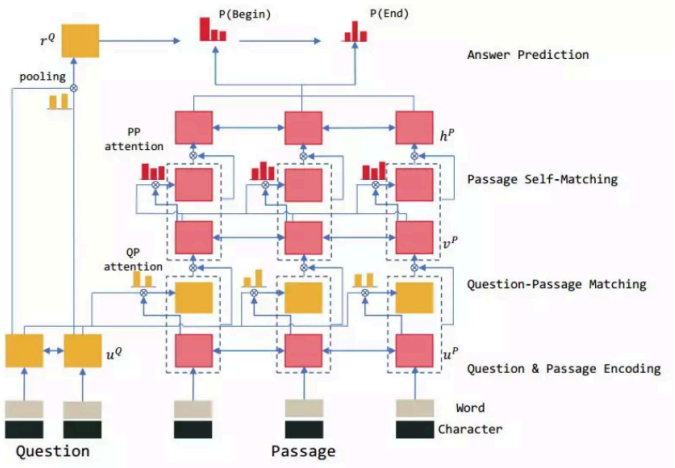
(1) Context2Query 总体上，与前面模型一样，考虑到用Q融合进P，去attend P里的每个词

(2) Query2Context 创新性的用P去attend Q，与Context2Query的唯一区别是对attention score取了max后再对h加权平均，得到单个向量。

(3) encoder输出层做了类似特征工程的trick

(4) decoder 用Q和P通过decoder每一步出来的M，去attend encoder的输出G，然后用softmax作用，去预测起始位置。只不过在预测位置终止时，M又额外经过了一层LSTM得到M2。损失函数仍然是cross entropy。

- R-Net

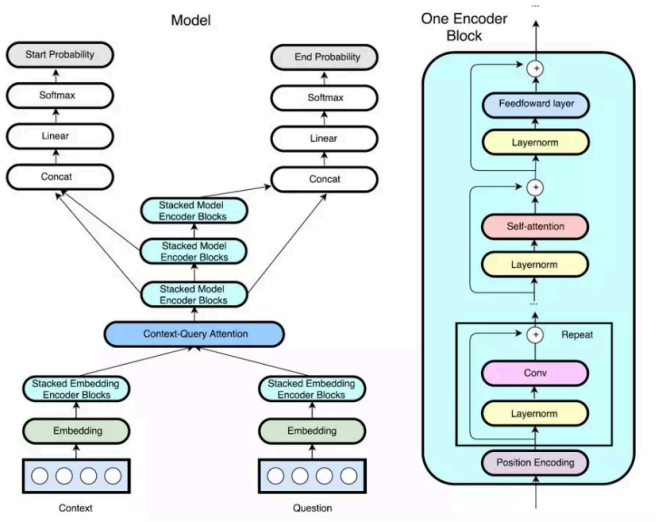


(1) encoder 中加入Gated Match-LSTM模块，可以理解为通过gate过滤掉输入中跟问题和答案不相关的部分。

(2) encoder 中还加入了self-attention机制，使得文章内部的词与词之间相互融合，通过自身上下文信息辅助筛选有价值的词。attention vector 是当前词和所在文章中所有词做attention得到的结果。

(3) decoder 仍然采用pointer network，但额外引入了question的信息，作为decoder隐藏层的初始输入。

- QANet



(1) encoder 采用CNN实现，由多个结构一样的encoder block实现。每个encoder block里又包含多个小单元：

(1a) Layer-norm 可以可以看做是batch-norm的变种，简单理解，batch-norm是对每一个batch里的x进行归一化，而layer-norm 是对每一层的x输出归一化。加快...

(1b) convolution 采用depthwise separable convolution，就是把一个 H x W x D 的卷积核，分解成 H x W x 1 和 1 x D 俩矩阵，从而降低矩阵的秩，减少参数量。 这是模型压缩很常见的一个做法。

(2) encoder 里有一层context to query的attention，与Bi-DAF类似但不一样，借鉴叫DCN的模型。

(3) decoder 沿用Bi-DAF。

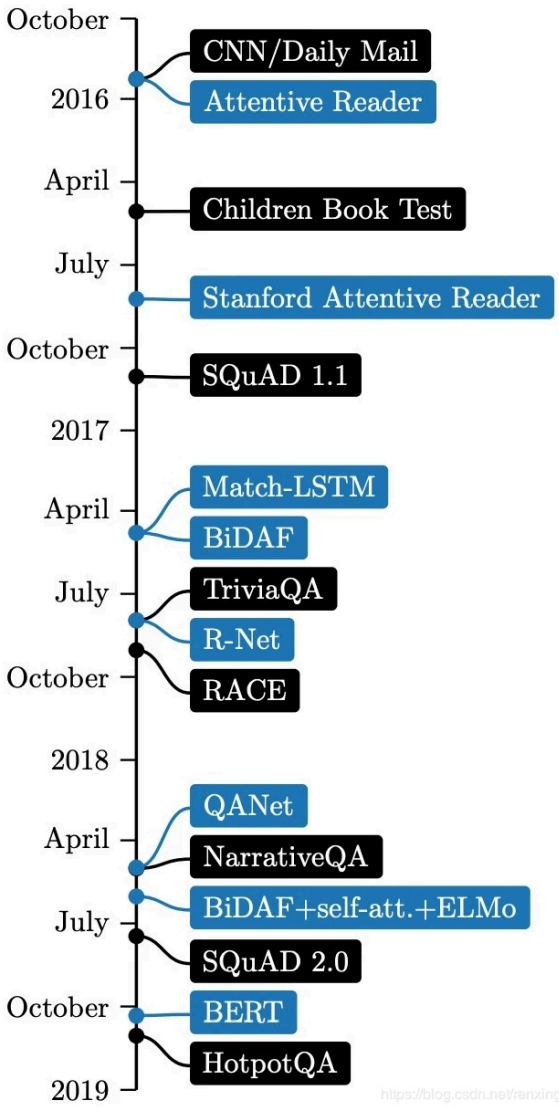
(4) 用English-French-English进行数据扩充(data augment)。

特点是训练快(Bi-DAF的5-8倍)，内存需求大。

- S-Net

  前面所有模型都是端到端的抽取式模型，而S-Net是分成了两部分的生成式模型。先从文章中抽取evidence snippets，再通过生成式的decoder生成答案。

- GPT & BERT
  - 预处理模型



state of the art

BERT+ AoA(Attention over Attention) + DAE(Data Augment Enhance, MT or Q-A)

Trick

- Teacher model => student model<Born-Again Neural Networks, Self-Competition>