计算机系统应用 ISSN 1003-3254, CODEN CSAOBN
Computer Systems & Applications,2022,31(6):158−166 [doi: 10.15888/j.cnki.csa.008496]

E-mail: csa@iscas.ac.cn
http://www.c-s-a.org.cn
Tel: +86-10-62661041

基于深度哈希的文本表示学习①

邹　傲, 郝文宁, 田　媛

(陆军工程大学 指挥控制工程学院, 南京 210007)
通信作者: 郝文宁, E-mail: hwnbox@163.com

摘　要: 文本表示学习作为自然语言处理的一项重要基础性工作, 在经历了向量空间模型、词向量模型以及上下文分布式表示的一系列发展后, 其语义表示能力已经取得了较大突破, 并直接促进模型在机器阅读、文本检索等下游任务上的表现不断提升. 然而, 预训练语言模型作为当前最先进的文本表示学习方法, 在训练阶段和预测阶段的时空复杂度较高, 造成了较高的使用门槛. 为此, 本文提出了一种基于深度哈希和预训练的新的文本表示学习方法, 旨在以更低的计算量实现尽可能高的文本表示能力. 实验结果表明, 在牺牲有限性能的情况下, 本文所提出的方法可以大幅降低模型在预测阶段的计算复杂度, 在很大程度上提升了模型在预测阶段的使用效率.

关键词: 深度哈希; 预训练语言模型; Transformer 结构; 文本表示学习; 深度学习; 注意力机制

引用格式: 邹傲,郝文宁,田媛.基于深度哈希的文本表示学习.计算机系统应用,2022,31(6):158–166. http://www.c-s-a.org.cn/1003-3254/8496.html

Text Representation Learning Based on Deep Hashing

ZOU Ao, HAO Wen-Ning, TIAN Yuan

(Command & Control Engineering College, Army Engineering University of PLA, Nanjing 210007, China)

Abstract: As a cornerstone of natural language processing, text representation learning has made a great breakthrough in its semantic representation ability when it undergoes the development of the vector space model, word embedding model, and contextual distributed representation. In addition, it directly promotes the continuous improvement of the performance of models in downstream tasks such as machine reading and text retrieval. However, as the most advanced text representation learning method, the pre-trained language model has high space-time complexity in the training and prediction stages, which results in a high threshold of use. Therefore, this study proposes a new text representation learning method based on deep hashing and pre-training, which aims to achieve as high a text representation ability as possible with less computation. The experimental results show that the proposed method can remarkably reduce the computational complexity and to a great extent improve the efficiency of the model in the prediction stage.

Key words: deep hashing; pre-trained language models; Transformer; text representation learning; deep learning; attention mechanism

文本数据是信息时代最常见的数据载体形式之一, 广泛存在于各种网页、书籍、报纸、期刊当中, 对文本数据的挖掘利用一直是学界多年来广泛关注的研究方向. 不同于一般的数值型数据, 采用机器学习的方法对文本数据进行处理首先需要将文本转换为计算机可处理的表示形式, 如向量 (vector) 或张量 (tensor), 一个好的表示形式需要尽可能地将原始数据中的语义特征保存在该表示所构成的高维向量空间中. 为不断提高机器学习系统的准确率, 需要一种算法能够自动地从输入样本中学习出有效的特征, 这种学习方式被称作表示学习 (representation learning).

文本表示学习的意义在于将非结构化的文本数据

① 基金项目: 国家自然科学基金 (61806221)
收稿时间: 2021-08-13; 修改时间: 2021-09-13; 采用时间: 2021-09-28; csa 在线出版时间: 2022-05-26

转换成计算机可处理的结构化文本数据. 对文本数据的表示学习可追溯到独热编码 (one-hot code) 为代表的词袋模型 (bag-of-words, BoW), 这种表示方法被称作局部表示 (local representation), 对语义特征的保留较低, 即使再引入词频-逆文本频率 (term frequency-inverse document frequency, TF-IDF) 等权重因子后仍然只能保留有限的语义特征. 2003 年 Bengio 等人的研究[1] 中采用了全连接网络训练一个语言模型, 这是将深度学习应用于文本数据的首次尝试, 在十几年的发展中, 基于深度学习的文本处理方法不断发展, 并在自然语言处理 (natural language processing, NLP) 中的各项子任务中都取得了较好效果. 在文本数据的表示学习方面, 基于深度学习的表示方法经历了从单词的分布式表示 (distributed representation) 到基于大规模预训练语言模型 (pre-trained language model, PLM) 的单词上下文表示 (contextual representation), 使得模型的表示学习能力取得了显著提升, 并推动在此基础上的各种算法在一系列下游任务中不断取得性能突破.

机器学习算法的性能在很大程度上依赖于数据表示的方法, 一个好的表示学习算法能够直接影响整个算法的表现, 有关表示学习的研究也逐渐从机器学习中的一个处理步骤逐渐演变为一个独立的研究方向.

在文本表示学习领域, 随着最新发布的预训练模型的参数数量呈指数增长, 基于大规模预训练深度模型的训练和使用成本对普通研究者来说越来越难以承受: 一方面, PLM 本身的训练需要大量语料和较大规模的算力资源作支撑; 另一方面, 即使可以通过现成的 PLM 获得较高质量的文本表示, 由于 PLM 的模型规模不断扩大, 通过 PLM 获得的单词实向量表示的维度已经达到了10^3的数量级, 依然有较高的算力门槛.

