QA-SYSTEM
问答系统

金卓林

哈尔滨工业大学, 150001
<kimnoic@gmail.com>

摘要

我们应用神经网络框架来解决非实质问题的回答任务。我们的方法不依赖于其他的语言工具，也可以应用在其他语言领域中。参考了 CNN 神经网络结构 [?] 中的第二种。通过测试 develop.txt 文件，实验结果表明 MRR 在 0.77, 并比较了不同训练测试集的次数，最终选择了 9000 次作为最终结果。

关键字— 答案选择，问题回答，卷积神经网络，tensorflow

1. 介绍

自然语言理解为基础的问答系统在过去几年的人工智能领域一直是一个热门话题，许多系统都包括了问答系统例如 Siri 和 Watson 等。我们的任务是给出训练集，其中包括问题和问题的答案候选池以及做出最匹配的正确选项，在此基础上给出其他问题和候选答案的相关度。为了找到最佳匹配对，我们需要一个度量来衡量每个 QA 队的匹配度，以便选择具有最高度量值的 QA 对。我们先做了一个实验，将问题和答案分词后，用 wiki_cn 训练出的词向量进行余弦计算，作为相关度结果。实验表明 MRR 在 0.42，效果不是很好。又阅读了一篇论文 [?]，找到了新的做法。

2. 模型描述

在这部分将介绍该框架的深度学习框架，以及这个方法的结构。学习给定问题的分布向量及其答案候选，然后是用相似性来度量匹配度。

2.1. 基准系统

基准系统是一个字模型，第一步是训练 word embedding，被认为是问题中的每个标记及其候选答案的加权和。这就产生了问题和每个候选答案的向量表示，最后一步是计算每个问题和候选对之间的余弦相似度，具有最高余弦相似度的作为答案返回。

2.2. 卷积神经网络框架

CNN 利用三个重要思想可以帮助改进机器学习系统：稀疏的交互，参数共享和等值表示。稀疏的交互与传统的神经网络相反，其中每个输入和输出交互。在 CNN 中过滤器尺寸通常比输入尺寸小，因此输出与输入的部分窗口交互。参数共享是指在卷积运算中再利用滤波参数，而传统神经网络矩阵中的参数只能使用一次来计算输出。以下是展示 CNN 实现的一个例子。

$$\begin{pmatrix} w_{11} & w_{21} & w_{31} & w_{41} \\ w_{12} & w_{22} & w_{32} & w_{42} \\ w_{13} & w_{23} & w_{33} & w_{43} \end{pmatrix} \bigodot \begin{pmatrix} f_{11} & f_{21} \\ f_{12} & f_{22} \\ f_{13} & f_{23} \end{pmatrix} \quad (1)$$

左面的矩阵是输入矩阵，每个单词都是由三维的自向量表示，输入长度为 4，右矩阵 F 表示滤波器。

$$\begin{aligned} o_1 &= w_{11}f_{11} + w_{12}f_{12} + w_{13}f_{13} + w_{21}f_{21} + w_{22}f_{22} + w_{23}f_{23} \\ o_2 &= w_{21}f_{11} + w_{22}f_{12} + w_{23}f_{13} + w_{31}f_{21} + w_{32}f_{22} + w_{33}f_{23} \\ o_3 &= w_{31}f_{11} + w_{32}f_{12} + w_{33}f_{13} + w_{41}f_{21} + w_{42}f_{22} + w_{43}f_{23} \end{aligned} \quad (2)$$

滤波器 F 将保留最大值，这表示 F 与 W 匹配的最好程度。

2.3. 训练和损失函数

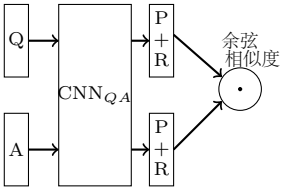
在训练过程中，每个问题都有一个最准确的答案 A+，通过将 A+ 和其他错误答案 A-配对进行训练。其中深度学习框架中输入为问题、正向答案、负向答案的向量。分别计算问题和两种答案的余弦相似度，并计较二者的差和边界的大小。当满足$cos(V_Q, V_{A+}) - cos(V_Q, V_{A-}) < m$ 时，表示正向答案排在负向答案后面。满足$cos(V_Q, V_{A+}) - cos(V_Q, V_{A-}) >= m$ 时，则不用更新参数。让正向答案和问题之间的向量 cosine 值要大于负向答案和问题的向量 cosine 值，大多少，就是 margin 这个参数来定义的。cos 值越大，两个向量越相近，所以通俗的说这个 Loss 就是要让正向的答案和问题愈来愈相似，让负向的答案和问题越来越不相似。损失函数为：

