文档相似度算法比较

Nicholas Gahman 和 Vinayak Elangovan

宾夕法尼亚州立大学阿宾顿分校，宾夕法尼亚州，美国，科学与工程部
计算机科学项目

摘要

文档相似度是自然语言处理的重要组成部分，最常用于抄袭检测和文本摘要。因此，找出总体上最有效的文档相似度算法，可能会对自然语言处理领域产生重大积极影响。本报告旨在考察众多文档相似度算法，并确定哪些算法最为实用。报告将文档相似度算法分为三类，以探讨其中最有效的算法：统计算法、神经网络以及语料库／知识型算法。我们还使用一系列基准数据集和评估方法，对每个类别中最有效的算法进行比较，以测试各算法可能适用的所有方面。

关键词

自然语言处理，文档相似度

1. 引言

文档相似度分析是一项自然语言处理（NLP）任务，通过分析两篇或多篇文档来识别它们之间的相似之处。文档相似度广泛应用于文本摘要、推荐系统、抄袭检测以及搜索引擎。根据内容识别两篇或多篇文档之间的相似或差异程度，是文档相似度分析的主要目标。余弦相似度、欧几里得距离等常用技术会比较文档的文本特征，并给出介于 0 和 1 之间的相似度得分，其中 1 表示完全相似，0 表示完全不相似。尽管文档相似度领域使用了众多算法，但对于特定领域中哪种算法最有效或最高效，尚无普遍认可的结论。

本文旨在通过将文档相似度算法分为三类来解决这一问题：统计算法、神经网络以及语料库／知识型算法。我们将仔细考察每一类中的算法，并从中选出该类别中表现最佳的若干算法。确定各类别中最有效的算法后，将使用四个不同的数据集和三种不同的指标对其进行评估。最后，我们将分析所得数据，以确定哪些算法最为有效。

本文结构安排如下：第 2 节“相关研究”讨论了多项类似研究。第 3 节“所提方法”概述了三类文档相似度算法，并对所用方法和比较过程进行了更深入的讨论。最后，第 4 节“结果”介绍了具体内容以及

各算法实现的局限性，并给出比较结果，最后在第 5 节总结结论。

2. 相关研究

Jesus M. Sanchez-Gomez 与其他研究人员比较了用于抽取式多文档文本摘要的词项加权方案和相似度度量 [1]。这项研究的结果表明，总体而言，余弦相似度似乎是该领域最有效的统计相似度度量。Kazuhiro Seki 创建了一种能够跨语言使用的文本相似度度量 [2]。该方案通过使用两个神经机器翻译模型构建词嵌入，并据此计算文档相似度。相关神经机器翻译模型会生成多种可能的译文，以应对误译。翻译结果形成矩阵，经归一化和转换后生成多语言词嵌入，可使用余弦相似度进行比较。该方案具有很强的灵活性，研究明确指出，还可以在这一系统中加入其他相似度分数。研究人员将该方案与 Doc2Vec、Sec2Vec 和 S2Net 孪生神经网络等其他多语言相似度算法进行了比较，结果表明该方案优于所有其他算法。将该方案的句子检索结果与谷歌翻译进行比较时，两者表现相近；尽管由于训练数据和模型相对较少，该方案的 BLEU 分数低得多。

Emrah Inam 提出了一种文档相似度算法，该算法通过名为 ConceptNet 的语义词典，将词嵌入与基于知识的度量相结合 [3]。句子经过 ConceptNet 处理后，该方法使用广度优先遍历算法，为转换后的句子生成向量表示。最后，使用软余弦相似度度量比较向量化后的句子。随后，将 ConceptNet 度量得出的相似度与依存句法分析模型得出的相似度相结合，得到最终的相似度得分。该方法名为 SimiT，并以微软研究释义语料库（MRPC）作为评估任务，用于确定其能否检测并理解释义。其结果通过皮尔逊相关系数进行衡量，并与其他几种相似度算法的结果进行比较。这些算法包括基本余弦相似度、Word2Vec 余弦相似度和软余弦，以及 BERT 及其变体等几种最先进的方法。在参与比较的算法中，SimiT 表现出色，性能超过了除最先进方法之外的所有算法。即使与最先进的方法相比，SimiT 仍然很有用，因为它的运行时间极短。