本文的主要工作在于提出了一种基于大规模预训练语言模型和深度哈希的文本表示学习方法, 以期算法所学习到的语义表示既能够包含较高质量的语义信息, 且在此基础上大幅减少其存储开销以及在下游任务中的计算开销. 通过充分的实验验证, 本文方法所学习到的文本表示在短文本检索、语义相似度匹配以及文本释义等任务中均取得了较好的性能表现, 并相较于以往方法获得了效率上的提升.

1 相关研究

本文提出的新的文本表示学习方法涉及深度语义挖掘和深度哈希的相关研究. 本部分将对文本语义挖掘和深度哈希的内容做简要介绍.

近年来, 文本语义挖掘的研究经历了 3 个阶段: 神经网络语言模型、词向量 (词嵌入) 和预训练语言模型. 文献 [1] 的研究中采用一个由全连接神经网络组成的模型进行语言模型任务的训练, 并将词向量矩阵作为模型训练的副成果, 这种词向量表示相较于传统基于统计的向量空间模型维度更低, 且能够在高维空间中保留更多的语义信息. 词向量技术是将文本表示学习直接作为模型的训练目标, Mikolov 等人所提出的 Word2Vec 模型是最早关于词向量的研究[2–4]. 该模型采用连续词袋模型 (continuous bag-of-words, CBOW) 和跳字模型训练作为词向量的训练任务. 之后也出现了许多对词向量进行改进的研究[5–7], 也有后续工作将词向量的方法用于更高文本单位的表示学习[8]. 预训练语言模型的出现是文本表示学习的最新成果, 但其意义又超越了单纯的表示学习并成为 NLP 领域的第三范式[9], 与深度学习早期的基于单词嵌入的方法相比, 它能够更充分地挖掘文本中包含的语义信息[4,5,7]. 预训练语言模型不仅在各种任务的公开数据集 (如 GLUE[10]、SQuAD[11] 和 RACE[12]) 上取得了显著的结果, 而且在工业上也有许多成熟的落地应用.

根据模型结构和训练方式, 预训练语言模型可分为 3 种: 以 BERT[13] 和 RoBERTa[14] 为代表的自编码语言模型 (autoencoding language model), 以 GPT[15] 和 XLNet[7] 为代表的自回归语言模型 (autoregressive language model), 以及以 BART[16] 和 UniLM[17] 为代表的端到端模型 (sequence-to-sequence language model).

预训练语言模型强大的语义表示能力主要来源于其内部的 Transformer[18] 结构. 具体而言, Transformer Encoder 的内部作用主要是多头注意机制[19], 以及残差连接结构[20], 一系列研究表明, 该结构能够比卷积神经网络捕获更多语义信息, 并且相较于循环神经网络具有更快的训练速度, 并能够较好地解决以往方法对输入样本的长距离依赖不足的问题, 使其成为目前 NLP 领域中使用最广泛的特征提取器.

在过去的几年中, 哈希已经成为解决大规模机器学习问题的一种较常用的方法[21–27]. 概括地来讲, 这种方法一般采用人工设计或系统自动学习的哈希函数将数据从高维的分布式表示映射到汉明空间 (Hamming space) 的二进制表示, 基于二进制码的表示方法在具体

下游任务的训练和使用中能够显著降低数据存储成本和通信开销. 紧凑的二进制码能够将数据压缩到更小的存储空间中, 并依然能够保留足够的特征信息用于各种下游任务. 具体来说, 哈希学习的目标在于学习数据样本的汉明空间表示, 使得哈希码能够尽可能地保持数据样本在原始语义空间中的最近邻关系, 从而维持其相似性. 因此, 每个数据样本将由一个紧凑的二进制码编码, 并且原始特征空间中相似的两个点应能够映射到汉明空间中相似的两个点.

深度哈希已经在大规模图像检索任务中得到了广泛的应用. 这种基于哈希的检索方法的基本思想是构造一系列哈希函数, 根据这一组哈希函数将每个文本对象映射到一个二进制的特征向量. 将高维实数特征向量编码为低维紧致的二进制码可以显著加快语义相似度的计算并能节省内存中的存储空间. 现有的基于哈希的方法分为两大类: 独立于数据 (data-independent) 的和依赖于数据 (data-dependent) 的. 第一种方法通常使用随机的映射函数将样本映射到特征空间, 然后再得到二进制码, 代表方法是局部敏感哈希 (LSH)[28] 及其扩展方法欧式局部敏感哈希 (E2LSH)[29] 等. 第二种方法是数据驱动 (data-driven) 的, 使用统计学习方法学习哈希函数, 将样本映射为二进制代码. 其代表方法是谱哈希 (spectral hash)[30] 等.

自 2014 年以来, 在图像检索领域出现了一系列将深度神经网络与哈希函数相结合的方法[31–34], 被称为深度哈希. 深度哈希学习的方法同样是数据驱动的, 相较于传统机器学习方法所采用的若干个随机的哈希函数, 该方法在训练过程中能够通过深度神经网络自动地学习生成哈希函数, 并进而获得每个输入样本的唯一哈希表示.

