基于深度神经网络的检索相似问题模型

邢超（chao xing）
白子薇（ziwei bai）
2016/12/27

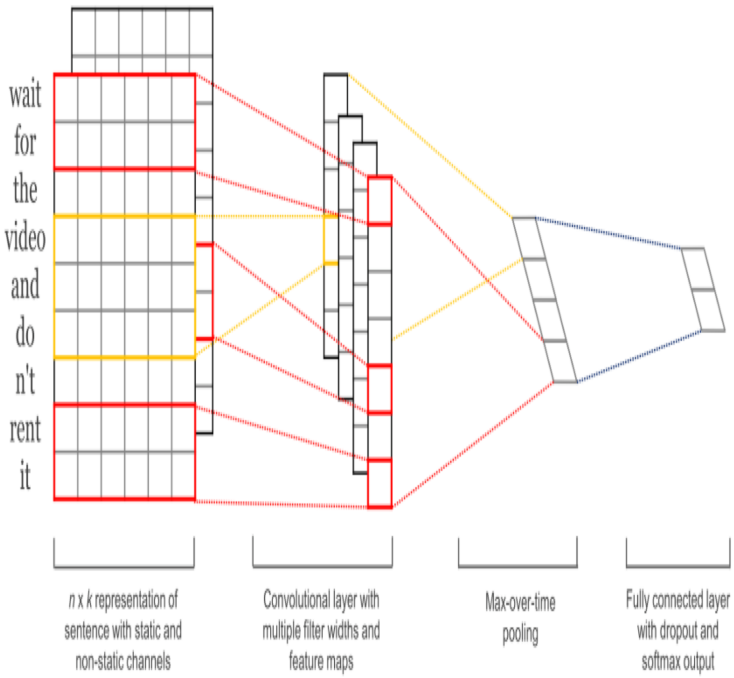
CSLT, RIIT, Tsinghua Univ.

1. 背景

随着网络大数据挖掘技术以及云计算技术逐渐成熟，深度神经网络在一些自然语言理解任务中展现出远超传统方法的效果以及计算速度。其中，由微软亚洲研究院提出的 DSSM（Deep Structured Semantic Model）模型在信息检索（IR, Information Retrieval）中被验证有着较高的性能。DSSM 模型的基本思路是将一个检错语句通过深度神经网络抽象成一个固定维度的向量表征，基于搜索与点击（Search-Click）关系构造分类模型，该模型试图将搜索语句与点击页面的标题在语义层面彼此靠近。

微软在 2012 年至 2014 年间，连续提出了多个 DSSM 模型，由不同的网络结构分为：DSSM（DNN based），CDSSM（CNN based），RDSSM（RNN based），LDSSM（LSTM based）等。其中，综合考虑计算时间以及在线策略，本文档主要介绍 CDSSM，旨在节约计算资源的前提下，保证最大性能。

CNN 应用于自然语言理解任务早在 2012 年就有研究人员提出，并且在文本分类、句子分类以及信息检索中得到了显著的成功。一个典型的 CNN 在语言理解任务中的应用 如图一所示。



