|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 请按照“注意事项”正确填写本表各栏 | | | | | 此框内容由国家知识产权局填写 | |
| ⑦  发明名称 | 一种基于正则变分嵌入式的软件需求聚类方法 | | | | ① | |
| 申请号　　　（发明） | |
| ②分案 | |
| 提交日 | |
| ⑧  发  明  人 | 康雁、崔国荣、李媛、 | | | | ③申请日 | |
| ④费减审批 | |
| ⑤向外申请审批 | |
| ⑨第一发明人国籍 **中国** 居民身份证件号码142622199512071019 | | | | | ⑥挂号号码 | |
| ⑩  申  请  人 | | 名称 云南大学 | | | 电话 | |
| 社会统一信用代码 125300004312032649 | | | | 电子邮箱 |
| 国籍或注册国家（地区） | | 经常居所地或营业所所在地 | | |
| 邮政编码 650091 | 详细地址 云南省昆明市五华区翠湖北路2号 | | | |

本发明公开了一种基于正则变分嵌入式的软件需求聚类方法，正则变分嵌入式聚类方法解决了两端式聚类不能反向传播优化聚类中心和样本分布的问题，通过学习软件需求数据的内部隐藏分布，保留数据局部结构，以重参数的技巧模拟原始数据，反映数据本质特征，提高了特征质量和聚类准确率，达到了最优效果。在文本表示上，使用BERT句向量模型将原始文本映射到向量空间，由于原始文本未经噪声处理，所以在模型输入端融合Dropout正则化随机抑制部分神经元的工作，降低噪声干扰，防止模型过拟合，增强模型鲁棒性，经过Dropout正则化的向量输入到变分嵌入式聚类方法中，通过编码器使嵌入空间可以学习原始数据分布，重参数技巧确保隐藏层能较好抽象输入数据特点，然后嵌入空间向量一方面进行解码进行特征学习，另一方面使用K-means聚类方法进行样本划分，通过反向传播优化损失函数，待达到迭代停止条件输出聚类结果。



1. 一种基于正则变分嵌入式的软件需求聚类方法，其特征在于，包括：

S1、数据收集模块，本发明的数据集在网上很稀少，所以利用Scrapy爬虫技术获取Softpedia网站下的软件需求数据；

S2、数据预处理模块，对获取的软件需求数据进行文本预处理，然后句向量模型将软件需求文本映射到向量空间；

S3、模型训练模块，对于处理好的句向量输入到正则变分嵌入式聚类模型中，通过编码将高维稀疏特征压缩成低维密集向量，以概率形式生成于原始数据类似的样本，用来反映数据本质特征，然后新生成的向量同时进行特征学习和聚类划分；

S4、聚类输出模块，待模型达到终止条件，输出最终的聚类结果以及多种评价指标。

1. 根据权利要求1所述的基于正则变分嵌入式的软件需求聚类方法，其特征在于，步骤S1所述的数据收集模块具体为：利用Scrapy技术获取Softpedia网站下下Windows平台的11类数据集，每类数据以csv的格式单独存储，同时对每类数据进行标注。
2. 根据权利要求1所述的基于正则变分嵌入式的软件需求聚类方法，其特征在于，步骤S2所述的数据预处理模块具体为：

S21、文本预处理，由于软件需求数据是直接在网站上获取的，无法避免html标签等内容，首先使用正则表达式去除英文文本中的html标签，然后更改文本中的缩略词，对乱码单词进行拼写校正，最终进行词干提取和词形还原，将所有类的数据放在一个csv文件中。

S22、文本向量化，以文本中每个句子为单位，使用句向量模型将其映射到向量空间，句向量代表整个句子的文本信息。

1. 根据权利要求1所述的基于正则变分嵌入式的软件需求聚类方法，其特征在于，步骤S3所述的模型训练模块具体为：

S31、Dropout正则化，在输入端融合Dropout正则化，按照一定概率将神经网络的部分神经元停止工作，防止网络过拟合。

S32、编码，对高维稀疏化向量进行特征压缩，使密集向量包含更多的语义信息，通过重参数重新成生成与原始样本类似的文本，编码器将正则化向量转换为隐藏空间的两个参数，记作和，是一个服从正态分布的张量，隐藏空间使用重参数技巧：