2 基于深度哈希的文本表示学习

在自然语言处理中, 文本的表示学习是非常重要一个步骤, 为了提高机器学习系统在下游任务的准确率, 首先需要将输入样本转换为有效的特征, 或更一般性地称为表示 (representation). 围绕表示学习的是两个核心问题: 一是“什么是一个好的表示”; 二是“如何学习到好的表示”.

在传统机器学习时代, 文本的表示学习更多地被看作是一种特征获取的步骤, 即通过设计特定的特征工程手段获得每个输入文本样本的表示, 然后再将该表示作为后续解决特定下游任务模型的输入特征. 这种文本表示学习与模型在特定任务上训练被割裂成两部分的状况一直延续到深度学习的词向量时代. 无论是词向量 Word2Vec[2–4]、GloVe[5] 还是语句向量或是文档向量等类似的方法[8], 都是先根据特定任务专门训练出特定文本粒度单元的分布式表示, 再将其应用于具体任务的优化过程, 由于这些表示并不会在训练中得到更新, 因此也被称作静态表示 (static representation). 这类方法通过设计准则或训练任务依靠这些准则或任务来从输入样本中选取有效的特征, 而特征的学习和最终预测模型的学习是分开进行的, 因此即使学习到的特征质量很高也不一定可以提升最终模型的性能. 在本文中, 文本的表示学习与下游任务的优化目标被当作一个整体在训练中同时进行. 实验证明, 良好的特征表示可以捕捉输入样本的本质结构, 完成输入文本语义空间到高维向量空间的映射, 并在映射的向量空间中反映输入之间的语义关系.

在本文的工作中, 我们从自然语言处理领域的 3 个子任务: 短文本检索、文本相似性度量和文本释义分别探索基于深度哈希的文本表示学习. 初步实验也验证了深度哈希方法在文本深度表示中的可行性和有效性.

基于深度哈希的文本表示学习的基本结构如图 1 所示, 其核心思想是使用预训练语言模型作为模型的主干, 并利用其参数来初始化所提出的模型. 然后, 根据特定下游任务的特点, 在模型末尾添加与任务高度相关的哈希学习层和结果输出层, 从而构建完整的深度哈希模型. 具体来讲, 图 1 的模型可以接受单一或者成对语句这两种输入方式. 对于单一语句的输入, 模型首先使用一个堆叠的 Transformer Encoder 结构对其进行处理, 对于该步骤的输出, 池化层提供了 3 种池化方式以供选择: “[CLS]”池化、最大值池化以及均值池化. 若输入为成对语句, 则使用一个孪生堆叠 Transformer Encoder 结构对两个语句分别编码并输出. 在池化层之后, 模型由一个全连接层、一个哈希学习层和一个面向具体下游任务训练的输出层组成, 全连接层的目的是进一步扩大模型的表示空间, 哈希学习层由一个阈值函数组成, 作用是实现从实向量空间到汉明空间的转换. 面向下游任务的输出层将根据具体的训练任务, 如短文本检索、语义相似度匹配等进行不同优化函数的选择 (均方差或二值交叉熵), 计算结果将用于整个模型的参数更新.

在对特定下游任务进行微调的过程中, 根据模型输出和真实数据标签计算的优化目标可以同时学习和优化模型参数、哈希函数和每个输入的深度哈希表示. 我们期望通过这种学习方法获取一种高效的文本表示方法, 以便将其应用于各种下游任务, 并期望达到或接近当前最先进水平 (state-of-the-art, SOTA) 的性能, 尽管其在准确率等指标上的表现可能达不到最优水平, 但它在特征表示的空间成本和模型预测的时间成本方面具有较大的优势, 这也是选择深度哈希技术的主要原因.

