

# 面向自然语言处理的词向量模型研究综述

安俊秀, 蒋思畅

(成都信息工程大学 软件工程学院, 四川 成都 610225)

**摘要:**从20世纪50年代至今,自然语言处理(Natural Language Processing, NLP)取得了长足的发展。早期的词向量模型证明了对该领域的研究需要使用数学方法,而不是人类的语言规则。进入21世纪后,静态模型以深度学习技术为基础,在很多任务中取得了不错的表现;动态模型再将预训练技术融入进来,实现了根据语境对词向量进行调整的功能,为NLP领域带来了里程碑式的突破,后续研究在此基础上向各领域延伸扩展,并且在现实生活中得到了大规模的应用。文章首先对词向量模型及其发展历史做了介绍,然后分析了现代的词向量模型(NNLM, Word2Vec, FastText, Glove, ELMo, GPT, BERT),其次说明了多种基于预训练技术的扩展模型和当前自然语言处理技术的应用现状,最后总结了目前存在的主要问题,并提出对未来研究的展望。

**关键词:**自然语言处理;词向量;深度学习;预训练技术;静态模型;动态模型

中图分类号:TP391.1

文献标识码:A

文章编号:1673-629X(2023)12-0017-06

doi:10.3969/j.issn.1673-629X.2023.12.003

## Survey of Word Vector Model for Natural Language Processing

AN Jun-xiu, JIANG Si-chang

(School of Software Engineering, Chengdu University of Information Technology, Chengdu 610225, China)

**Abstract:** Since the 1950s, Natural Language Processing (NLP) has made great progress. The early word vector model demonstrates that the study of NLP requires mathematical methods rather than human language rules. After entering the 21st century, the static model which based on deep learning techniques achieves good performance in many tasks. The dynamic model makes use of pre-training techniques and realizes the function of adjusting word vectors according to the context, which brings a milestone breakthrough in the field of NLP. On this basis, the follow-up research extends to various fields, and has been applied on a large scale in real life. We firstly introduce the word vector model and its development history, then analyze the modern models based on deep learning (NNLM, Word2Vec, FastText, Glove, ELMo, GPT, BERT). Secondly, we explain a variety of extended models based on pre-training technology, and describe the current application status of natural language processing technology. Finally, we summarize the main problems at present, and put forward the prospect of future research.

**Key words:** natural language processing; word vector; deep learning; pre-training technique; static model; dynamic model

### 0 引言

自然语言处理主要研究实现人与计算机之间用自然语言进行有效通信的理论方法及应用,它的一般流程如图1所示。

语言的数据称为语料库(Corpus),特征工程指的就是将自然语言转换成词的特征向量(简称词向量)的处理过程,因为词是自然语言中表示语义的最小单位。词向量的好坏直接决定了任务模型的性能高低,因此NLP的特征工程是关键的一步,为此构建、训练针对词向量的模型是最优解法。

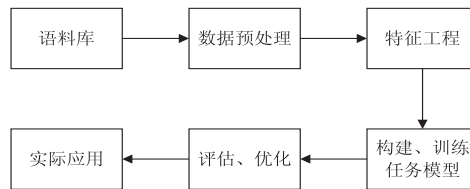


图1 NLP的一般流程

词向量的发展过程大致分为“规则-统计-深度学习”三个阶段<sup>[1]</sup>。20世纪50至70年代,学者们用电脑模拟人脑,结果是一无所获。70年代后,研究者们重新审视这个问题,提出了基于统计和数学模型的方

收稿日期:2023-03-06

修回日期:2023-07-07

基金项目:国家社会科学基金项目(22BXW048)

作者简介:安俊秀(1970-),女,教授,硕士,CCF高级会员(17133S),研究方向为大数据、云计算;通信作者:蒋思畅(1998-),男,硕士研究生,研究方向为自然语言处理。