S33、解码，我们对嵌入空间向量进行解码，解码的过程与编码的过程相反，将嵌入空间向量还原成原始向量。

S34、聚类划分，嵌入空间向量输入到K-means聚类方法中，K-means按照近邻原则划分样本到最近的类中，即满足，，则。

S35、优化损失函数，特征学习和聚类划分后，以反向传播的形式优化聚类中心和样本分布，提高嵌入空间向量的特征质量，优化的损失函数涉及三个，重构损失、聚类损失和重参数采样的损失：

为了尽可能的在特征空间中保留数据的局部结构信息，保留解码器结构，使用控制聚类损失扭曲嵌入空间的程度，损失函数定义为：

1. 根据权利要求1所述的基于正则变分嵌入式的软件需求聚类方法，其特征在于，步骤S4所述的聚类输出模块具体为：待达到聚类截止条件，输出最终的聚类划分结果，以“少数服从多数”的原则计算统计每个类的样本，同时使用Silhouette coefficient、Calinski Harabasz和Davies-Bouldin index等评价指标来衡量方法好坏。

**一种基于正则变分嵌入式的软件需求聚类方法**

**技术领域**

本发明设计自然语言处理中的数据挖掘领域，尤其是在无监督学习中对未标记的样本进行相似性划分。

**背景技术**

软件开发流程包括需求分析、系统设计、详细设计、测试和评估，想要设计出一款好的软件，首要工作便是需求分析，但实际开发过程中，人们忽略了需求分析的重要性，将关注点放在了设计阶段。需求分析引起的错误在软件开发过程中是看不到的，只要在测试阶段才会被发现，但这时修改错误将付出双倍的代价。软件需求描述不仅是用户和开发者沟通的桥梁，更是功能设计和性能指标的依据，它贯穿于整个软件开发过程，需求会随着时间推移发生变更，这会给后期开发带来巨大的风险。

需求分析阶段往往存在以下问题：（1）用户和开发人员所在研究领域的差异，两者熟悉自己的领域但是却对对方的领域很陌生，导致用户和开发人员需求沟通存在障碍；（2）软件需求描述中潜在需求未能挖掘出来，由于用户理解的计算机领域的专业知识有限，所要求的功能和性能设计表达不完整，导致在需求分析时可能遗漏；（3）软件需求文本描述模糊，存在稀疏性、歧义性、不可验证性等问题，同种意思的文本表达在不同领域的理解也有所区别，文本描述简短的缺点具体体现在功能描述冗余，开发流程繁琐，模块耦合性较高等方面，导致用户体验感和可操作性较差，软件成本和开发效率无法得到保证，从而造成项目失败。

需求分析阶段的问题给软件开发造成了不便，如果我们通过聚类方法将相似性描述聚集起来，我们只需要关注同一个类内的描述，那我们就可以指导这个类代表的是软件哪方面的功能以及应用领域。传统的文本聚类方法首先要进行文本预处理，然后将对预处理过后的单词序列进行特征提取，再由词向量工具映射到向量空间上，基于度量函数完成样本划分。传统的特征提取技术是基于统计分析的，利用评估函数为已有的特征参数进行权值分配，特征权重代表了单词在整句话中的重要程度，但是这种方式往往忽略了序列的位置关系和语义信息，文本中的单词被当作孤立的个体，仅依靠频率无法准确表达语义，忽略了上下文关系。

常用的聚类方法大部分基于划分法、密度法和层次法，例如K-means、DBSCAN、Agglomerative Clustering等。已知的聚类方法在文本领域和图像领域都达到了不错的效果，但这未必能说明聚类方法适用于所有的领域知识划分，传统聚类还是极其依赖于所提取特征的质量和词向量模型的训练。深度学习的引入开始将工作集中在学习表征上，以深度模型学习文本的特征表达，相比于传统的特征提取非线性和拟合能力更强，所提取的特征在传统聚类上有大幅度提升，通过网络参数的调整，可以取得较高的聚类准确率。但是大部分深度聚类是基于两段式聚类，首先对文本进行特征提取，然后将压缩特征由传统聚类划分。两段式聚类结构清晰，特征提取器将高维特征压缩成低维特征，降低了数据稀疏性，包含了更多的文本语义，对于聚类具有更好的连续性和解释性，但是两段式聚类的缺点也很明显，就是表征学习和聚类过程分两个步骤进行，聚类中心和提取的特征不能根据聚类结果改善，且还是存在易陷入局部最优的缺点，前向聚类过程是一次性的，只能小幅度改善聚类输出。