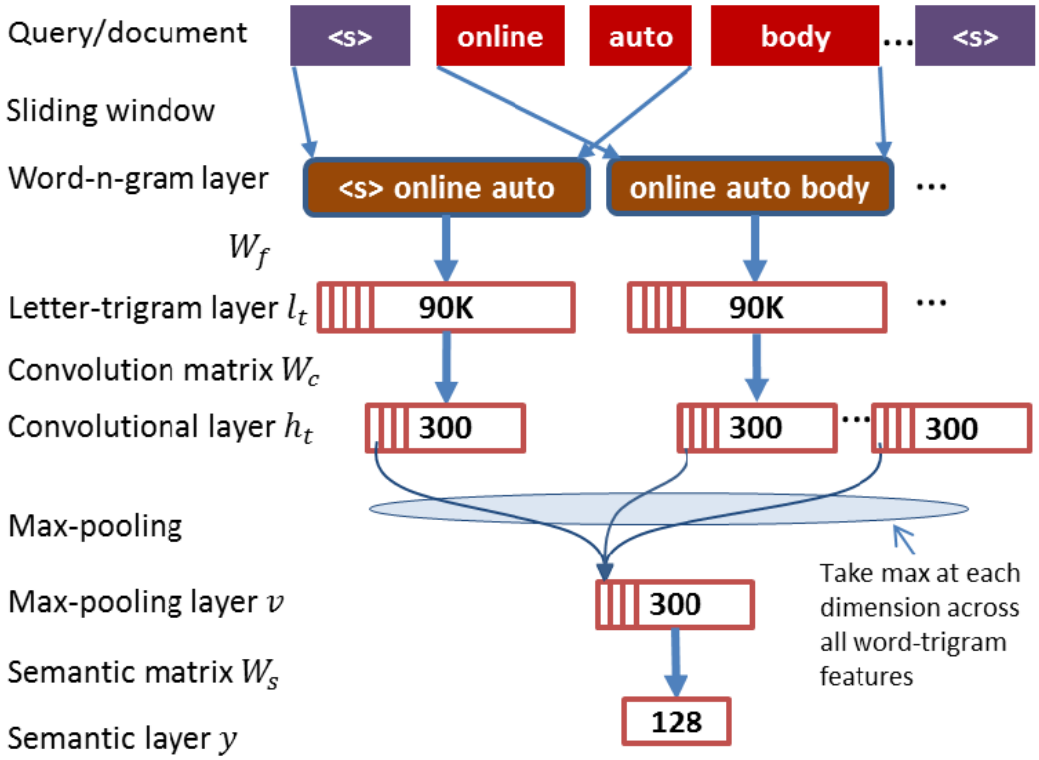
图一 CNN 提取文本信息图

CNN 能够捕获句子中不同位置的信息，并且这些信息抽象出来，用以获得较好的句子向量表征。微软提出的 CDSSM 模型，主要的贡献是：

1、提出了 Tri-Letter 的方法，试图解决 OOV 的问题

2、使用 CNN 抽象搜索语句表达，并且与另一个页面 CNN 抽象模型进行 Softmax 分类，优化不同的搜索 Query 对应点击页面的分类准确率。

CDSSM 模型图如图二所示：



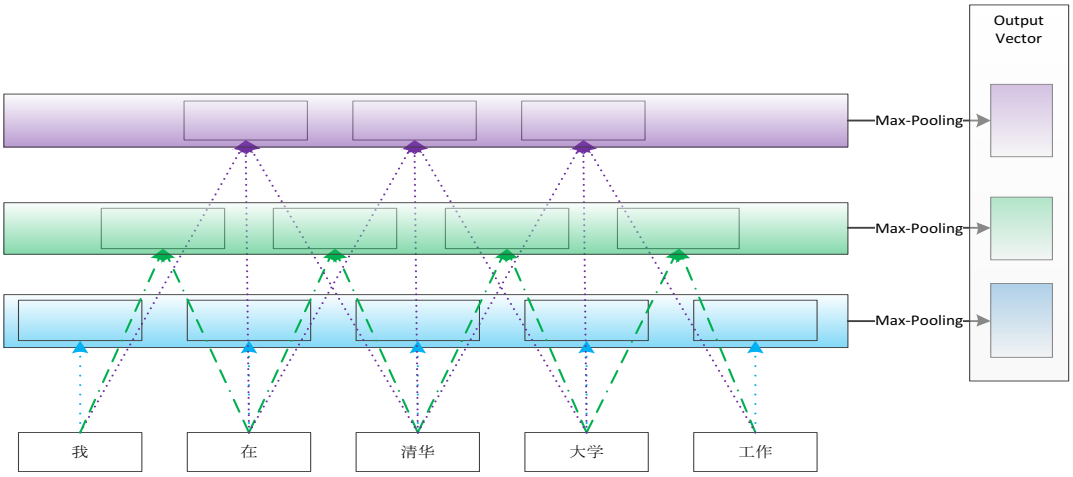
图二 CDSSM 模型图

2. 相似问题匹配模型

由于上述模型在微软 Bing 搜索引擎中的成功实践，我们提出了一种基于 CDSSM 模型的相似问题匹配模型，本模型针对智能问答系统中模板匹配的痛点，使用 CDSSM 模型的基本思想，将模板问题与标准问题进行分组，优化目标试图使得组内问题的相似度大，而不同组之间的问题相似度小。模型目标是在基于已有的标注模板进行训练，使得系统可以识别出大部分类似标注问题的问题从而减轻标注人员的工作量。本模型与微软提出的 CDSSM 模型基本一致，但汉语和英语具有差异性，汉语无法使用 letter tri-gram，因此，我们对模型进行了一些改进：逻辑输入由 letter tri-gram 改为 word，真实的输入是由 word2vec 训练得到的词向量，而非与 letter tri-gram 对应的 one-hot 向量。除此之外，其它两个不同点为：

1. 损失函数由 softmax 改为：$\text{Loss} = \max(0, cos(x, x^-) - cos(x, x^+) + margin)$
2. CNN 的 Region 数可配置，采用拼接的方式连接不同 region。