法,加之机器学习的快速发展,NLP 领域终于有了实质性的突破。进入 21 世纪后,Bengio 等人<sup>[2]</sup>在 2003 年提出的 NNLM 模型首次采用了神经网络来训练词向量,为研究者们打开了新的思路。Hinton 等人<sup>[3]</sup>在 2006 年首次提出了深度学习的概念,将人工智能领域的研究推向了高潮,随后一系列经典的词向量模型被陆续提出,尤其是预训练技术成为主流的 2018 年,其训练出的词向量在大多数 NLP 的下游任务中都表现出了优异的性能,为后续发展带来了更多可能性。

预训练技术来源于迁移学习<sup>[4]</sup>,最初应用在计算机视觉(Computer Vision, CV)领域,大获成功后被应用到 NLP 领域。其本质是利用海量数据通过无监督学习,更准确地说是自监督学习的方式去训练一个模型,得到数据的通用特征,然后通过微调(Fine Tune)词向量模型的参数以便应用于不同的任务模型。因此,词向量模型、任务模型越来越多地被称为预训练模型、下游任务。

## 1 现代词向量模型

### 1.1 NNLM

针对语义问题,Harris<sup>[5]</sup>提出了分布假说:“上下文相似的词,其语义也是相似的”,Firth<sup>[6]</sup>提出了语言学概论,完善了这一假说,指出“词的语义由其上下文决定”。针对编码稀疏、维数灾难<sup>[7]</sup>问题,Hinton<sup>[8]</sup>提出了分布式表示方法,指将高维稀疏向量嵌入(也称映射)到低维稠密向量中,该方法将信息分布在向量的各个维度上(即值不为 0),此时词向量也称为词嵌入(Word Embedding),具有更强的表示能力。

为了解决语义和维度这两个问题,Bengio 等人借鉴了基于统计的  $N$  元模型<sup>[9]</sup>( $N$ -Gram Model)的思想,提出了神经网络(概率)语言模型(Neural Network (Probabilistic) Language Model, NNLM)。

语言模型指在给定若干个词的情况下预测下一个词。假设  $S$  是某个句子,由词  $w_1, w_2, \dots, w_n$  组成,其中  $n$  表示句子长度。 $S$  出现的概率等于每个词出现的条件概率的乘积,计算式表示为:

$$P(S) = P(w_1, w_2, \dots, w_n) = P(w_1) \cdots P(w_n | w_1, w_2, \dots, w_{n-1}) \quad (1)$$

式(1)的缺点很明显,位置越靠后的词的概率越难计算,涉及到的变量太多。 $N$  元模型为了简化问题,利用了  $N-1$  阶马尔可夫假设,即某个词  $w_i$  出现的概率仅与它前面的  $N-1$  个词有关,计算式表示为:

$$P(S) = P(w_1, w_2, \dots, w_n) = \prod_{i=1}^n P(w_i | w_{i-N+1}, w_{i-N+2}, \dots, w_{i-1}) \quad (2)$$

$N$  的取值一般为 1, 2, 3, 对应的语言模型分别称为

一元模型(Unigram)、二元模型(Bigram)、三元模型(Trigram),  $N$  取更大的值时,性能提升并不显著,但对资源的耗费愈发严重,因此  $N > 3$  的情况极少。因此上下文的跨度不能太大,否则会出现长距离依赖<sup>[10]</sup>(Long Distance Dependency)问题。

NNLM 将深度学习的思想融入语言模型中,其结构依次为输入层(Input Layer)、嵌入层(Embedding Layer)、隐含层(Hidden Layer)、输出层(Output Layer)。模型训练好后,词向量就是输入与嵌入矩阵的乘积。这直接启发了后来的研究。与  $N$  元模型不同的是,NNLM 中  $N$  可以取更大的值,减小了长程依赖性的影响。

### 1.2 Word2Vec

NNLM 所产生的词向量,其语义只是基于上文,没有包含下文,同时定长向量使得计算复杂度并不低,Google 团队对此提出了 Word2Vec<sup>[11-12]</sup>,包含两种模型,结构依次为输入层、隐含层、输出层。连续词袋模型(Continuous Bag-of-Words, CBOW)的思想是通过上下文词去预测中心词,跳元模型(Skip-Gram)的思想与之相反,通过中心词去预测上下文词。上下文词的数量需要人为设置。