综上所述，现有的技术问题为：由于软件需求文本存在离散度高、噪声大和数据稀疏等特点，传统的文本特征提取技术所抽取的特征维度高，单词序列的前后位置关系未考虑进去，因此不能准确表达语义，仅依靠词频贡献或词与类的关系抽取特征容易丢失重要信息，例如，低频词可能在整个文本序列具有表征意义，但因频率低去掉则会使文本表达失去象征意义。

传统的文本聚类方法易受初始聚类中心的影响，且大部分基于距离度量，方法依旧存在易陷入局部最优的缺点，深度聚类结合了深度学习技术和传统聚类方法，模型结构容易了解，聚类结果相较于传统聚类有小幅提升，但是大部分深度聚类方法基于两段式聚类，不能反向传播优化聚类中心和样本分布。

技术问题不能有效解决的原因：目前有关软件需求文本的聚类方法很稀少，大部分聚类方法重点关注方法的改进，而忽略了特征提取方式的优化，实际情况是并不是所有的聚类方法都具有普适性，往往都是根据数据分布而采取使用某种特征提取方式和聚类方法，而特征提取器往往也是难以抉择的，从某种程度来说间接决定样本划分。由于属于无监督训练，词向量模型的训练也至关重要，但大规模的软件需求语料库很少，没有好的词向量，聚类方法将变得毫无意义。

解决的难度在于：软件需求文本是一堆杂乱无章的非结构化数据，存在诸多冗余信息，因此无法输入到机器中学习，我们需要对单词缩略词更改、拼写校正、词干提取和词形还原，传统的特征提取器是线性映射，结果缺乏合理的解释和描述，文本映射成计算机可以识别的向量形式，才可以进行聚类划分。软件需求数据经过分析后，根据软件需求数据的样本分布，判断采用哪种聚类方法以及特征提取器，以及反向传播的神经网络聚类模型如何设计，同时我们还要保证特征向量不会受聚类损失导致嵌入空间损坏，保留样本的局部结构。

解决的意义在于：软件需求文本聚类为软件质量提供了可靠的保障，最大限度地减少了需求分析的风险，降低了软件开发的成本。国内外对于软件需求文本的特征挖掘研究尚少，同时聚类方法过于单一，对于不同的功能性需求划分模糊，对于噪声模型鲁棒性较差。而采用本发明可以有效解决上述问题，聚类结果方便用户和开发者进行功能和性能设计上的沟通，在需求分析时可以挖掘潜在用户需求，避免了软件设计阶段功能冗余或者功能欠缺，明确的需求定义决定了的软件的生存周期和质量，减小了开发风险，是软件开发成败的关键因素。

**发明内容**

针对软件需求文本存在离散度高、噪声大和数据稀疏等特点，目前有关聚类的工作局限于单一类型的文本，鲜有考虑软件需求的功能语义，本发明提出了一种基于正则变分嵌入式的软件需求聚类方法，将特征学习和聚类方法同时进行，优化特征损失、重参数损失和聚类损失，学习样本分布，取得最优聚类结果。

本发明所采用的方案是，基于正则变分嵌入式的软件需求聚类方法，包括：

S1、数据收集模块，本发明的数据集在网上很稀少，所以利用Scrapy爬虫技术获取Softpedia网站下的软件需求数据；

S2、数据预处理模块，对获取的软件需求数据进行文本预处理，然后句向量模型将软件需求文本映射到向量空间；

S3、模型训练模块，对于处理好的句向量输入到正则变分嵌入式聚类模型中，通过编码将高维稀疏特征压缩成低维密集向量，以概率形式生成于原始数据类似的样本，用来反映数据本质特征，然后新生成的向量同时进行特征学习和聚类划分；

S4、聚类输出模块，待模型达到终止条件，输出最终的聚类结果以及多种评价指标。

以上方法，首先分析软件需求文本的特征，主要从样本分布决定特征提取器和聚类方法的选择。数据预处理阶段，经过文本处理，得到符合语法规则和句序的软件需求文本；使用句向量工具将软件需求文本处理成了计算机可以识别的形式，句向量包含了整个句子的文本信息，考虑了语法结构和单词前后位置关系，对于聚类方法的性能有很大的提升。