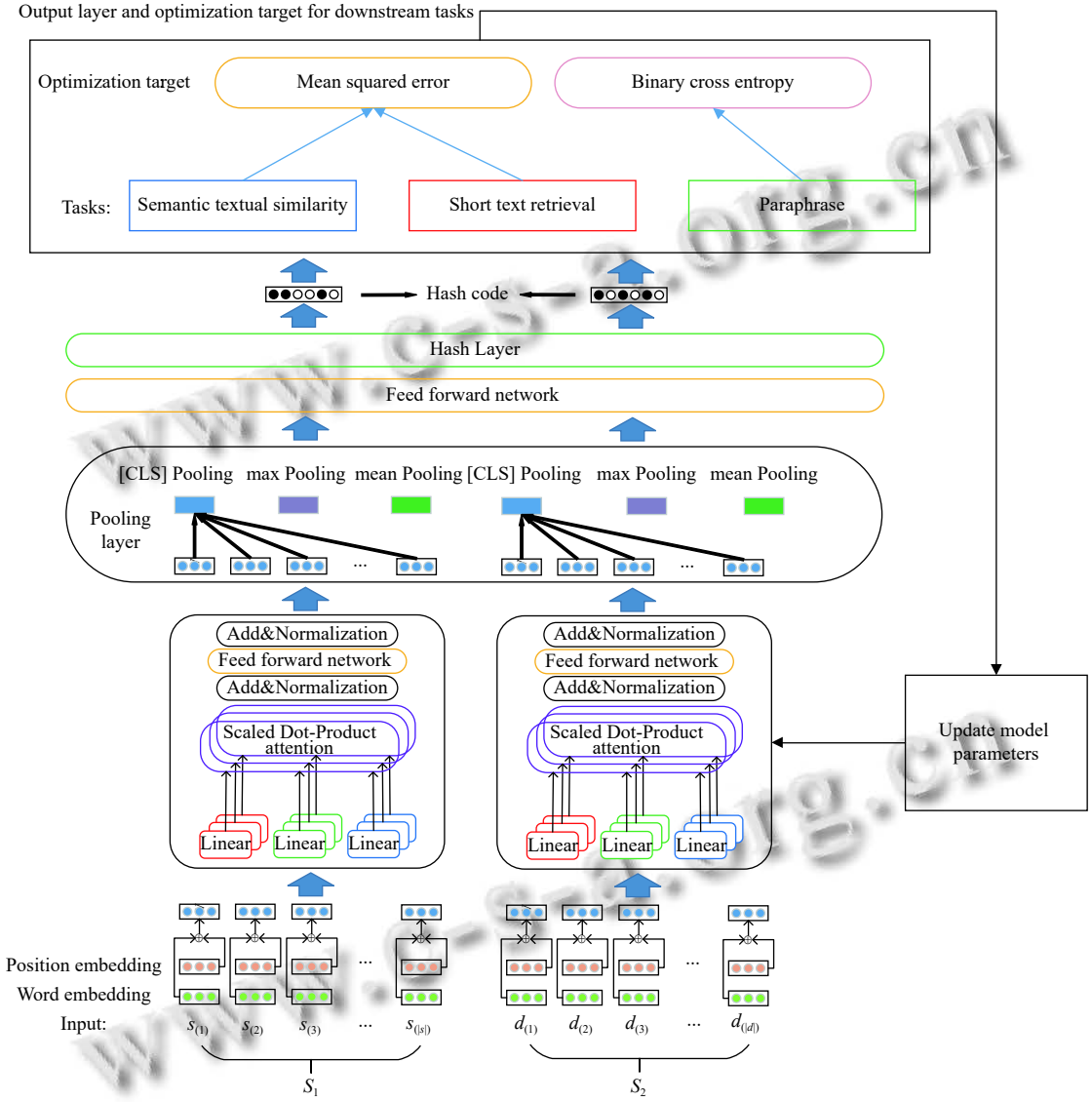

图 1 基于深度哈希的文本表示学习模型总体结构

在这项工作中, 使用了 Huggingface 提供的开源预训练语言模型库 Transformers[35]. 该语料库中的所有模型都经过大规模语料库的训练, 具有丰富的语义先验知识. 以实验中使用的一个模型 RoBERTa base 为例, 该模型使用 16 GB 的英语语料库进行 10 万次迭代训练. 实验中使用的其他预训练模型也使用类似级别的数据和训练进行预训练.

基于预训练模型的下游任务应用分为基于特征和微调两种方法. 基于特征是指使用语言模型的中间结果作为特征提取, 直接引入特定的下游任务作为输入; 微调是根据特定的下游任务修改模型的输出层, 添加少量与任务相关的参数, 然后在新的下游任务中重新训练整个模型的方法. 在实验中, 我们首先尝试将基于特征的方法应用于下游任务, 使用最后一刻 [CLS] 标

签输出的向量表示作为输入文本的向量表示 (我们还使用每个维度上输出层的最大池和平均池的方法来获得相应的表示), 然后, 通过设置阈值, 实数字段的向量表示被转换为散列码.

下游任务有多种形式的特定输入. 在短文本检索、文本语义相似度度量和释义任务中, 输入可以分为单个文本和文本对. 对于两种不同的输入, 分别采用单网络和双网络的模型结构. 建议的模型如图 1 所示. 以单一输入为例, 假设输入文本样本为$S=(s_{(1)},s_{(2)},\cdots,s_{(|S|)})$, 首先通过输入端的词向量映射矩阵和位置向量映射矩阵将其转化成初始分布式表示$X=(x_{(1)},x_{(2)},\cdots,x_{(len)})$, len是模型的超参数, 设置输入样本的序列长度, 若输入长度超过或小于len, 则通过截断或填充的方式将其长度限定在len. 其中每个$x_i(1\leqslant i\leqslant len)$计算如下:

$$x_i=e_{\text{word\_emb}}(s_{(i)})+e_{\text{position\_emb}}(s_{(i)}) \tag{1}$$

随后, 输入样本初始化后的分布式表示将输入到堆叠的 Transformer 结构中. Transformer Encoder 的第一部分为多头自注意力层, 假设模型隐层维度为d_{mdl}, 注意力头的个数为n_{head}. 对每一个注意力头$head_i$的计算, 需要对应的一组线性变换矩阵$W_i^Q\in\mathbb{R}^{d_{mdl}\times d_k}$, $W_i^K\in\mathbb{R}^{d_{mdl}\times d_k}$以及$W_i^V\in\mathbb{R}^{d_{mdl}\times d_v}$, 其中$d_k=d_v=d_{mdl}/n_{\text{head}}$. 对于每个输入样本$X\in\mathbb{R}^{len\times d_{mdl}}$, 根据线性变换矩阵可得$Q_i=X\times W_i^Q$, $K_i=X\times W_i^K$以及$V_i=X\times W_i^V$. 然后可得$head_i$计算如下:

$$head_i=Softmax\left(\frac{Q_iK_i^T}{\sqrt{d_k}}\right)V_i \tag{2}$$

最终多头自注意力层的输出将是所有注意力头的拼接再经过一个线性转换层的结果:

$$M_{\text{attention}}=Concat\left(head_1,head_2,\cdots,head_{n_{\text{head}}}\right)W^O \tag{3}$$

其中, $W^O\in\mathbb{R}^{n_{\text{head}}d_v\times d_{mdl}}$是多头自注意力层输出时的线性变换矩阵.