为了减小损失函数的计算复杂度  $O(|V|)$  ( $|V|$  指词表中词的数量),Google 提出了两种训练方式:负采样(Negative Sampling)和层次(Hierarchical) Softmax。负采样指的是将预测的词  $w_0$  记为正例,然后根据词频和概率从词表中抽取  $K$  个词  $w_i (i = 1, 2, \dots, K)$  作为负例,训练目标就成了最大化似然函数,如式(3)所示。

$$\prod_{i=1}^K P(\text{context}(w_0), w_i) \rightarrow \prod_{i=1}^K \log P(\text{context}(w_0), w_i) \quad (3)$$

其中,  $\text{context}(w_0)$  表示词  $w_0$  的上下文词,  $P()$  是 Sigmoid 函数。这使得计算复杂度  $O(|V|)$  降到了  $O(K)$ 。

层次 Softmax 指根据词频对词表构建哈夫曼树(Huffman Tree),每个叶节点表示一个单词,且从根节点到叶节点只有一条路径,因此不再需要像普通的 Softmax 去遍历词表,而是计算从根到对应叶的路径的概率,计算复杂度便从  $O(|V|)$  降到了  $O(\log |V|)$ 。

需要注意的是,在应用时,CBOW 模型使用上下文词向量作为词表示,而 Skip-Gram 模型使用中心词向量作为词表示。

### 1.3 FastText

Facebook 团队在 Word2Vec 的基础上提出了基于

CBOW 的 FastText 模型<sup>[13]</sup> 和基于 Skip-Gram 的 FastText 模型<sup>[14]</sup> (分别简称为 CBOW-FT, SG-FT)。CBOW-FT 的网络结构与 CBOW 的网络结构是相似的,但训练任务是对文档进行分类,属于监督学习。在模型中,文档的所有词都会作为输入,并且增加了词级的 N-Gram 输入(以“I am fine”的 Bigram 为例,会增加“I am”、“am fine”两个输入)。输入层会对两者都进行嵌入处理,考虑到 N-Gram 的数据量庞大以及存在重复的情况,作者做了哈希处理。

SG-FT 的网络结构和训练任务与 Skip-Gram 是一样的。输入层除了独热编码的词向量以外,还增加了字符级的 N-Gram 嵌入(也称子词嵌入(Subword Embedding))输入(以“word”的 Trigram 为例,会增加“<wo”,“wor”,“ord”,“rd>”四个输入,其中“<”,“>”表示边界)。

作者为了减小损失函数的计算复杂度,同样采用了层次 Softmax 的训练方式。

FastText 模型遇到未知的词时能够直接使用 N-gram 嵌入向量的和去表示该词,提高了容错能力,这是 Word2Vec 所不具备的。另外一点,FastText,正如其名,速度非常快,通常能够将耗时单位为小时甚至是天的训练时间大幅缩短到分钟之内。

#### 1.4 GloVe

对于语义问题,Word2Vec 与 FastText 仅利用了窗口范围内的局部上下文的信息,Pennington 在此基础上将语料库中关于词的全局共现统计信息考虑进来,提出了 GloVe<sup>[15]</sup> (Global Vectors for Word Representation)。需要注意的是,该模型不属于神经网络语言模型,而是基于矩阵分解的方式。

作者用词的共现矩阵来表示全局信息,假设矩阵为  $X$ ,那么  $X_{ij}$  表示词  $w_j$  出现在词  $w_i$  的上下文的次数,  $X_i = \sum_k X_{ik}$  表示所有词出现在词  $w_i$  的上下文的次数。对于词  $w_j$  出现在词  $w_i$  的上下文的概率,用式子表示为:

$$P(w_j | w_i) = X_{ij} / X_i \quad (4)$$

计算该矩阵需要遍历一次语料库,虽然计算量较大,但这仅是一次性的前期投入成本。

在 Skip-Gram 模型中,词  $w_j$  出现在词  $w_i$  的上下文的概率是以 Softmax 的方式计算的,如式(5)所示。

$$Q(w_j | w_i) = e^{w_j^T w_i} / \sum_{w=1}^W e^{w^T w_i} \quad (5)$$

其中,  $W$  表示词的数量。

结合式(4)与式(5),作者提出了最小二乘法的损失函数,用式子表示为:

$$\text{Loss} = \sum_{i=1}^W \sum_{j=1}^W X_{ij} (Q_{ij} - P_{ij})^2 \quad (6)$$

为了降低计算复杂度,作者用了 3 种方法:丢弃计算概率用的归一化因子;对概率取对数;引入权重函数,如式(7)所示。

$$f(x) = \begin{cases} (x/x_{\max})^\alpha & 0 \leq x < x_{\max} \\ 1 & x \geq x_{\max} \end{cases} \quad (7)$$

其中,  $\alpha, x_{\max}$  为超参数。

最终的损失函数如式(8)所示。

$$\text{Loss} = \sum_{i=1}^W \sum_{j=1}^W f(X_{ij}) (w_j^T w_i + b_i + b_j - \log X_{ij})^2 \quad (8)$$

其中,  $b_i, b_j$  是偏置项。

全局对数双线性回归模型的名称便是由此而来。GloVe 模型结合了矩阵分解和 Word2Vec 的优势,能够充分利用全局统计信息和局部上下文信息,训练速度更快,且词向量拥有了更丰富的语义信息。

#### 1.5 ELMo

Word2Vec, FastText, GloVe 模型在语义上还存在一个缺陷:一个词只有一种词向量,即仅能表示词的一种含义。由此这些模型又被称为静态词向量模型。为了使多义词在不同的语境下能够拥有对应的词向量,学者们将研究重点放在了动态词向量模型上, Peters 等人提出的 ELMo<sup>[16]</sup> (Embeddings from Language Models)便是一个典范,其网络结构依次为输入层、隐含层、输出层。

输入层是预训练的静态词向量,一般使用上文介绍的模型,但作者用的是 Char-CNN 与 LSTM 相结合的模型<sup>[17]</sup>。隐含层的双向 LSTM 用来从前后两个方向去“阅读”文本(即综合上下文的信息),其层数为  $2L$  ( $L$  是超参数)。正向 LSTM 的任务是给定前文,预测下一个词,反向 LSTM 的任务是给定后文,预测上一个词,两者目标都是最大化预测概率,如式(9)所示。

$$\prod_{i=1}^N [P(w_i | w_1, w_2, \dots, w_{i-1}) + P(w_i | w_{i+1}, w_{i+2}, \dots, w_N)] \quad (9)$$

为了降低计算复杂度,作者对概率取了对数。ELMo 的思想体现在输出层,该层会对双向 LSTM 所训练出的隐含输出(hidden output,简称为  $h$ )和输入层进行组合得到词向量,如式(10)所示。

$$T_i = \{x, \overrightarrow{h_{i,j}}, \overleftarrow{h_{i,j}} | j = 1, 2, \dots, L\} = \{h_{i,j} | j = 0, 1, \dots, L\} \quad (10)$$

其中,  $h_{i,0}$  是输入层的词向量,  $h_{i,j}$  ( $j \neq 0$ ) 是双向 LSTM 的  $h$  拼接起来的向量。对于不同的下游任务,作者认为可以再有针对性地微调词向量  $T_i$ , 如式(11)所示。



$$\text{ELMo}_i^{\text{task}} = E(T_i; \Theta^{\text{task}}) = \gamma^{\text{task}} \sum_{j=0}^L s_j^{\text{task}} h_{i,j} \quad (11)$$

其中,  $\Theta^{\text{task}}$  表示模型参数,  $s^{\text{task}}$  是归一化的 Softmax 权重,  $\gamma^{\text{task}}$  是缩放系数。

可以看出, ELMo 的本质其实就是根据完整的文本信息对静态词向量做调整, 使词向量能够准确反映出当前语境下的含义, 这便有效地解决了一词多义的问题。在自然语言处理领域的 6 个典型下游任务的数据集上全面刷新了最优成绩。