其次，将软件需求句向量输入到正则变分嵌入式聚类方法中，解决了特征学习和聚类输出共同优化的问题，输入端融合正则化避免了噪声和过拟合问题，增强模型鲁棒性，嵌入空间采用重参数技巧反映了原始数据的本质特征，隐变量引入概率视角，用于描述数据的显著特征。

最后，通过反向传播优化模型损失函数，引入扭曲因子控制因聚类损失而引起的特征空间的破坏，待达到迭代停止条件输出最终的聚类结果。

进一步的，所述步骤S1包括：

利用Scrapy爬虫技术获取Softpedia网站下的Windows平台的11类数据集，由于防爬程序设置IP代理池和时间延迟，每类数据以csv的格式单独存储，同时对每类数据集中的样本进行标注。

进一步的，所述步骤S2包括：

S21、文本预处理，由于软件需求数据是直接在网站上获取的，无法避免html标签内容，首先使用正则表达式re.compile(“/<V?.+?V?>/g”)等去除英文文本中的html标签；然后更改文本中的缩略词，例如re.sub(r"can\'t", "can not", text)，使用[pyenchant](http://pythonhosted.org/pyenchant/)库拼写检查，找出错误后改正；使用SnowballStemmer完成词干提取，WordNetLemmatizer完成词形还原,最终将所有类的数据放在一个csv文件中。

S22、文本向量化，以文本中每个句子为单位，使用BERT预训练的句向量模型将句子映射到向量空间，句向量代表整个句子的文本信息，考虑了单词前后位置关系和内部关联。

以上方法，使用正则表达式校正软件需求文本，尽量去除噪声数据，然后使用将文本映射成向量形式，便于计算机处理

进一步的，所述步骤S3包括：

步骤S31、Dropout正则化，在模型输入端融合Dropout正则化，随机停止一些神经元不参与训练，其余神经元共同工作，简化了网络模型，降低了时间复杂度，消除减弱了神经元节点间的联合适应性。

步骤S32、编码，采用全连接层对高维稀疏化向量进行特征压缩，编码器网络将正则化向量转换为隐藏空间的两个参数，记作和，是一个服从正态分布的张量，通过重参数技巧以概率形式生成与原始样本类似的新样本，隐藏空间向量公式如下：

步骤S33、解码，将重参数形成的嵌入特征解码，解码同样采用全连接层，解码过程与编码过程相反，经过解码还原成原始向量。

步骤S34、聚类划分，嵌入特征作为K-means算法的输入，K-means按照近邻原则划分样本到最近的类中，即满足，，则。

步骤S35、优化损失函数，经过解码和聚类划分后，通过反向传播的形式优化损失函数，损失函数涉及三个，样本重构产生的损失，聚类产生的损失，重参数产生新样本的损失：

为了尽可能的在特征空间中保留数据的局部结构信息，保留解码器结构，使用控制聚类损失扭曲嵌入空间的程度，损失函数定义为：

以上方法，特征学习和聚类过程同时进行，保留解码器结构，优化特征质量，以良好的特征进行聚类，重参数技巧反映原始数据的本质特征，通过反向传播的形式优化定义的三种聚类损失。

进一步的，所述步骤S4包括：

经过反向传播，达到迭代停止条件后，输出最终的聚类结果，每个类内以“少数服从多数”的原则统计样本数最多的类别，这个类定义为样本数最多的类别，同时使用Silhouette coefficient、Calinski Harabasz和Davies-Bouldin index等评价指标来衡量方法好坏。

以上方法，输出最终的样本划分结果，计算整个数据集的准确率，同时采用其它三种聚类指标评价类内部紧密度和类间疏离度。

综上所述，采用了上述技术方案，本发明与背景技术相比，具有如下优点：

本发明能够让业务人员在软件需求文本中筛选出有价值的聚类特征，该特征可以作为功能设计的依据，聚类分群结果的呈现降低了开发设计的难度，整个过程操作更加简便，节约了资源耗损和时间成本。将深度学习技术和传统聚类结合，以反向传播的形式提高特征质量，从而进一步提高聚类准确率，特征学习和聚类过程同时进行，保留了局部结构和内部联系。