现假设输入样本$S=(s_{(1)},s_{(2)},\cdots,s_{(|S|)})$, 最后一个 Transformer Encoder 层的输出为$Z=(z_{(1)},z_{(2)},\cdots,z_{(len)})$. 为生成整个输入文本的表示, 本文采用了 3 种池化策略, 即 [CLS] 池化、最大值池化和平均值池化. 其中 [CLS] 池化的策略为直接选取输入端所添加的 [CLS] 所对应输出的向量即$z_{([\text{CLS}])}$作为文本的初步表示:

$$CLS\_Pooling(z_{(1)},z_{(2)},\cdots,z_{(len)})=z_{([\text{CLS}])} \tag{4}$$

最大值池化即对输出的文本表示矩阵$Z\in\mathbb{R}^{len\times d_{mdl}}$按列依次选取最大值, 即对输出的表示空间的每一维都在输出值中选取最大值:

$$Max\_Pooling(z_{(1)},z_{(2)},\cdots,z_{(len)})=z_{\max} \tag{5}$$

其中, $z_{\max}\in\mathbb{R}^{d_{mdl}}$的每一维都满足$z_{\max,i}=\max(z_{(1),i},z_{(2),i},\cdots,z_{(len),i})$.

同理, 平均值池化即对输出的文本表示矩阵$Z\in\mathbb{R}^{len\times d_{mdl}}$按列依次选取平均值:

$$Mean\_Pooling(z_{(1)},z_{(2)},\cdots,z_{(len)})=z_{\text{mean}} \tag{6}$$

至此, 我们可以获得每个输入的文本样本S在经过预训练语言模型后的初步表示z. 在哈希层, z将与下游任务特定训练目标的优化过程中实现从实向量空间到汉明空间的映射, 具体细节将在下一节中对各个任务进行分别实现时阐明.

3 实验设计与结果分析

实验选取了 NLP 领域的短文本检索、语义相似度匹配以及文本释义 3 个子任务进行实验, 并在所有的这些任务上都分别进行了中文语料数据集和英文语料数据集的实验, 从而证明本文所提出的方法具有较强的普适性.

3.1 短文本检索

在本任务中采用 MRPC[36] 数据集和 GLUE[9] 中的 STS-B 数据集作为实验的英文数据源. 这两个数据集属于自然语言理解领域, 属于文本相似性任务, 不能直接用于本实验. 因此, 我们需要对数据集进行预处理以适应这个实验. MRPC 数据集本身是一个句子级的相似性匹配问题, 其中输入是一个句子对, 输出是一个标签, 用来标记两个输入句子是否相似. 对该数据集进行修改的方法如下: 在训练集和验证集中, 分别集成判断为“1” (即相似) 的句子对和判断为“0” (不相似) 的句子对. 类似地, 在 STS-B 数据集中被判断为“5” (相似) 和“0” (不相似) 的句子对可以分别被集成. 经过上述预处理, 共获得 5 173 条数据. 在实验中, 我们按照 80%、10% 和 10% 的比例将数据集分为训练集、验证集和测试集. 在实验中, 深度哈希模型采用了图 1 中设计的架构, 模型在短文本检索任务中的结构如图 2 所示. 在进行训练后, 对于任意短文本的输入, 模型能够输出其对应的唯一哈希表示. 将查询文本的哈希码与全部带检索文本的哈希码进行汉明距离的比较, 选取与其汉明距离最近的 K 个文本组成 top-K 被检索文本集并输出.