## 1.6 预训练语言模型

ELMo 使用大规模无监督语料进行预训练和在下游任务中微调的做法为词向量模型的研究提供了全新思路, 成功证明了预训练和微调的可行性。基于此思想, NLP 领域的学者大展身手, 提出了更加优秀的动态词向量模型, 尤其是 OpenAI 团队的 GPT<sup>[18]</sup> (Generative Pre-Training) 与 Google 团队的 BERT (Bidirectional Encoder Representations from Transformers<sup>[19]</sup>)。两模型都是基于 Transformer<sup>[20]</sup> 改进得来的。

### 1.6.1 GPT

GPT 是基于 Transformer 的解码器改进而来的, 属于自回归 (Auto-Regressive) 语言模型, 也称单向语言模型, 即只能利用上文的信息去生成下文。这类模型适合自然语言生成类 (Natural Language Generation, NLG) 的下游任务, 如问答系统。

在预训练阶段, GPT 的目标是最大化预测概率, 如式 (12) 所示。

$$\sum_i \log P(w_i | w_{i-k}, \dots, w_{i-1}) \quad (12)$$

其中,  $N$  表示某文档中词的数量,  $k$  表示窗口大小。

在微调阶段, GPT 的目标仍是最大化预测概率, 不同的是此时的数据是有标签的, 如式 (13) 所示。

$$\sum_{(x,y)} \log P(y | x_1, x_2, \dots, x_m) \quad (13)$$

其中,  $x$  表示文档中的词, 共有  $m$  个,  $y$  表示对应的标签。

为了更好地适应迁移学习, GPT 对输入和输出做了精简和通用的设计。对于输入, 会添加首尾标记和分隔标记; 对于输出, 只需要接入全连接层或其它简单结构。

得益于 Transformer 强大的表征能力, GPT 在公布的结果中, 一举刷新了自然语言处理领域的 9 项典型任务。但 GPT 的不足在于单向, 利用不了下文的信息, 无法发挥出 Transformer 的最佳效果。但让人惊奇的是, GPT 可以实现一些零样本学习 (Zero-Shot Learning<sup>[21]</sup>) 的任务。随后为了进一步研究零样本学习, 并加强 GPT 的泛化能力, OpenAI 团队提出了 GPT-

2<sup>[22]</sup>。进而为了继续增强泛化能力, 聚焦于更通用的模型, OpenAI 团队提出了规模更加庞大的 GPT-3<sup>[23]</sup>, 该模型尝试用更少的领域数据, 且不再进行微调, 训练好后直接应用于下游任务。

GPT, GPT-2, GPT-3 的模型结构都是基于 Transformer 改进而来, 创新性并不高, 但证明了规模越来越大的语言模型是可行的, 只要算力足够强, 性能还会不断提升。

### 1.6.2 BERT

BERT 是基于 Transformer 的编码器改进而来的, 属于自编码 (Auto-Encoder) 语言模型, 也称双向语言模型, 即能够同时利用上下文的信息。这类模型适合自然语言理解类 (Natural Language Understanding, NLU) 的任务, 比如文本分类。

BERT 在模型结构上并无大的改进, 其创新性体现在预训练上, 不再是之前的模型那样根据上文或下文去预测下一个词, 而是掩码语言模型 (Masked Language Model, MLM) 和预测下一句 (Next Sentence Prediction, NSP) 两个任务。MLM 如同英语考试中的完型填空, 模型会预先对 15% 的词做掩码处理, 训练目标便是预测这些词。NSP 的提出是为了更高层次地去理解文本, 训练目标是判断两个句子的关系, 连贯的或是随机的。

BERT 将无监督的预训练和有监督的微调应用到深层的双向结构中, 弥补了 GPT 单向的不足, 在自然语言处理领域的 11 项基本任务上都获得了显著的效果提升。BERT 可谓是集前期研究大成之作, 极大地推动了自然语言处理的发展, 成为一个里程碑事件, 自此预训练和微调的方法成为了主流。

## 2 扩展模型

大规模的预训练语言模型都存在参数量、数据量过大的问题, 从而导致极大的资源消耗。对此, 目前主要的轻量化方法有两种: 剪枝 (Pruning<sup>[24]</sup>) 和知识蒸馏 (Knowledge Distillation<sup>[25]</sup>)。