**附图说明**

本发明将通过例子并参照附图的方式说明，其中：

图1为本发明一种基于正则变分嵌入式的软件需求聚类的算法流程图。

图2为本发明一种基于正则变分嵌入式的软件需求聚类的网络架构图。

图3为实施例中Dropout正则化的示意图。

**具体实施方式**

本说明书中公开的所有特征，或公开的所有方法或过程中的步骤，除了互相排斥的特征和/或步骤以外，均可以以任何方式组合。

本说明书（包括任何附加权利要求、摘要）中公开的任一特征，除非特别叙述，均可被其他等效或具有类似目的的替代特征加以替换。即，除非特别叙述，每个特征只是一系列等效或类似特征中的一个例子而已。

如图1和2所示，本发明一种基于正则变分嵌入式的软件需求聚类方法，包括如下步骤：

S1、数据收集模块，由于网上关于软件需求描述的开源数据很少，所以采用Scrapy爬虫技术在Softpedia网站下获取Windows平台下的11类数据集，这11类数据分别是Antivirus、authoring-Tools、CD-DVD-Blu-ray-Tools、Compression-tools、Desktop-Enhancements、File-managers、Gaming-Related、iPod-Tools、Maps&GPS、Mobile-Phone-Tools、Network-Tools，由于防爬程序设置IP代理池和时间延迟，每类数据以csv的格式单独存储，同时对11类数据中的样本进行标注，类别名按照Ascii码排序，然后以数字0-10的形式给同类的样本中打标签，最红将标注好的11类数据合成一个文件；

S2、数据预处理模块，对获取的软件需求数据进行文本预处理，然后句向量模型将软件需求文本映射到向量空间；

S21、文本预处理，由于软件需求数据是直接在网站上获取的，无法避免html标签内容，标签的格式有以下几种：<div class="test"></div>，<img />，自定义标签<My-Tag></My-Tag>，使用正则表达式re.compile(“/<V?.+?V?>/g”)等去除英文文本中的html标签，表达式中<表示尖括号,第一个V?表示</div>这种标签的情况,第二个V?表示<img />这种情况；后更改文本中的缩略词，例如re.sub(r"can\'t", "can not", text)，使用[pyenchant](http://pythonhosted.org/pyenchant/)库拼写检查，找出错误后改正；使用SnowballStemmer完成词干提取，例如protecting提取后为protect，WordNetLemmatizer完成词形还原,最终将所有类的数据放在一个csv文件中；

S22、文本向量化，以文本中每个句子为单位，使用BERT预训练的句向量模型将句子映射到向量空间，句向量代表整个句子的文本信息，考虑了单词前后位置关系和内部关联；

S3、模型训练模块，对于处理好的句向量输入到正则变分嵌入式聚类模型中，通过编码将高维稀疏特征压缩成低维密集向量，以概率形式生成于原始数据类似的样本，用来反映数据本质特征，然后新生成的向量同时进行特征学习和聚类划分；

步骤S31、Dropout正则化，在输入端融合Dropout正则化，按照一定概率将神经网络的部分神经元停止工作，防止模型对数据中不重要的特征进行贪婪学习，这会造成特征在压缩过程中融合过多噪声，对文本聚类造成损失，随机抑制神经元相当于降噪，在模型学习时使模型具有鲁棒性，增强模型泛化能力。神经网络训练过程无法逃避两个问题：（1）费时；（2）容易过拟合。融合Dropout恰好解决了这两个问题，它随机停止一些神经元不参与训练，其余神经元共同工作，简化了网络模型，降低了时间复杂度，消除减弱了神经元节点间的联合适应性。假设输入向量集合为，其中代表每个样本，隐藏层向量为，重构向量为，网络的全连接层的激活函数为函数。在输入端融合正则化，通过Dropout随机映射成正则化向量：

步骤S32、编码，采用全连接层对高维稀疏化向量进行特征压缩，编码器网络施加一个约束项，让网络生成服从高斯分布的样本集，利用Reparameterization trick按照高斯分布里的均值和方差规则就可以取任意相关的数据，使嵌入空间学习原始数据分布，动态调节噪声的强度，然后解码层重构样本，生成与原始样本相似的数据：