表 1. 所选数据集的任务和用途。

	MRPC	**AFS**	SICK-R	SICK-E
基准测试用途	测试算法是否能够检测释义	测试是否算法能否检测相似的论点	测试是否算法能够识别词汇相似度句子的	测试算法是否能够识别语义相似性/差异之間句子

Aminul Islam 和 Diana Inkpen 合作提出了一种方法，使基于知识的度量能够确定句子相似度 [4]。基于知识的度量主要用于比较概念。将单词转换为概念很容易，但要将句子转换为概念则困难得多。作为该方法的一部分，作者提出了一种基于字符串相似度的单词方法，采用最长公共子序列的三种变体：归一化最长

公共子序列（NLCS）、从首字符开始的归一化最大连续最长公共子序列（$NMCLCS_1$），以及从任意字符开始的归一化最大连续最长公共子序列（$NMCLCS_n$）。将这三种改进的最长公共子序列算法各自得出的分数合并后再除以三，从而得到一个介于 0 和 1 之间的平均值。

为开发一种使用基于知识的度量的句子相似度算法，作者首先对句子进行处理，删除所有特殊字符、标点符号、大写字母和停用词。此时，将两个句子的长度保存下来以供后续使用。随后，他们使用基于知识的度量或字符串相似度度量，找出两个句子中相互匹配的所有单词，统计匹配单词的数量，并从句子中删除所有匹配的单词。接着，使用基于知识的度量比较所有剩余单词，并将结果放入一个矩阵中；该矩阵的长度等于第一个句子的长度，宽度等于第二个句子的长度。然后采用相同的方法创建另一个矩阵，只是将基于知识的度量替换为字符串相似度方法。将基于知识的度量矩阵与字符串相似度矩阵相加，得到联合矩阵。接下来，找出联合矩阵中的最大值并将其加入列表，然后删除该值所在的列和行。重复此过程，直到最大值为零，或矩阵中不再有行或列。最终，用于确定两个句子相似度的算法是：匹配单词数加上列表中的所有值之和，再乘以两个句子去除停用词后的总长度。然后，将结果除以第一个句子去除停用词后的长度的两倍，再乘以第二个句子去除停用词后的长度。表2总结了不同算法类别的优缺点。

表2。算法类别。

	优点	缺点
基本统计 技术	简单，易于使用	未能获得足够的语义 信息来做出准确 预测，判断蕴含还是 矛盾
神经网络	效果极佳，能够实现 最先进的结果	计算开销极高 而且占用内存较多，难以 调试
基于知识/语料库的 度量方法	可以轻松获取并 处理语义 信息，从而实现 准确预测	严重依赖大型语料库和 语义网络才能正常工作， 专为词语/概念相似度设计 因此难以扩展

Yigit Sever 和 GonencErcan 通过构建一个涵盖七种不同语言的跨语言文本相似度评测数据集，对跨语言语义相似度方法进行了比较 [5]。词网用于衡量概念；有些概念为特定语言所独有，而另一些则为多种语言共享。通过将其中若干同义词集相互关联，为构建评估数据集奠定了基础。该数据集可用于评估无监督和有监督文本相似度算法的效果，从而便于比较这两类算法。词嵌入会受到枢纽性问题的影响，即许多向量嵌入与其他向量嵌入相似。为探究这类枢纽如何影响相似度结果，评估任务被划分为两项不同的任务：对齐和伪检索。

除构建评估数据集外，论文还比较了若干无监督和有监督文本相似度模型。测试的算法包括机器翻译单语基线（MT）、句子嵌入（SEMB）之间的余弦相似度、词移距离（WMD）、Winkhorn（SNK）以及孪生长短期记忆网络（LSTM）。在检索任务中，WMD 的表现出人意料地最佳，其次是 SNK。LSTM 排名第三，但值得注意的是，它仅使用数量有限的实例进行训练，因此在这方面仍有可能取得更好的成绩。在匹配任务中，SNK 表现最佳，其次是 WMD，再次是 SEMB。