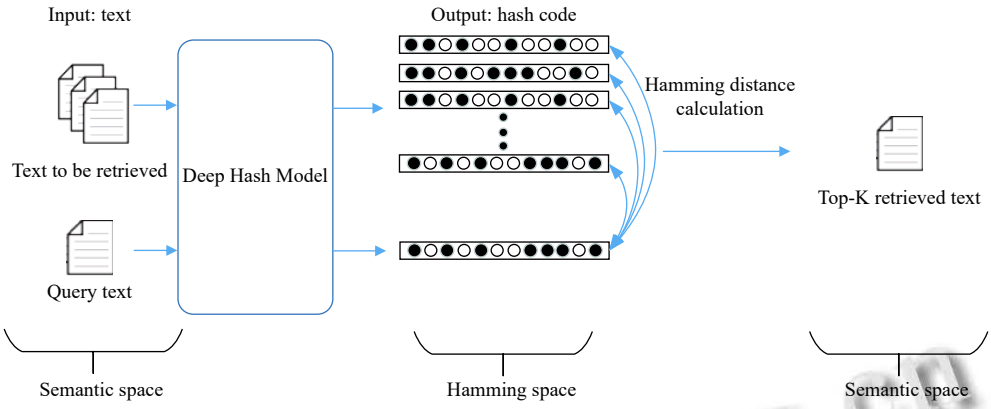

图 2 基于深度哈希表示的短文本检索流程

除在英文数据集上进行实验外, 本部分还使用 CLUE[37] 中文数据集中的 AFQMC 数据集进行了补充实验. 从格式上来看, AFQMC 的数据形式与 MRPC 数据集类似, 因此我们采用与 MRPC 相同的预处理方式进行处理, 并按照该数据集原本的划分将其分为训练集 (包含 34 334 个样本)、验证集 (包含 4 316 个样本) 和测试集 (包含 3 861 个样本).

在训练过程中, 采用成对输入相似度比较的方式进行间接训练, 假设一个训练样本包含的两个文本输入分别为S_1和S_2, 经过预训练语言模型层所获得的初步表示为z_1和z_2, 标签为$label \in \{0,1\}$, 则优化目标可表示如下:

$$L = \frac{1}{2}(z_1 - z_2)^2 \times (-1)^{label+1} \tag{7}$$

哈希层的作用是将表示从实向量空间映射到汉明空间, 在本文中采用设置阈值的方式进行二值化. 假设哈希层的输入为$z = \left(z_1, z_2, \cdots, z_{d_{mdl}}\right)$, 则输出为哈希码$h = \left(h_1, h_2, \cdots, h_{d_{mdl}}\right)$, 其中每一维的计算方式如下:

$$h_i\,(1 \leqslant i \leqslant d_{mdl}) = \begin{cases} 1, & h_i > \dfrac{1}{d_{mdl}} \displaystyle\sum_{j=1}^{d_{mdl}} h_j \\ 0, & h_i < \dfrac{1}{d_{mdl}} \displaystyle\sum_{j=1}^{d_{mdl}} h_j \end{cases} \tag{8}$$

当优化目标达到预期值, 就可以直接使用模型输出的哈希表示直接进行短文本检索任务. 具体来讲, 对于包含全部文本的待检索文本集$D = \{d_1, d_2, \cdots, d_{|D|}\}$, 采用基于深度哈希的模型进行表示学习可获得其哈希表示$H \in \{1,0\}^{|D| \times d_{mdl}}$. 假设查询文本$q$的哈希表示为$h \in \{1,0\}^{d_{mdl}}$, 则通过比较$h$与$H$每一行向量之间的汉明距离即可得出模型的预测结果.

3.2 语义相似度匹配

在文本语义相似性匹配任务中, 我们使用 GLUE 中的 STS-B 数据集, 该数据集是从新闻标题、视频标题、图像标题和自然语言推理数据中提取的句子对的集合. 每个句子对都由人工标注, 其相似度得分为 0–5. 任务的目标是预测输入句子对的相似度得分. 样本数量如下: 序列集 5 749、验证集 1 379、测试集 1 377.

如第 2 节所述, 文本相似性任务的输入是语句对, 利用均方误差损失 (mean square error loss, MSELoss) 目标函数优化模型参数, 从而可以同时学习显式的模型参数和隐式的哈希函数, 并进而根据隐式的哈希函数间接地获得每个输入样本的哈希表示. 第一种映射函数类似于所有传统的深度神经网络模型, 能够将输入文本从语义空间映射到高维实向量空间, 具有很强的相似性保持能力; 第二个映射函数是该模型所特有的, 它可以将输入从高维实向量空间映射到汉明空间. 因此, 我们可以获得一个自学习哈希函数和对应于每个输入的唯一哈希代码表示. 使用此哈希代码, 我们可以更有效地完成文本相似性任务.

在语义相似度任务中, 训练中的优化目标采用类似于 3.1 节中的均方差损失, 哈希层的机制参照式 (7) 的规则计算.

3.3 文本释义

在文本释义任务中, 本实验使用了 GLUE 的 MRPC 数据集. 根据惯例, 样本数量如下: 序列集 3 668、验证集 408、测试集 1 725. 发布该数据集的目的是鼓励在与释义和句子同义词及推理相关的领域进行研究, 并帮助建立一个关于正确构建训练和评估用释义语料库的论述.

同样地, 除英文数据集外, 本部分也采用了中文数

据集 AFQMC 进行补充实验, AFQMC 是与 MRPC 相同的子任务, 在数据格式完全一样, 只不过数据集全部采用中文语料.

尽管文本语义相似性任务和文本相似性任务都涉及到文本语义相似性, 但文本短语的输出是一个二元结果, 即“是” (用 1 表示) 或“非” (用 0 表示). 因此, 在任务的微调中, 我们不使用 mselo, 而是使用二进制交叉项的目标函数来优化模型参数, 隐式地研究了 hash 函数和相应的输入 hash 码. 类似地, 与短文本检索和文本相似性一样, 模型输出的哈希码之间的汉明距离也用于测试集中, 以获得模型的最终判断输出.

3.4 实验结果与分析

如上所述, 我们希望在每个下游任务中使用哈希代码来提高模型的效率. 根据前 3 部分描述的方法, 我们对 3 个任务的数据集进行了实验. 实验中使用的超参数设置如表 1 所示.