$$L = max\{0, m - cos(V_Q, V_{A+}) + cos(V_Q, V_{A-})\} \quad (3)$$

2.4. 结构

Q、A：句子中的每个字都需要转换成对应的字向量

CNN：列大小和字向量大小是一样的，对于每个句子而言，结果是一个列向量，取 max-pooling 就变成了一



个数字，得到了一个句子的语义表示向量。

R：是输出结果上加上 Relu 激活函数

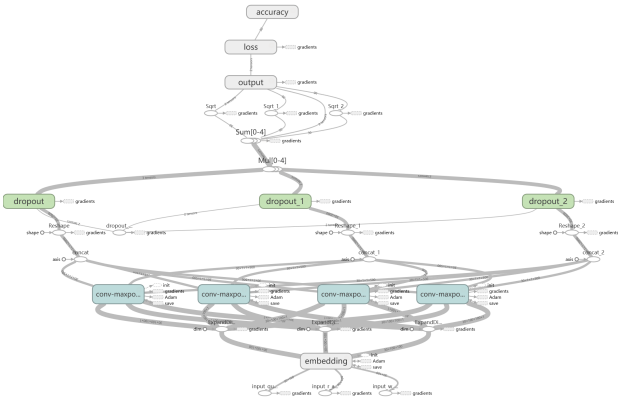
最后一层为计算向量的余弦夹角

原结构 [?] 中包含一层隐含层，因为测试效果不佳删去了。

3. 实验情况

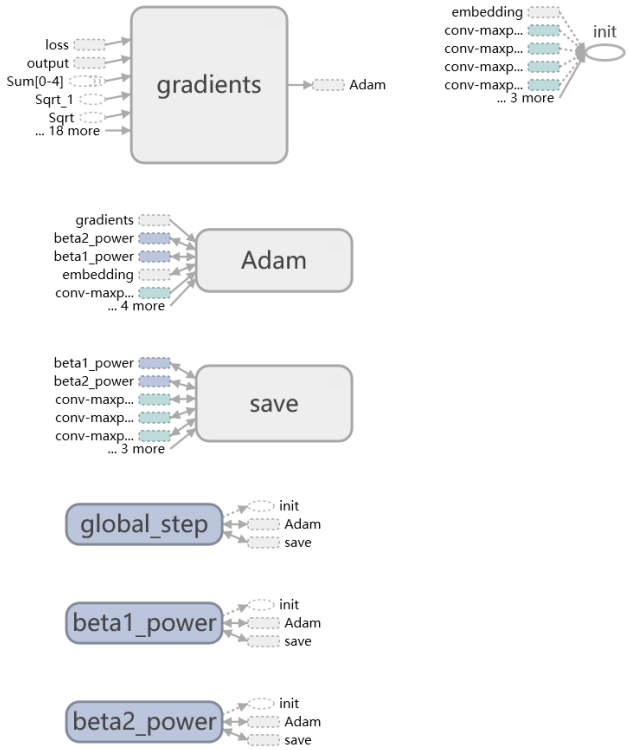
我们将上文提到的神经网络结构，使用 python 下的 tensorflow 库实现并用老师提供的数据集和开发集进行开发、用测试集进行测试。Tensorflow 工具集提供了深度学习所需要的大部分常用数据结构和函数的支持，此次未使用其他工具集。

我们部署的神经网络部分包括 word embedding 层，卷积-MaxPooling 层和输出层



上图为 tensorflow 下构建的神经网络的数据流图

其中 word embedding 时采用 100 维空间，并且每个字 word embedding 结果也是可优化参数。CNN 采用 4 个 filter，filter 大小分别为 1、3、4、5。MaxPooling 后输入项转换为 2000 维向量，输出层将三个输入项（问题，正向回答，负向回答）经过前两层得到的向量按照前述的方法计算余弦相似度，并取 m=0.05 按照前述公式计算损失函数 L。



在训练时，利用 tensorflow 的 feed 机制，从训练集中随机选择 30 个问题，然后每个问题随机选择一个正向回答和负向回答（如果没有负向回答则随机选择一个无关回答）作为一个批次 (batch)，参数的优化采用内建的 Adam 优化器 [?]，learning rate 为 0.1。在回答时，传入问题和待选答案，根据输出层中问题和答案的余弦相似度作为答案的权重。

4. 总结

最后时间比较仓促没有对迭代次数进行分析，可能会造成结果不同的影响。估计了一个大致收敛的位置，并对小规模数据测试，最终决定了迭代次数。

通过这次课程作业，还是学到了不少新的知识，包括对 QAsystem、CNN 的理解和 tensorflow 的简单使用。

5. 引用

[1] Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, Bowen Zhou, "APPLYING DEEP LEARNING TO ANSWER SELECTION: A STUDY AND AN OPEN TASK," ASRU,in the proceedings of ASRU 2015.

[2] 感谢 white127 , https://github.com/white127/insuranceQA-cnn-lstm 我们参考了他写的 cnn 网络, 并对一些参数做出了调整.

[3] Diederik P.Kingma , Jimmy Lei Ba, "ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION" ICLR,Published as a conference paper at ICLR 2015.