Yuki Arase 和 Junichi Tsuji 开发了一种通过迁移微调改进 BERT 模型的方法 [6]。他们的预训练方法侧重于语义对建模，使所提出的方法相比常规 BERT 模型取得了显著提升。此外，该方法不仅优于基线 BERT 模型，而且成本效益更高，因为它侧重于可自动生成的短语对齐。
作为评估过程的一部分，论文使用了两种不同的基准测试：GLUE基准和 PAWS数据集。所使用的 GLUE基准包含九个不同的评估数据集，其任务涵盖自然语言理解的多个方面。不过，与论文最相关的任务是语义等价，其子任务包括释义理解和语义文本相似度（STS）的理解。与 MRPC数据集一样，PAWS数据集侧重于释义理解，但通过受控的词语交换和回译，判断算法对语境和词序是否敏感。在 GLUE基准中，所提出的方法除 QNLI 和 SST 外，在所有方面的表现都优于对应的 BERT 方法；而在 PAWS 基准测试中，该方法在所有方面都优于 BERT。此外，评估结果显示，使用较小的微调训练语料库时，所提出的方法表现更好，这也让创建此类训练语料库变得容易得多。

3. 所提方法

统计技术是三类文档相似度算法中最简单的一类。它们首先将句子转换为向量，再比较这些向量，以此比较文本。最常用且最有效的比较方法是余弦相似度，但有时也会使用欧几里得距离等其他方法。在可选的预处理器中，本文认为 Sentence-BERT 最为理想。这是因为在针对这一用途测试的算法中，它表现极佳，且计算效率排名第二 [7]。神经网络是另一种文档相似度方法，而且非常有效。基于神经网络的技术原理是，先向网络输入成对的训练文档，其中两篇文本要么彼此相似，要么彼此不同。通过理解语义信息，神经网络逐渐学会区分相似文本对和不同文本对。通常，文本会先经过分词器处理，使网络更容易理解其中的语义信息。BERT 和 XLNet 等一些模型经过预训练，因此网络无需针对每项任务进行长时间训练并使用海量语料库，也能做出准确预测。图 1 展示了使用统计算法进行文档分析的典型方法。

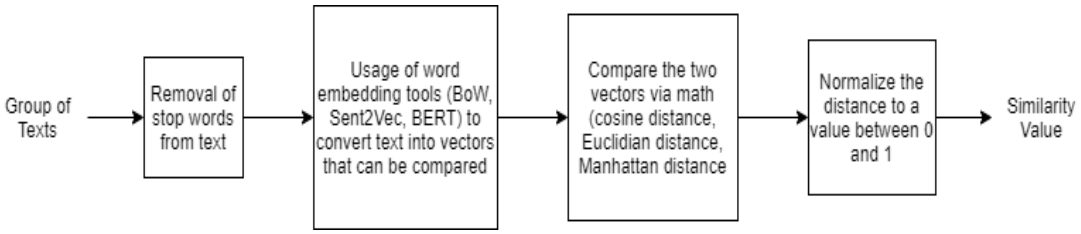


图1. 统计算法的方法。

基于语料库的度量利用大型语料库确定不同词语之间的相似度，从而衡量文本相似度 [8]。点互信息和潜在语义分析是两种运用这一技术的方法。点互信息通过确定被比较的词语共同出现的频率来实现这一点，而潜在语义分析则通过对语料库进行奇异值分解来实现。基于知识的度量采用类似的方法，将词语转换为概念，再利用语义网络比较这些概念 [8]。

其中包含两个神经网络系统，因为它们总体上似乎最有效，分别是知识蒸馏 MT-DNN 和 XLNet [9]。[10] 提出的多任务深度神经网络通过知识蒸馏得到改进。研究人员针对每项任务分别构建了多个“教师”MT-DNN。随后，每个“教师”MT-DNN都会生成一组“软”目标，并将其与相应任务的正确“硬”目标相结合。一个“学生”MT-DNN则使用这些组合目标作为各项任务的目标。该过程最终使“学生”MT-DNN的表现显著优于训练过程中未使用知识蒸馏的其他 MT-DNN。XLNet结合使用自回归语言建模和自编码，构建出一种预训练模型；该模型无需像 BERT [11]那样依赖数据扰乱，就能从前后文中获取上下文信息。因此，在大多数自然语言处理任务中，XLNet 的表现都显著优于基线模型 BERT。此外，文中还采用了一种统计技术，即 Sentence-BERT + 余弦相似度。Sentence-BERT 是 BERT 的一种改进版本，它使用孪生三元组网络生成兼具语义意义且易于通过余弦相似度进行比较的句子嵌入 [7]。除这些算法外，文中还结合使用了 [8] 提出的 Lin 基于知识的度量，以及 [12] 和 [4] 提出的字符串相似度方法。Lin 基于知识的度量专为理解语义相似度而设计，而字符串相似度方法旨在理解词汇相似度。因此，两种算法结合后能够理解的信息远多于单独使用其中任一种算法。