具体的实验结果如表 2 所示. 从表中可以看出, 本文在短文本检索、语义相似度匹配以及文本释义 3 种子任务的 5 个数据集上进行了充分实验, 其中短文本检索选取 top-5 结果的准确率作为评测指标, 语义相似度匹配选取相似度得分作为评测指标, 文本释义任务选取准确率作为评测指标. 除这些指标以外, 本文还对不同种类方法在预测阶段的时间进行了比较. 实验结果表明, 首先, 尽管微调方法需要对下游任务数据集进行额外的训练, 但它在性能上比基于特征的方法要好得多. 重要的是, 与动态上下文词嵌入方法相比, 文本表示学习的深度哈希方法在准确性和其他指标上有大约 3%–5% 的性能损失, 但空间成本和时间成本大大降低, 直接提升了模型在下游任务预测阶段的处理效率. 这在涉及大规模语义表示学习 (如文本检索) 的任务中尤其重要.

表 1 实验中采用的主要超参数

超参数	设置值
初始学习率	10^{-4}
文本表示维度	768
Transformer层数	24
Dropout	0.1
批次大小	32
优化策略	AdamOptimizer

表 2 基于深度哈希的文本表示学习在 3 个子任务上的实验结果

模型		短文本检索				语义相似度匹配		文本释义			
		英文数据集		中文数据集				英文数据集		中文数据集	
		Acc@ top5 (%)	Time (s)	Acc@ top5 (%)	Time (s)	Score	Time (s)	Acc (%)	Time (s)	Acc (%)	Time (s)
基于特征	BERT-base	35.7	—	31.4	—	56.3	—	61.3	—	59.4	—
	RoBERTa-base	36.1	—	33.5	—	59.7	—	64.5	—	61.3	—
	XLNet-base	34.5	—	31.7	—	58.4	—	62.7	—	59.9	—
基于微调	BERT	83.4	66.3	81.2	310.2	85.5	9.5	87.3	13.5	86.4	64.4
	RoBERTa	85.9	74.2	83.8	364.7	89.2	11.4	89.4	14.1	88.2	71.2
	Deep Hashing (BERT)	80.3	1.6	79.6	3.1	81.7	1.3	85.8	1.2	84.6	2.1
	Deep Hashing (RoBERTa)	81.4	1.7	79.2	3.3	84.5	1.4	87.1	1.2	86.9	2.3

4 结论与展望

本文探究了深度哈希技术在自然语言处理领域的一些应用场景. 其主要思想是使用深度哈希技术进行文本表示学习. 与传统的用高维实向量嵌入文本的深度学习方法不同, 本文设计了一种特定的深度神经网络模型. 在传统的深度神经网络模型的基础上, 增加了针对不同下游任务的哈希学习层和输出层. 最后, 可以通过训练学习每个输入样本对应的唯一哈希表示. 实验结果表明, 只要适当设计模型结构和训练过程, 可以在尽可能少的语义信息损失的情况下, 显著降低存储空间开销和计算时间开销, 从而极大地提高相关任务的处理效率.

参考文献

1 Bengio Y, Ducharme R, Vincent P, *et al*. A neural probabilistic language model. The Journal of Machine Learning Research, 2003, 3: 1137–1155.

2 Mikolov T, Chen K, Corrado G, *et al*. Efficient estimation of word representations in vector space. Proceedings of the 1st International Conference on Learning Representations. Scottsdale: ICLR, 2013.

3 Bojanowski P, Grave E, Joulin A, *et al*. Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics, 2017, 5: 135–146. [doi: 10.1162/tacl_a_00051]

4 Mikolov T, Sutskever I, Chen K, *et al*. Distributed

representations of words and phrases and their compositionality. Proceedings of the 26th International Conference on Neural Information Processing Systems. Lake Tahoe: Curran Associates Inc., 2013. 3111–3119.

5 Pennington J, Socher R, Manning CD. Glove: Global vectors for word representation. Proceedings of 2014 Conference on Empirical Methods in Natural Language Processing. Doha: ACL, 2014. 1532–1543.

6 McCann B, Bradbury J, Xiong CM, *et al*. Learned in translation: Contextualized word vectors. Proceedings of the 31st International Conference on Neural Information Processing Systems. Long Beach: Curran Associates Inc., 2017. 6297–6308.

7 Joulin A, Grave E, Bojanowski P, *et al*. Bag of tricks for efficient text classification. Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics. Valencia: ACL, 2017. 427–431.

8 Le Q, Mikolov T. Distributed representations of sentences and documents. Proceedings of the 31st International Conference on Machine Learning. Beijing: JMLR.org, 2014. II-1188–II-1196.

9 Liu PF, Yuan WZ, Fu JL, *et al*. Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing. arXiv: 2107.13586, 2021.

10 Wang A, Singh A, Michael J, *et al*. GLUE: A multi-task benchmark and analysis platform for natural language understanding. Proceedings of the 7th International Conference on Learning Representations. New Orleans: OpenReview.net, 2019.

11 Rajpurkar P, Zhang J, Lopyrev K, *et al*. SQuAD: 100, 000+ questions for machine comprehension of text. Proceedings of 2016 Conference on Empirical Methods in Natural Language Processing. Austin: The Association for Computational Linguistics, 2016. 2383–2392.