相似问题匹配模型如图三所示。



图三 相似度问题匹配模型图

3. 运行说明

3.1. 模型语料准备

模型的训练语料主要采用汇联标注好的模板，标注了：扩展问题以及标准问题。
通过将扩展问题中的相似词展开，删除语句中包含非中文的语句。得到多对相似问题。
对相似问题对进行初步处理以便我们后续过程的使用。

1、对数据进行聚类，每类分别保存在一个文件中
把所有直接相似的问题和间接相似的问题归并为一类，如果 A 与 B 相似，B 与 C 相似，可以认为 A 和 C 也是一对相似问题，即 ABC 彼此都相似，ABC 属于一类。

2、使用结巴分词对数据进行分词

通过以上方式得到的数据在之后的训练测试过程中不会再更改，并且之后训练、测试用到的数据都是由这些数据而来。我们提交的数据就是这些数据。

3.2 训练与测试脚本

3.2.1 采样脚本

初始语料的数据量太大，训练缓慢，对机器的内存和显存要求过高，因此我们需要对这些数据进行采样，以获得适量的数据进行模型的构建。为了更好的验证模型，我们采样了多组数据进行训练和测试。

使用 get_data.py 对初始数据进行采样,获取 N 组数据，每组数据有 1000 类，每类有 500 个数据。
使用示例：python get_data.py ../data/CDSSM_FILTER_WORD_v0/ test 4
（第一个参数是‘../data/CDSSM_FILTER_WORD_v0/’是初始数据存放目录，目录中每一个文件是一类数据，第二个参数是输出数据存储路径，第三个参数是采样的数量。示例中一共得到四个文件，每个文件代表一组采样数据，分别为 test_0,test_1,test_2,test_3，0、1、2、3 为每组数据的编号）

3.2.2 自动训练脚本

程序名称：

run_train.sh

功能：

1、 分割采样数据，90%作为训练数据，10%作为测试数据
2、 把训练数据整理成训练过程中所需要的矩阵
3、 训练模型并保存

使用示例：

sh run_train.sh 0 0
第一个参数‘0’是数据组的编号，用编号区分不同组数据及对应的模型，
第二个参数‘0’代表实验运行指定的 GPU

3.2.3 自动测试脚本

程序名称：

run_test.sh

功能：

1、 把测试数据整理成测试过程中所需要的矩阵
2、 根据训练得到的模型进行测试，分别测试相同数据在 top1、top5、top10 三种情况下的正确率,并记录 Bad case

使用示例：

sh run_test.sh 0 0

第一个参数‘0’是数据组的编号，根据编号可以定位到对应的测试数据与模型，第二个参数‘0’代表实验运行指定的 GPU

3.2.4 备注

Bad case 存在 result 文件夹中，文件名称与编号和 topk 有关。如 result_0_top1 即第 0 组数据在 top1 下的 Bad case。

因为机器不同，代码存储路径不同，在运行时，需要对 run_train.sh 和 run_test.sh 中 tool_path ， src_path 进行修改。

3.3 核心程序接口说明

本次提交 python 核心脚本分别为：simple_train_process.py，simple_test_process.py，toolbox。

其中：

simple_train_process.py 为训练脚本，参数为：

-source 接收一个训练样本矩阵，注：本脚本只接受一个每一个类中样本数为 450 的样本矩阵，并且样本矩阵已经转换为 word-index 矩阵

-dict 接收一个词向量文件，注：词向量文件中词向量需要进行 mean variance normalization ,词向量文件中的中文编码为 UTF-8

-linear CDSSM 的输出向量之后，按需要考虑是否配置线性层，使用方法为：线性层的数量 每层的节点数，如：3 1024 512 1024，按空格隔开。

-non CDSSM 的输出向量之后，按需要考虑是否配置非线性层，非线性层的激活函数为 tanh，使用方法为：非线性层的数量 每层的节点数，如：3 1024 512 1024，
按空格隔开。