步骤S33、解码，将重参数形成的嵌入特征解码，解码同样采用全连接层，解码过程与编码过程相反，经过解码还原成原始向量。

步骤S34、聚类划分，嵌入特征作为K-means算法的输入，K-means按照近邻原则划分样本到最近的类中，即满足，，则。每轮训练重新计算聚类中心：

利用分布刻画嵌入特征与聚类中心的相似度：

利用该相似度，定义目标分布为：

步骤S35、优化损失函数，经过解码和聚类划分后，通过反向传播的形式优化损失函数，损失函数涉及三个，样本重构产生的损失，聚类产生的损失，重参数产生新样本的损失：

若无解码器结构，优化式Cluster Loss的目标函数会损毁特征空间，导致学到无意义的表达和损害聚类的性能。为了尽可能的在特征空间中保留数据的局部结构信息，保留解码器结构，使用控制聚类损失扭曲嵌入空间的程度，损失函数定义为：

S4、聚类输出模块，经过反向传播，达到迭代停止条件后，输出最终的聚类结果，每个类内以“少数服从多数”的原则统计样本数最多的类别，这个类定义为样本数最多的类别，同时使用Silhouette coefficient、Calinski Harabasz和Davies-Bouldin index等评价指标来衡量方法好坏。

**下面以具体实验验证本发明方法的有效性。**

1.数据集

由于网上有关软件需求的公开数据集不完全或者稀少，本发明采用Scrapy爬虫技术获取Softpedia平台上的软件需求，这些需求都是对软件功能的客观描述。Softpedia平台为用户提供了各种系统平台下的各种工具，这些系统包括Windows、Linux、Android和IOS等几乎所有的平台。每个平台下有杀毒软件、压缩软件、文件管理、游戏软件和地图定位软件等几十种软件类别，不同平台下的工具使用描述也会有所差异。截至北京时间2019年5月27号，该网站总共收录应用程序有113016个，应用程序被下载的总次数为3320687342次。

本发明的软件需求数据为Windows平台下的软件功能描述，该平台下拥有不同类别的软件，每个软件类别下拥有大量的App，每个App的功能信息和作用都有对应的描述信息，我们一共爬取了11类软件数据，共15598条，数据长度各不相同，爬取到的软件类型和数量如表1所示。

表1软件需求数据表

| 软件类别 | 数量 |
| --- | --- |
| Antivirus | 625 |
| Authoring-Tools | 676 |
| CD-DVD-Blu-ray-Tools | 1545 |
| Compression-tools | 466 |
| Desktop-Enhancements | 7612 |
| File-managers | 546 |
| Gaming-Related | 114 |
| iPod-Tools | 174 |
| Maps&GPS | 25 |
| Mobile-Phone-Tools | 433 |
| Network-Tools | 3382 |

2、评价指标

本发明主要在聚类准确率方面进行对比，采用纯度作为聚类准确率，公式如下：

本发明除了在准确率上对比算法模型，还将在SC(Silhouette Coefficient)、CH(Calinski Harabasz)、DB(Davies Bouldin)等聚类指标上评价算法的优劣,公式如下：

3、实验结果和分析

本发明将不同长度的文本经过BERT训练后，每个样本特征维度是768维，由于文本长度不同所带来的向量不规则问题，且考虑了上下文语义信息，为了验证BERT句嵌入模型优于平均向量，本文在传统聚类算法DBSCAN，Spectral Clustering，Hierarchical Clustering，GMM，K-means和SOM与本发明进行对比，如图4所示。

根据图4可知，本文的软件需求数据比较适用于K-means和SOM聚类算法，对于其它传统聚类算法提升效果不明显，在SOM算法上，聚类准确率相比于平均向量提高了7.79%，在K-means聚类算法上，聚类准确率提升了5.16%，说明明句向量模型相比于平均向量确实可以有效提高聚类算法性能。在传统的聚类算法与本章模型相比下，可以观察到DVEC模型的聚类准确率是最高的，平均向量是60.11%，句向量是62.92%，相比于传统聚类准确率最高的K-means算法，在平均向量上提高了5.72%，在句向量上提高了3.37%，说明正则变分嵌入式聚类算法在学习特征表示的同时，其嵌入空间的向量可以提高聚类算法准确率，两者共同优化对算法性能具有很大的提升。同时，本章聚类算法模型也更加适用于软件需求文本数据。