12 Lai GK, Xie QZ, Liu HX, *et al*. Race: Large-scale reading comprehension dataset from examinations. Proceedings of 2017 Conference on Empirical Methods in Natural Language Processing. Copenhagen: ACL, 2017. 785–794.

13 Devlin J, Chang MW, Lee K, *et al*. BERT: Pre-training of deep bidirectional transformers for language understanding. arXiv: 1810.04805, 2019.

14 Liu YH, Ott M, Goyal N, *et al*. RoBERTa: A robustly optimized BERT pretraining approach. arXiv: 1907.11692, 2019.

15 Radford A, Narasimhan K, Salimans T, *et al*. Improving language understanding by generative pre-training. 2018.

16 Lewis M, Liu YH, Goyal N, *et al*. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. Association for Computational Linguistics, 2020. 7871–7880.

17 Dong L, Yang N, Wang WH, *et al*. Unified language model pre-training for natural language understanding and generation. Proceedings of the 33rd International Conference on Neural Information Processing Systems. Vancouver: NIPS, 2019. 13063–13075.

18 Vaswani A, Shazeer N, Parmar N, *et al*. Attention is all you need. Proceedings of the 31st International Conference on Neural Information Processing Systems. Long Beach: Curran Associates Inc., 2017. 6000–6010.

19 Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. Proceedings of the 3rd International Conference on Learning Representations. San Diego: ICLR, 2015.

20 He KM, Zhang XY, Ren SQ, *et al*. Deep residual learning for image recognition. Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas: IEEE, 2016. 770–778.

21 Gionis A, Indyk P, Motwani R. Similarity search in high dimensions via hashing. Proceedings of 25th International Conference on Very Large Data Bases. Edinburgh: Morgan Kaufmann, 1999. 518–529.

22 Gu Y, Ma C, Yang J. Supervised recurrent hashing for large scale video retrieval. Proceedings of the 24th ACM International Conference on Multimedia. Amsterdam: ACM, 2016. 272–276.

23 Li P, Shrivastava A, Moore J, *et al*. Hashing algorithms for large-scale learning. Proceedings of the 24th International Conference on Neural Information Processing Systems. Granada: Curran Associates Inc., 2011. 2672–2680.

24 Liu HM, Wang RP, Shan SG, *et al*. Deep supervised hashing for fast image retrieval. Proceedings of 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas: IEEE, 2016. 2064–2072.

25 Liu W, Wang J, Ji RR, *et al*. Supervised hashing with kernels. Proceedings of 2012 IEEE Conference on Computer Vision and Pattern Recognition. Providence: IEEE, 2012. 2074–2081.

26 Shen FM, Shen CH, Liu W, *et al*. Supervised discrete hashing. Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). Boston: IEEE,

2015. 37–45.

27 Xia RK, Pan Y, Lai HJ, *et al*. Supervised hashing for image retrieval via image representation learning. Proceedings of the AAAI Conference on Artificial Intelligence, 2014, 28(1): 2156–2162.

28 Slaney M, Casey M. Locality-sensitive hashing for finding nearest neighbors [Lecture Notes]. IEEE Signal Processing Magazine, 2008, 25(2): 128–131. [doi: 10.1109/MSP.2007.914237]

29 Datar M, Immorlica N, Indyk P, *et al*. Locality-sensitive hashing scheme based on p-stable distributions. Proceedings of the 20th Annual Symposium on Computational Geometry. Brooklyn: ACM, 2004. 253–262.

30 Weiss Y, Torralba A, Fergus R. Spectral hashing. Proceedings of the 21st International Conference on Neural Information Processing Systems. Vancouver: Curran Associates Inc., 2008. 1753–1760.

31 Lin K, Yang HF, Hsiao JH, *et al*. Deep learning of binary hash codes for fast image retrieval. Proceedings of 2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops. Boston: IEEE, 2015. 27–35.

32 Yao T, Long FC, Mei T, *et al*. Deep semantic-preserving and ranking-based hashing for image retrieval. Proceedings of the 25th International Joint Conference on Artificial Intelligence. New York: AAAI Press, 2016. 3931–3937.

33 Lu JW, Liong VE, Zhou J. Deep hashing for scalable image search. IEEE Transactions on Image Processing, 2017, 26(5): 2352–2367. [doi: 10.1109/TIP.2017.2678163]

34 Zhang SF, Li JM, Zhang B. Semantic cluster unary loss for efficient deep hashing. IEEE Transactions on Image Processing, 2019, 28(6): 2908–2920. [doi: 10.1109/TIP.2019.2891967]

35 Wolf T, Debut L, Sanh V, *et al*. Transformers: State-of-the-art natural language processing. Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations. Association for Computational Linguistics, 2020. 38–45.

36 Dolan WB, Brockett C. Automatically constructing a corpus of sentential paraphrases. Proceedings of the 3rd International Workshop on Paraphrasing (IWP2005). 2005.

37 Xu L, Hu H, Zhang XW, *et al*. CLUE: A chinese language understanding evaluation benchmark. Proceedings of the 28th International Conference on Computational Linguistics. Barcelona: International Committee on Computational Linguistics, 2020. 4762–4772.