-conv CDSSM 模型中的 Region 数量，以及不同 Region 中的 Step 数，按空格隔开，如：3 1 2 3。

-num-filter CDSSM 中 filter 的数量。

-batch 本文中描述的 CDSSM 模型，采用 mini-batch 方法更新参数。

-epoch 模型训练的轮数。

-output 输出模型存放的位置以及名称。

使用示例：python simple_train_process.py -source ../data/Data-1000-500.train.npy -dict ../data/word_vec_norm -linear 0 -non 0 -conv 3 1 2 3 -num-filter 200 -batch 500 -epoch 100 -output model/model

simple_test_process.py 为测试脚本，参数为：

-source 接收一个候选集矩阵。

-target　接收一个测试集矩阵。
-dict　接收一个词向量文件，注：词向量文件中词向量需要进行 mean variance normalization ，词向量文件中的中文编码为 UTF-8
-batch　由于内存、显存的限制，将不同词矩阵转化为 CDSSM 的输出矩阵时，需要按 batch 进行。
-match　每个测试集中的问题对应的候选集中的正确答案范围的 dict
-linear, -non, -conv, -num-filter, -output　与训练脚本中设置的参数一致。

使用示例：python simple_test_process.py -source ../data/Data-1000-500.train.npy -target ../data/Data-1000-500.test.npy -dict ../data/word_vec_norm -output model/model-90000 -match ../data/match_dict -linear 0 -non 0 -conv 3 1 2 3 -num-filter 200 -batch 500

4. 实验

4.1. 词向量准备

我们选择谷歌的开源工具 word2vec 对语料进行训练。

- 语料分布：汇联提供的 PlainText 数据 1.7G
- 分词工具：
  - 词向量：使用 jieba 分词工具，通过结巴分词中的默认词表对数据进行分词，允许使用 HMM 模型。
  - 字向量：将训练语料每个句子逐字隔开。
- Word2vec 训练参数设置如下表所示

类型	Window	Iteration	cbow	Size	sample	negative	hs	binary
词向量	8	20	0(skip-gram)	200	1e-4	5	0	0
字向量	8	18	0(skip-gram)	200	1e-4	5	0	0

4.2. 参数设置

- non:0
- linear:0
- conv: 3 1 2 3
- num_filter:100
- batch:500
- epoch:10

参数说明：不设置线性层和非线性层，卷积层有三层，每层的 Step 数分别为 1，2，3，filter 数量为 100。一个 batch 的大小为 500，一共训练 10 轮。

4.3. 实验结果

我们一共准备了四组训练语料与测试语料。记为 data_0; data_1;data_2; data_3.

四组数据 top1、top5、top10 的正确率如下

数据	data_0	data_1	data_2	data_3
Top1	0.9735	0.9697	0.97218	0.9449
Top5	0.9924	0.99092	0.99082	0.99238
Top10	0.9945	0.99388	0.99356	0.99516

从实验结果可以看到，预测过程中，考虑 top5 的问题比只考虑 top1 的问题正确率提升显著。但是考虑 top10 的问题与只考虑 top5 的问题相比，正确率提升不明显。

Bad case 主要集中在关键词区分的类上面，即关键词不一致，上下文基本一致，分到两个不同的类中。或关键词一致，上下文不一致，分到两个不同的类中。

示例：

1、关键词不一致，上下文基本一致：

眉粉怎么使用　　浴花怎么使

代替收付费有哪类交钱规则　　额外手续费存在哪类收款规则

派即刻清设有现货么　　预先整洁药剂有现货哇

支票业务的收缴规范是多少　　践约保单的收缴规范包含几种

温和型皮肤可以那个么　敏感型皮肤可以那个勒

腮红刷色怎么样　睫毛夹怎么用

2、关键词一致、上下文不一致

电话端领航能否准许设立　电话端领航怎么样设立呢

咋样查看订单哇　啥样取消订单呢

有关帮得佳家用用品的种类　　帮得佳家用用品有啥功能

红果清亮露具有副作用呢　红果清亮露拥有新的哇

俺要来双洁面膏　洁面膏清洁干净

注：因为部分数据分类有错误，导致在计算正确率时，一些实际分类正确的数据被判定为错误，导致正确率低于实际值。 我们随机采样了 200 个 Bad case，人工检查后发现有 118 个属于误判，占 59%。考虑误判的情况，不同数据集 top1 的实际正确率如下表所示：

数据	data_0	data_1	data_2	data_3
Top1	0.989135	0.987577	0.9885938	0.977409

5. 参考文献

【1】Huang P S, He X, Gao J, et al. Learning deep structured semantic models for web search using clickthrough data[C]//Proceedings of the 22nd ACM international conference on Conference on information & knowledge management. ACM, 2013: 2333-2338.

【2】Gao J, Deng L, Gamon M, et al. Modeling interestingness with deep neural networks: U.S. Patent Application 14/304,863[P]. 2014-6-13.

【3】Shen Y, He X, Gao J, et al. A latent semantic model with convolutional-pooling structure for information retrieval[C]//Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management. ACM, 2014: 101-110.