剪枝技术是指通过减少神经网络中的冗余部分, 从而达到压缩模型的目的。基于 BERT, McCarley 等人<sup>[26]</sup> 为提升问答系统的速度, 在保证精确度损失最小的情况下裁剪了隐含层和全连接层的参数, 另外又研究了针对 Transformer 的冗余结构的剪枝方法; Michel 等人<sup>[27]</sup> 受到门控机制的启发, 通过裁剪注意力头加快了 BERT 约 17.5% 的速度, 且性能在机器翻译和自然语言推理任务上几乎没影响; Gordon 等人<sup>[28]</sup> 不同程度地去除掉接近零的参数, 发现 30% ~ 40% 的裁剪程度不会影响精确度; Lan 等人<sup>[29]</sup> 提出了 ALBERT, 对每个注意力层使用同样的权重矩阵, 将参数减少至

BERTlarge 的 1/18,速度加快了 1.7 倍。

知识蒸馏是指将强力的大模型所包含的知识,蒸馏到轻量的小模型中,使得后者能拥有前者的能力,就像教师对学生传道授业解惑,因此大小模型又分别称为教师 (Teacher) 模型、学生 (Student) 模型。基于 BERT, Sanh 等人<sup>[30]</sup>提出了更小、更快、更廉价、更轻量的 DistilBERT,它的规模只是 BERT 的 40% 左右,但语言理解能力依然强悍,并且速度加快了约 60%; Jiao 等人<sup>[31]</sup>在预训练和微调两个阶段都进行蒸馏,实现了 TinyBERT,它不仅掌握到了通用知识,还学到了特定任务的知识; Sun 等人<sup>[32]</sup>只在输出层进行蒸馏,并提取出教师模型的中间层的知识输入给学生模型,最终只损失了些许性能。

为了使模型学习出更准确的语义表示,除了所给文本的上下文语境之外,加入外部知识是必要的。对此,百度团队提出了 ERNIE<sup>[33]</sup>、清华大学团队提出了 ERNIE (THU)<sup>[34]</sup>,两模型的结构与 BERT 基本一致,区别在于 BERT 是对字进行掩码,而 ERNIE 是对词进行掩码,并且在预训练阶段将知识图谱的信息实体整合到模型中,要求模型同时汇聚上下文和先验知识的信息; Liu 等人<sup>[35]</sup>提出了 K-BERT 模型,将各个领域的下游任务的知识整合起来进行预训练; Peters 等人<sup>[36]</sup>提出了 KnowBERT 模型,一种将先验知识嵌入到模型中的通用方法。

### 3 应用现状

对 GPT-3 进行改进和优化得到 GPT-3.5 后,进一步地,受 InstructGPT<sup>[37]</sup>的启发, OpenAI 团队通过专业的人工标注训练数据和来自人类反馈的强化学习方法 (Reinforcement Learning from Human Feedback, RLHF) 调整模型参数,让 GPT-3.5 具备了更加逼真、自然的语言能力。基于 GPT-3.5 开发的 ChatGPT,代表着自然语言生成领域的巨大突破,使得计算机能够与人类进行自然语言交互,任何领域的问题都可以解答,类似于百科全书,还能实时学习新的知识。一经推出便引起了全社会的极大关注,具有革命性的意义,让人们不得不重新思考人工智能的发展及其影响。

在 ChatGPT 之后,学术界、工业界迅速意识到了通用大模型的主流趋势,纷纷加入到了对该方向的研究。大模型的有复旦大学的 MOSS、清华大学的 ChatGLM、百度的文心一言、阿里的通义千问、亚马逊 (Amazon) 的 Titan 等,小工具的有 ChatPDF, ChatDOC, ChatPPT, ChatExcel, ChatPaper, 各种 AI 问答机器人等。