所使用的数据集包括微软研究释义语料库（MRPC） [13], 、论点方面相似度（AFS）数据集 [14], 以及涉及组合知识的句子（SICK）数据集[15]。MRPC数据集旨在测试算法能否识别释义，而AFS数据集旨在测试算法能否识别相似的论点。SICK数据集分为两个任务，分别称为SICK-R和SICK-E。SICK-R用于测试算法对句子词汇相似度的判断能力，而SICK-E则通过判断句子是相似、无关还是含义相反，测试算法是否具备高层语义知识。各项任务采用的评估指标为：AFS和SICK-R使用皮尔逊相关系数和斯皮尔曼等级相关系数，MRPC和SICK-E使用分类准确率。

4. 结果

Sentence-BERT + 余弦相似度的实现总体进展顺利。大多数语料库下载后，实现起来都很简单。最大的问题来自AFS数据集：它以逗号分隔值格式存储，因此难以区分句子中的逗号和用于分隔数据的逗号。解决方法是编辑数据集，将句子中的每个逗号都改为两个逗号，然后使用正则表达式，仅根据CSV格式中的单个逗号拆分句子。拆分句子后，再将双逗号替换为单个逗号就很简单了。评估指标的代码编写起来有些困难，但找到相应公式后，实现起来就相对简单了。遗憾的是，由于SICK-E的分类方式，SemanticBERT + 余弦相似度无法正确对信息进行分类，因此无法得出分数。

遗憾的是，XLNet 的实现并不简单。最初的计划是使用原论文中链接的实现 [16], ，但经过深思熟虑后，改为使用 Hugging Face Transformers [17] ，因为导入和使用起来会容易得多。起初，XLNet 模型存在严重问题，因为输入维度总是出现偏差。通过使用填充解决了这一问题。随后，训练和评估又成为问题，因为运行其中任何一项任务都会因使用过多 GPU内存而持续失败。通过减少批处理步骤解决了这一问题。之后，我能够顺利找到分类数据集的准确率，但遗憾的是，找到能够执行回归的 XLNet 模型仍然是一个重大难题。

从一开始，实现 MT-DNN 就是一个重大难题。花了几天时间才弄清楚，一个不断出错的 TensorFlow 导入要求使用较旧版本的 Python（3.6）；又过了一天才意识到，GitHub 项目 [18] 是为 Linux操作系统设计的。发现这两个问题后，便将实现转移到名为 Paperspace Gradient 的云计算服务器上。完成这些工作后，模型得以配置完成。导入错误有时会造成困难，但通过 apt install 安装相关组件很好地解决了这些问题。此外，为了适应使用 python3 而非 python2 运行 Python 程序，通常还需要重写程序中的代码。为了让 MT-DNN 快速运行，还必须重新安装一个仅部分安装的 CUDA驱动程序；由于其中存在依赖关系，简单删除驱动程序并不起作用，因此这一过程相当困难。最终，手动删除依赖该驱动程序的系统程序，再删除驱动程序，解决了这一问题。起初，上传数据集似乎很简单，但很快就变得困难起来，因为每项任务都有特定的数据排列方式，而上传的数据集并不匹配。重新进行预处理以解决这一问题，并将数据集划分为训练集/测试集/开发集划分后，评估工作所幸变得简单了。

基于知识的度量的实现总体来说很简单，尽管过程中也遇到了一些小波折。最初的计划是使用[8], 中提到的实现，但论文中没有任何地方链接到它，且在网上也找不到。最终，改为将 Sematch [19,20] 纳入，作为基于知识的度量的实现。原本计划将多个基于知识的度量结合起来，但很快发现，在剩余时间内这样做困难至几乎不可能。最终选择的基于知识的度量是林氏方法，因为它似乎是 [8] 中提到的各个算法中最有效的，而且可以轻松归一化到 0 到 1 之间。选定算法和 GitHub 实现后，实现中最困难的问题便显现出来了。基于知识的度量通过将词语转换为概念并比较这些概念来工作，因此它们是为词语相似度而非句子相似度设计的。所有评估指标比较的是句子