本章除了在传统聚类算法上进行准确率对比，还将与比较流行的深度聚类算法AE+K-means、DEC、IDEC上进行对比，准确率对比如图5所示。

通过对比可以发现，AE+K-means的准确率最低，其原因在于自编码器只是压缩数据特征，减小了维度，将压缩特征由K-means聚类算法完成，降低了算法复杂度，但是却未做其它方面的改进，与本章模型相比，平均向量上低4.09%，句向量上低3%。DEC模型在平均向量基础上低于本章模型2.03%，在句向量基础上低于2.75%，本章模型相比于DEC并未去除解码器结构，维护特征局部结构，避免聚类损失影响特征空间。由于IDEC模型只是学习样本输入与输出的误差，未考虑向量的噪声和样本分布的问题，本文所提出的DVEC模型在输入端去除噪声，同时使用VEC学习样本分布和聚类划分。在Average Embedding上，DVEC模型的聚类准确率略低于IDEC，这是由于平均向量加权求和未能充分表达语义，造成准确率不增反降。在Sentence Embedding上，DVEC的效果是最好的，准确率高达62.92%，对比Average Embedding提高了2.81%，对比IDEC提高了1.14%，句向量考虑了词语前后位置关系，但未考虑噪声影响，DVEC输入端融合Dropout正则化解决了这个问题，同时利用重参数技巧使嵌入空间根据正太分布来重构样本，适用于数据量比较少的情况，其编码解码过程类似于生成对抗的过程，该过程使嵌入空间的特征向量提取效果较好，对于提高聚类算法准确率有效。

本章除了在准确率方面进行对比，也在SC、CH、DBI上评价聚类结果的性能，如表2所示。

表2 多种评价指标的对比

|  |  |  |  |
| --- | --- | --- | --- |
| Method | Silhouette Coefficient | Calinski  Harabasz | Davies  Bouldin |
| AE+K-means | 0.036 | 383.631 | 3.688 |
| DEC | 0.015 | 328.96 | 4.05 |
| IDEC | 0.024 | 328.66 | 3.97 |
| DVEC | 0.026 | 321.324 | 3.681 |

由表2可知，DVEC的Silhouette Coefficient仅次于AE+K-means，比DEC和IDEC的值都要高，说明了DVEC的类内距离比较小，而类间距离比较大，能较好的划分类别。在Calinski Harabasz评价指标上，DVEC的值在对比模型中最小，这也就表明了本章算法模型在类别内部数据的协方差最小，类之间的协方差最大，聚类划分效果最优。在Davies Bouldin评价指标上，DVEC模型的值同样也是最小的，值越小代表了不同类之间的相似度也最低，说明聚类划分比较清晰。