遵循 GPT, GPT-2, GPT-3 和 GPT-3.5 的研究路径, OpenAI 团队提出了更加复杂和强大的 GPT-4<sup>[38]</sup>。

与 ChatGPT 相比, GPT-4 具有以下优势: 接受图像作为输入,具有多模态能力; 能够处理超过 25 000 个词的超长文本; 实际应用中更具创造性和技术性; 引入了更多的人工反馈并实时监控, 保证安全性。

### 4 结束语

从 20 世纪 50 年代, 自然语言处理取得了长足的发展, 该文概述了主要成果, 并介绍了近些年的扩展、改进和应用现状。当前存在的挑战在于语料、规模和安全三方面: 获取大规模且高质量的语料非常困难, 数据标注是必需的; 规模不断增大, 模型更加复杂和黑盒化, 训练和部署需要耗费大量的能源和资源, 其预测结果和内部决策过程也难以解释; 科研人员需要及时维护以严防恶意攻击和恶意内容, 同时需要遵循相关法律和标准, 避免产生危害。

未来的研究既会致力于解决语料、规模、安全的问题, 也会加快实践多模态的思想, 并结合其它领域共同发展, 单模态、单领域信息的局限性无法避免, 将图像、文本、语音、视频等信息综合起来学习, 势必会取得更大进展。

### 参考文献:

- [1] 吴 军. 数学之美[M]. 北京: 人民邮电出版社, 2014.
- [2] BENGIO Y, DUCHARME R, VINCENT P, et al. A neural probabilistic language model[J]. Journal of Machine Learning Research, 2003, 3: 1137-1155.
- [3] HINTON G E, OSINDERO S, TEH Y W. A fast learning algorithm for deep belief nets[J]. Neural Computation, 2006, 18(7): 1527-1554.
- [4] YOSINSKI J, CLUNE J, BENGIO Y, et al. How transferable are features in deep neural networks? [J]. Advances in Neural Information Processing Systems, 2014, 27: 3320-3328.
- [5] HARRIS Z S. Distributional structure[J]. Word, 1954, 10(2-3): 146-162.
- [6] FIRTH J. A synopsis of linguistic theory, 1930-1955 [J]. Studies in Linguistic Analysis, 1957, 1: 1-32.
- [7] WRIGHT E M. Adaptive control processes; a guided tour. By Richard Bellman. 1961. 42s. Pp. xvi+ 255. (Princeton University Press) [J]. The Mathematical Gazette, 1962, 46(356): 160-161.
- [8] HINTON G E. Learning distributed representations of concepts [C]//Cognitive science society. Amherst: Lawrence Erlbaum Associates Inc, 1986: 12.
- [9] BROWN P F, DELLA PIETRA V J, DESOUZA P V, et al. Class-based n-gram models of natural language[J]. Computational Linguistics, 1992, 18(4): 467-480.
- [10] HOCKETT C F. A formal statement of morphemic analysis [J]. Studies in Linguistics, 1952, 10: 27-39.
- [11] MIKOLOV T, CHEN K, CORRADO G, et al. Efficient esti-