相似度，因此有必要将词语相似度转换为句子相似度。幸运的是，找到了几篇正是这样做的论文[12,13]。它们通过构建一个比较矩阵来实现这一点，该矩阵比较两个句子中每个词语之间的相似度。完成后，这些论文从矩阵中取出最高值，将其加入列表，删除该值所在的行和列，并持续进行，直到最高值为零，或不再剩余任何列或行。作者们还将基于知识的度量与一种相似度度量结合起来，以使算法对词汇信息更加敏感。利用这些论文将实现中的词语相似度转换为句子相似度并不困难，之后一切进展顺利。

表3. 各任务/评估组合的结果分析。

	SICK-R 皮尔逊相关系数	SICK-R 斯皮尔曼相关系数	SICK-E 准确率	**AFS** 皮尔逊相关系数	**AFS** 斯皮尔曼相关系数	**MRPC** 准确率
Sentence-BERT + 余弦相似度	73.61%	75.038%	N/A	28.034%	75.063%	68.017%
MT-DNN	92.185%	88.567%	90.706%	80.860%	78.178%	88.286%
XLNet2（第二实现）	47.697%	N/A	57.383%	72.752%	不适用	66.236%
XLNet1（第一实现）	N/A	N/A	56.911%	N/A	N/A	66.983%
Lin度量 + 字符串相似度	60.546%	75.038%	N/A	32.273%	75.063%	70.172%

在 XLNet 的实现过程中，我们发现所使用的 transformers XLNet 代码是为分类问题设计的，因此无法处理连续数据。为此，我们决定采用 XLNet 原论文中链接的另一种实现[16]，以确保能够收集有关 SICK-R 和 AFS 的 XLNet 数据。主要问题在于安装 CUDA 10.0：Paperspace Gradient 会自动安装 CUDA 10.2，因此必须先卸载它，才能安装 CUDA 10.0。此外，Nvidia 驱动程序也带来了一些问题，导致 CUDA 无法完整安装。最终，我们从系统中清除了所有 CUDA 痕迹，部分安装 CUDA 10.0，手动移除了 nVidia 驱动程序及其依赖项，然后运行 apt --fix-broken install，以正确安装 CUDA 10.0。此后，实现过程基本顺利。之后唯一真正的障碍是，XLNet 只使用皮尔逊相似度作为指标，而且原本不包含 MRPC 处理。因此，我们需要基于其基本的 MNLI匹配处理器编写一个 MRPC处理器子类，之后评估过程便顺利完成。遗憾的是，添加斯皮尔曼秩相似度要困难得多，而且已没有时间完成这项工作。所有结果均使用 60% 的训练数据集、20% 的测试数据集和 20% 的开发数据集。图 2 展示了文档相似度算法的准确率。