图1 算法模型



图2 正则变分嵌入式聚类

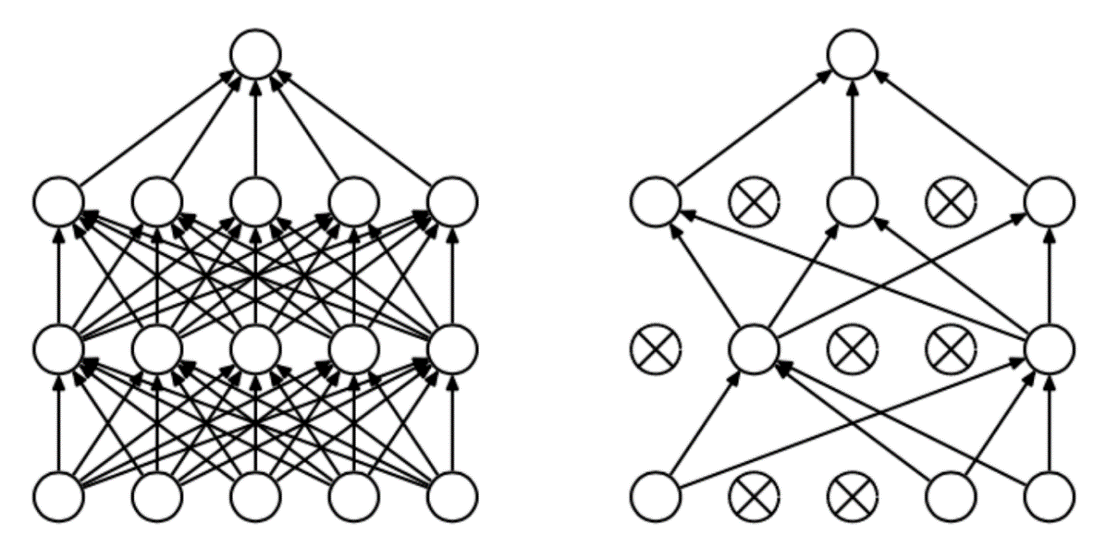


图3 Dropout正则化

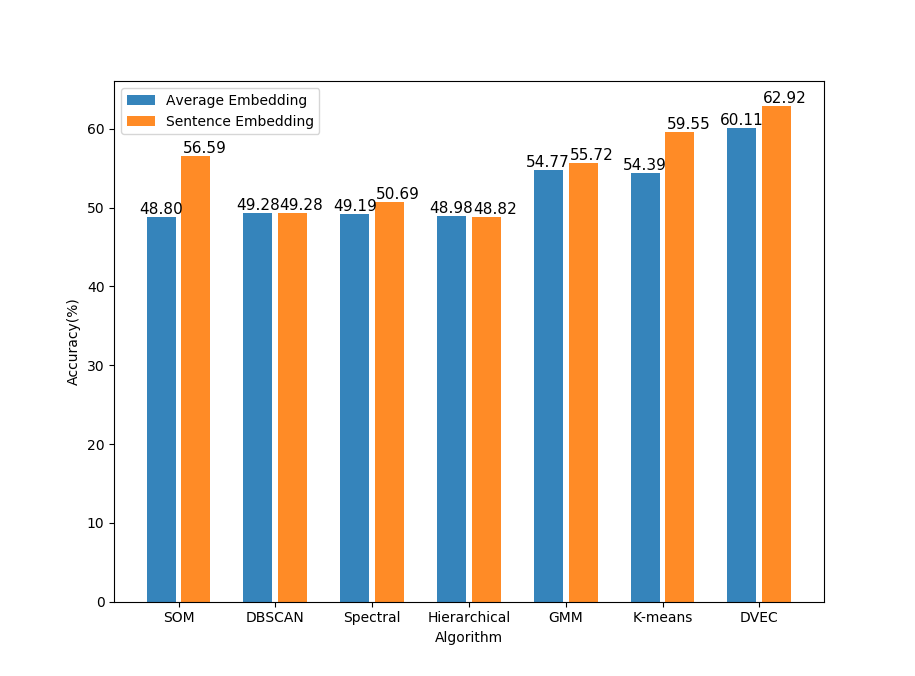


图4 传统聚类对比

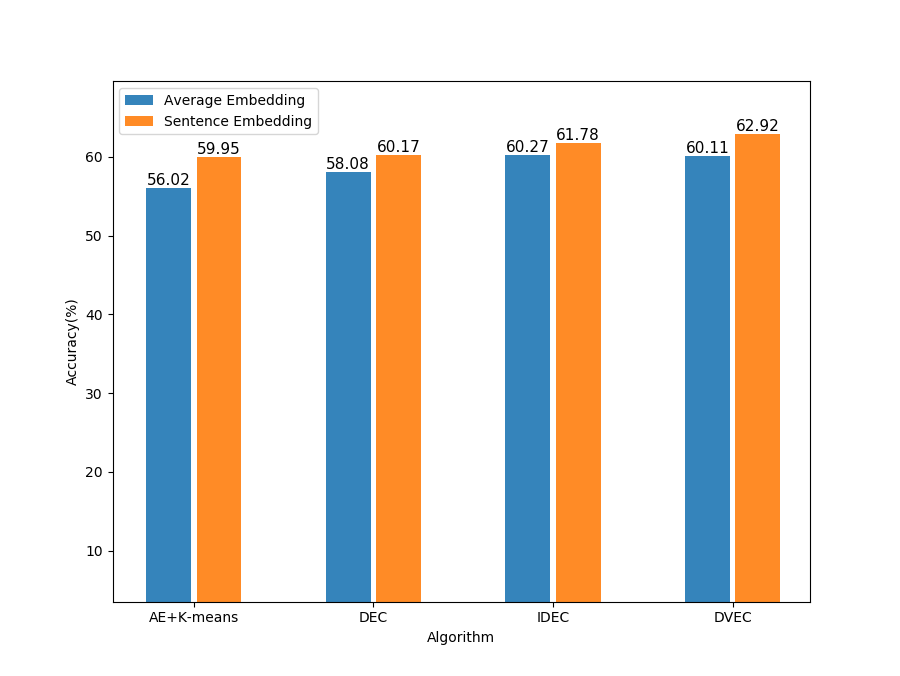


图5 深度聚类对比