- mation of word representations in vector space[C]//International conference on learning representations. Scottsdale; [s. n.], 2013:1–12.
- [12] MIKOLOV T, SUTSKEVER I, CHEN K, et al. Distributed representations of words and phrases and their compositionality[J]. *Advances in Neural Information Processing Systems*, 2013, 26:3111–3119.
- [13] JOULIN A, GRAVE E, BOJANOWSKIO P, et al. Bag of tricks for efficient text classification[C]//European chapter of the association for computational linguistics. Berlin: ACL, 2016:427–431.
- [14] BOJANOWSKI P, GRAVE E, JOULIN A, et al. Enriching word vectors with subword information[J]. *Transactions of the Association for Computational Linguistics*, 2017, 5:135–146.
- [15] PENNINGTON J, SOCHER R, MANNING A C D. Glove: global vectors for word representation[C]//Empirical methods in natural language processing. Doha: ACL, 2014:1532–1543.
- [16] PETERS M, NEUMANN M, IYYER M, et al. Deep contextualized word representations[C]//North American chapter of the association for computational linguistics. New Orleans: ACL, 2018:2227–2237.
- [17] JOZEFOWICZ R, VINYALS O, SCHUSTER M, et al. Exploring the limits of language modeling[C]//Empirical methods in natural language processing. Austin: ACL, 2016:1302–1312.
- [18] RADFORD A, NARASIMHAN K, SALIMANS T, et al. Improving language understanding by generative pre-training[R]. San Francisco: OpenAI, 2018.
- [19] DEVLIN J, CHANG M W, LEE K, et al. BERT: pre-training of deep bidirectional transformers for language understanding[C]//North American chapter of the association for computational linguistics. Minneapolis: ACL, 2019:4171–4186.
- [20] VASWANI A, SHAZEER N, PARMAR N, et al. Attention is all you need[C]//Neural information processing systems. Long Beach: Curran Associates Inc, 2017:5998–6008.
- [21] PALATUCCI M, POMERLEAU D, HINTON G E, et al. Zero-shot learning with semantic output codes[C]//Neural information processing systems. Vancouver: Curran Associates Inc, 2009:1410–1418.
- [22] RADFORD A, WU J, CHILD R, et al. Language models are unsupervised multitask learners[J]. *OpenAI Blog*, 2019, 1(8):9.
- [23] BROWN T, MANN B, RYDER N, et al. Language models are few-shot learners[J]. *Advances in Neural Information Processing Systems*, 2020, 33:1877–1901.
- [24] HE Y, ZHANG X, SUN J. Channel pruning for accelerating very deep neural networks[C]//IEEE international conference on computer vision. Venice: IEEE, 2017:1389–1397.
- [25] HINTON G, VINYALS O, DEAN J. Distilling the knowledge in a neural network[J]. *arXiv*:1503.02531, 2015.
- [26] MCCARLEY J S, CHAKRAVARTI R, SIL A. Structured pruning of a bert-based question answering model[J]. *IEEE Access*, 2021, 9:49610–49620.
- [27] MICHEL P, LEVY O, NEUBIG G. Are sixteen heads really better than one? [J]. *Neural Information Processing Systems*, 2019, 32:16075–16086.
- [28] GORDON M A, DUH K, ANDREWS N. Compressing bert: studying the effects of weight pruning on transfer learning[J]. *IEEE Access*, 2020, 8:91916–91928.
- [29] LAN Z, CHEN M, GOODMAN S, et al. Albert: a lite bert for self-supervised learning of language representations[J]. *Advances in Neural Information Processing Systems*, 2019, 32:14096–14105.
- [30] SANH V, DEBUT L, CHAUMOND J, et al. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter[C]//International conference on learning representations. [s. l.]: [s. n.], 2020:1–9.
- [31] JIAO X, YIN Y, SHANG L, et al. Tinybert: distilling bert for natural language understanding[C]//International conference on information and knowledge management. Beijing: ACM, 2019:399–408.
- [32] SUN S, CHENG Y, GAN Z, et al. Patient knowledge distillation for bert model compression[C]//Empirical methods in natural language processing. [s. l.]: ACL, 2020:2219–2230.
- [33] SUN Y, WANG S, LI Y, et al. Ernie: enhanced representation through knowledge integration[C]//Empirical methods in natural language processing and international joint conference on natural language processing. Hong Kong: ACL, 2019:1–10.
- [34] ZHANG Z, HAN X, LIU Z, et al. ERNIE: enhanced language representation with informative entities[C]//Association for computational linguistics. Florence: ACL, 2019:1441–1451.
- [35] LIU W, ZHOU P, ZHAO Z, et al. K-bert: enabling language representation with knowledge graph[C]//Artificial intelligence. New York: AAAI Press, 2020:2901–2908.
- [36] PETERS M E, NEUMANN M, LOGAN I V R L, et al. Knowledge enhanced contextual word representations[C]//Empirical methods in natural language processing. Macau: ACL, 2019:43–54.
- [37] OUYANG L, WU J, JIANG X, et al. Training language models to follow instructions with human feedback[J]. *Neural Information Processing Systems*, 2022, 35:27730–27744.
- [38] OpenAI. GPT-4 technical report[R]. San Francisco: OpenAI, 2023.