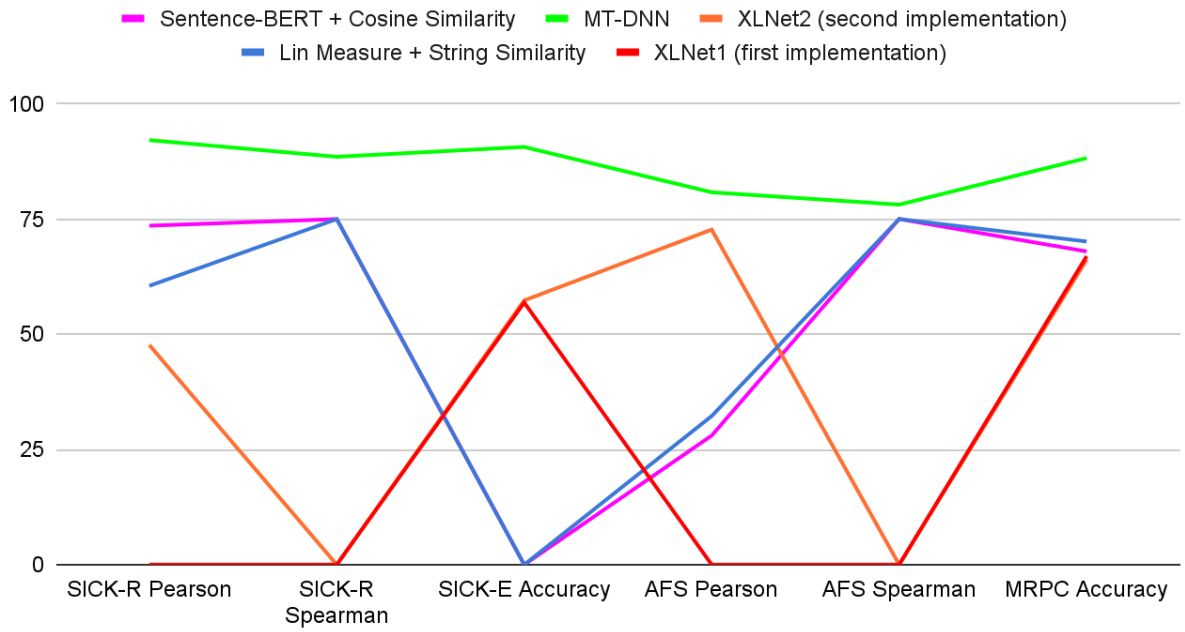


图2. 各任务/评估组合的结果分析。

Sentence-BERT 的余弦相似度+ 和非官方 XLNet 代码均在一台 Alienware m15 R4 上实现，该设备配备英特尔 i7-10870H 处理器、32 GB 内存和 NVIDIA GForce RTX 3070 显卡。MT-DNN、第二种 XLNet 实现以及 Lin度量+ 字符串相似度则在 Paperspace 上实现，所用设备配备英特尔至强 E5-2623 v4 处理器、30 GB 内存和 NVIDIA P5000 显卡。

在测试的算法中，MT-DNN 的效果明显最佳，因为它是唯一能够成功处理所有评估指标的算法，并且在所有评估指标中都取得了最高分。Lin度量 + 字符串相似度和 Semantic BERT + 余弦相似度的结果大致相同；其中，Semantic BERT + 余弦相似度在 SICK-R 皮尔逊任务中的表现更好，但在 AFS 皮尔逊任务和 MRPC 任务中的表现略差。官方 XLNet 代码在 AFS 皮尔逊任务中的表现极佳，但在 SICK-R 皮尔逊任务中的表现很差，其 MRPC 得分也略低于 Lin + 字符串相似度和 BERT + 余弦相似度的得分。然而，与 Lin + 字符串相似度和 BERT + 余弦相似度不同，第二个 XLNet 实现能够为 SICK-E 给出得分，因此在灵活性方面具有明显优势。第一个 XLNet 实现的得分与第二个 XLNet 实现相近，但无法处理连续数据，因此就多功能性和结果而言，它是所有受测算法中表现最差的。

未来可以进一步研究 XLNet。XLNet 的第二种实现理论上能够使用斯皮尔曼等级相关相似度作为指标，因此它是除 MT-DNN 之外唯一能够处理所有评估指标的算法。

5. 结论

本研究全面考察了四种算法：Semantic BERT + 余弦相似度、MT-DNN、XLNet 和 Lin度量 + 字符串相似度。通过使用 MRPC、AFS、SICK-E 和 SICK-R 数据集，以及斯皮尔曼等级相关、皮尔逊相关系数和准确率等评估指标，本研究分析了这四种算法各自的优势与不足，从而回答了哪种文档相似度算法最为有效这一问题。

根据本研究获得的数据，MT-DNN显然是最有效的文档
相似度算法。

缩略词

MRPC：微软研究释义语料库。AFS：论点方面相似度。
SICK：涉及组合知识的句子。BERT：来自 transformers 的
双向编码器表示。MT-DNN：多任务深度神经网络。

