周报\_2018\_10\_07

1. LINE Large-scale Information Network Embedding （2015）

该文章提出了一种可以适用于大规模网络的嵌入模型，并且该模型可以应用于任意类型的网络：无向的、有向的或者是带权的网络。与DeepWalk相比而言，它形式化的提出了目标函数，提出了一阶近似以及二阶近似及其计算方法。对于一阶近似而言，它是考虑的网络的局部结构，例如如果是对于带权的网络而言，他们的权值即是他们的一阶近似的值，如果两个顶点之间没有边，那么他们的一阶近似的值就是0。对于二阶近似而言，它考虑的是网络的全局结构，即是对于邻居相似的节点的embedding向量也相似（类似与Skip-Gram算法）。DeepWalk算法仅仅考虑了二阶近似，最终的性能较LINE而言较差。该文章说明了一阶近似与二阶近似是互补的关系，可以将一阶近似与二阶近似生成的向量直接连接起来，会使得最终的效果有所提升。该文章还提出了edge sampling方法，解决了在带权网络中ASGD的梯度爆炸的问题。

1. Distributed Representations of Sentences and Documents （2014）

bag-of-word具有一系列的缺点：丢失了词序信息、忽略了词的语义以及稀疏高维等。该文提出了paragraph vector方法，来得到可变长度的输入序列的representations。相比之下，其他很多的方法仅仅适用于单个的句子的representations，例如RNN。Paragraph vector中有两个模型：A distributed memory model和Distributed bag of words，这两个模型都能得到paragraph vector，实验证明前者得到的vector比后者得到的更好；最终的paragraph vector是将两者拼接起来，实验证明拼接（concatenation）的效果要比sum的效果要好。前者与word2vec中的CBOW方法类似，仅仅是在输入层中添加了一个paragraph vector向量而已，直观上理解，这个paragraph vector就像是一个记忆单元，存储了当前上下文所丢失的信息。（是否可以使用LSTM来记录当前上下文所丢失的信息，然后进行embedding得到词向量，这样得到的词向量是否会优于word2vec所得到的词向量？）

1. Semantic proximity search on graphs with metagraph-based learning （2016）

该文章是第一个提出Semantic proximity search的：在异构图中，给一个查询节点以及一个期望的类，最终得到一个属于期望的类并且与查询节点最相似的节点。在异构图中，因为节点的类型不同，因此节点之间会存在不同的关系，即会产生不同的语义类。该文章使用metagraph来侧重于不同的语义类，例如：两个user，如果他们的school相同，major相同，那么他们就是classmate。类似的，可以定义不同的metagraph。该文章利用GRAMI算法从graph中生成metagraph。在得到metagraph之后，该文章从数据集中对metagraph（相当于模式）进行匹配，得到一系列的instances（这个操作即是对所定义的模式，在数据集中进行模式匹配）。再利用instances生成metagraph vectors。考虑这样的情况：例如所期望的类是classmate，那么如果两个user的address相同，surname相同，这并不能说明它们是classmate，即是这样的metagraph与所期望的类无关。于是可以增加参数w来衡量各个metagraph对于所期望的类的贡献程度，使用监督学习来训练w即可，从而得到最优的w\*。这样做可以减少不必要的计算（减少了不相关的metagraph的计算）。之后，在测试阶段，则是给出一个查询节点q，然后利用metagraph vectors以及w\*来估计proximity，最终根据proximity的大小对节点进行排序。

（该文章的metagraph有点类似于文章“metapath2vec\_ Scalable Representation Learning for Heterogeneous Networks”[1]的metapath，不同之处metagraph=n\*metapath，metapath是线性结构，而metagraph是非线性结构，对[1]而言，是采用的表示学习的方法，而对于本文章而言，只是采用的普通的监督学习的方法）。、

1. Convolutional Neural Nertworks for Sentence Classification （2014）

该文章通过实验证明了word2vec所生成的词向量是具有通用性的，能够跨数据集使用。即是在一个语料库中所生成的词向量，可以用在其他的场景中。该文章使用Mikolov已经训练好的词向量，与CNN结合，对数据集中的句子进行分类（例如：positive，negative等）。该文章的CNN只有一个卷积层，使用了非线性的卷积核，在maxpool层之后使用了dropout方法来避免过拟合。它提出了四个模型：CNN-rand：所有的向量是随机初始化的（并没有使用word2vec的向量）；CNN-static：使用word2vec中的向量，对其中没有的词，进行随机初始化，之后保持词向量不变；CNN-non-static：与CNN-static类似，只是要对词向量进行微调；CNN-multichannel：它的词向量的初始化与CNN-static类似，只是它们的CNN结构不同，CNN-multichannel的结构是它的输入层是有两个sentence向量，然后保持其中的一个不变，对于bp算法传回来的梯度，仅仅改变另一个sentence向量，从而实现对词向量进行微调的目的。

5、A Neural Probabilistic Language Model （2003）

该文章指出统计语言模型的目标是学习词序列的联合概率分布，但是由于维数灾难的存在，使得学习是困难的。对于统计语言模型而言，在此之前解决维数灾难的方法是n-gram模型，但是n-gram模型本身是有缺陷的：首先是它没有考虑到足够长的上下文（一般上下文的长度是1,2,3，继续增加会使得计算复杂度急剧增加，然而性能的提升很少），其次是没有考虑到词的相似度。本文提出的神经概率语言模型，因为概率函数是特征值的光滑函数，即是对特征的较小的改变只会使得概率函数发生小的改变，由于具有这样的优秀性质，使得了相似的词具有相似的词向量，其次因为参数的减少，使得其可以考虑更长的上下文（例如：5甚至更大）。这样神经概率语言模型就较好的解决了n-gram存在的问题。该文章还提出了一些加速模型训练的方法以及一些tricks：1、short list：在已知上下文的情况下，接下来的一个词其实并不是完全随机的，在整个单词表中只有一部分词可能会出现在其后，因此使用trigram模型计算出一个short list，然后最后的softmax可以基于short list，而不是整个单词表，从而减少了计算量；2、对大多数频繁输入的上下文，可以预先计算出它的输出；3、SGD；4、使用trigram与神经概率语言混合模型可以提高性能；5、使用随机初始化以及特征矩阵的SVD初始化；6、对于单词表中没有的词，使用short list中所有词的向量的加权平均进行估计；最终的实验证明了：1、更大的上下文可以获得更好的性能；2、性能高于n-gram模型；3、使用SVD初始化无效，可以直接使用随机初始化。

课外阅读：

1、A SOLUTION TO CHINA COMPETITIVE POKER USING （2019）

该文章实现了斗地主（CCP）的AI程序。整个模型框架被称作是DeepRocket，其中包含三个模块：the bid module，the policy network，the kicker network。其中第一个模块是用来评估在每个玩家拿到牌之后，谁最有可能是地主。第二个模块是是策略模块，用来确定玩家的下一步会有什么操作。第三个模块是用来确定“三带一”中，所带的“一”是什么牌（规定可以三带一对）。该程序所采用的是CNN，因为首先CNN在其他perfect information game中取得了比人类更好的性能（CCP是imperfect information game，不太清楚游戏的类型是怎么区分的），其次是因为CCP具有半平移不变性（它具有半平移不变性与使用CNN有什么关系？）。