参考文献

[1] J. M. Sanchez-Gomez, M. A. Vega-Rodríguez, C. J. Pérez. “词项加权方案和相似度度量对抽取式多文档文本摘要的影响。”《专家系统与应用》。
2021 169 10 1016 2020 114510 ; :N.PAG. doi: . /j.eswa. . K. Seki. “利
[2]
用神经机器翻译模型进行跨语言文本相似度计算。”《信息科学杂志》，第47卷，第3期，第404–418页，2021年6月。[3] E. Inan. “SimiT：一种使用词典和依存表示的文本相似度方法。”《新一代计算》，第38卷，第3期，第509–530页，2020年7月。[4] A. Islam 和 D. Inkpen，“使用基于语料库的词语相似度和字符串相似度计算语义文本相似度”，《ACM数据知识发现汇刊》，2008年。 [在线]。可访问：https://www.academia.edu/4418471/Semantic_text_similarity_using_corpus_based_word_similarity_and_string_similarity。[访问日期：28-Jul-2021]。[5] Y. Sever 和 G. Ercan，“词典对齐问题上的跨语言文本相似度评估”，SpringerLink，2020年。[在线]。可访问：https://link-springer-com.ezaccess.libraries.psu.edu/article/10.1007/s10579-020-09498-1。[访问日期：29-Jul-2021]。[6] Y. Arase 和 J. Tsujii，“使用短语释义对 BERT 进行迁移微调”，ScienceDirect，2021年3月。[在线]。h
tt
ps://www.sciencedirect.com.ezaccess.libraries.psu.edu/science/article/pii/S0885230820300978。[访问日期：29-Jul-2021]。[7] N. Reimers 和 I. Gurevych，“Sentence-BERT：使用孪生 BERT 网络进行句子嵌入”，arXiv.org，2019年8月27日。 [在线]。可访问：https://arxiv.org/abs/1908.10084。[访问日期：03-Jun-2021]。[8] R. Mihalcea、C. Corley 和 C. Strapparava，“文本语义相似度的基于语料库和基于知识的度量。”[在线]。可访问：https://web.eecs.umich.edu/~mihalcea/papers/mihalcea.aaai06.pdf。[访问日期：02-Jun-2021]。[9] “语义文本相似度”，NLPProgress。[在线]。可访问：http://nlpprogress.com/english/semantic_textual_similarity.html。[访问日期：02-Jun-2021]。[10] X. Liu、P. He、W. Chen 和 J. Gao，“通过知识蒸馏改进用于自然语言理解的多任务深度神经网络”，arXiv.org，2019年4月20日。 [在线]。可访问：https://arxiv.org/abs/1904.09482。[访问日期：eh, we ca03-Jun-2021]。[11] Z. Yang、Z. Dai、Y. Yang、J. Carbonell、R. Salakhutdinov 和 Q. V. Le，“XLNet：用于语言理解的广义自回归预训练”，arXiv.org，2020年1月2日。 [在线]。可访问：https://arxiv.org/abs/1906.08237。[访问日期：03-Jun-2021]。[12] C. F. Ho、M. A. A. Murad、R. A. Kadir 和 S. C. Doraisamy，“基于词义消歧的句子相似度”，ACL Anthology，2010年。 [在线]。可访问：https://aclanthology.org/C10- 2048/。[访问日期：07-Jul-2021]。[13] W. B. Dolan 和 C. Brockett，“自动构建句子释义语料库”，ACL Anthology，1970年1月1日。[在线]。可访问：https://aclanthology.org/I05-5002/。[访问日期：22-Jul- 2021]。[14] Amita Misra、Brian Ecker、Marilyn Walker。“衡量对话中句子论点的相似度”。发表于第17届年度 SIGdial 话语与对话会议（SIGDIAL），美国加利福尼亚州洛杉矶，2016年。[15] “SICK数据集”，SICK。 [在线]。可访问：http://marcobaroni.org/composes/sick.html。[访问日期：03-Jun-2021]。

[16] Zihangdai，“zihangdai/xlnet”，GitHub，2019年12月16日。[在线]。可访问：https://github.com/zihangdai/xlnet/。[访问日期：03-Jun-2021]。[17] T. Wolf，“huggingface/transformers”，GitHub，2021年6月16日。[在线]。可访问：https://github.com/huggingface/transformers。[访问日期：16-Jun-2021]。[18] Namisan，“namisan/mt-dnn”，GitHub，2021年2月16日。[在线]。可访问：https://github.com/namisan/mt-dnn/。[访问日期：03-Jun-2021]。[19] G. Zhu 和 C. A. Iglesias，“Sematch：知识图谱语义相似度框架”，ScienceDirect，2017年8月15日。[在线]。可访问：https://www-sciencedirect-com.ezaccess.libraries.psu.edu/science/article/pii/S0950705117302447?via%3Dihub。[访问日期：06- Jul-2021]。[20] G. Zhu 和 C. A. Iglesias，“gsi-upm/sematch”，GitHub，2019年3月26日。[在线]。可访问：https://github.com/gsi-upm/sematch。[访问日期：06-Jul-2021]。[21] AndriyMulyar，“AndriyMulyar/semantic-text-similarity”，GitHub，2019年10月8日。[在线]。可访问：https://github.com/AndriyMulyar/semantic-text-similarity。[访问日期：03-Jun-2021]。