(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 117932066 A
(43)申请公布日 2024.04.26

(21)申请号 202410072559.9

(22)申请日 2024.01.18

(71)申请人 东北大学
地址 110819 辽宁省沈阳市和平区文化路3号巷11号

(72)发明人 王智慧 王浩彤 钟粮宇 吴刚

(74)专利代理机构 沈阳东大知识产权代理有限公司 21109
专利代理师 梁焱

(51)Int.Cl.
G06F 16/35(2019.01)
G06F 16/332(2019.01)
G06F 16/33(2019.01)
G06F 40/289(2020.01)
G06F 40/30(2020.01)
G06N 3/09(2023.01)
G06N 3/084(2023.01)
G06N 3/0455(2023.01)
G06N 3/047(2023.01)
G06N 3/0464(2023.01)
G06N 3/0442(2023.01)

权利要求书5页 说明书18页 附图5页

(54)发明名称

一种基于预训练的“提取-生成”式答案生成模型及方法

(57)摘要

本发明设计一种基于预训练的“提取-生成”式答案生成模型及方法;所述模型包括数据预处理模块、信息提取模块和答案生成模块;数据预处理模块对输入的文本进行标记,把和答案相关的输入句子打上标签,形成一个二分类数据集;所述信息提取模块提取二分类数据集中与答案有关的句子,同时屏蔽掉文本中的无用信息;所述答案生成模块将信息提取模块的输出作为输入,然后得到最终输出即答案;针对开放性问题,提出了“提取-生成”式两阶段答案生成模型;“提取”阶段,使用门控卷积神经网络提取与答案相关的信息,提升答案生成的精确性;“生成”阶段,将提取阶段作为输入,使用统一语言模型进行整理、去重,得到语义完整、语句通顺的长答案。

CN 117932066 A

1.一种基于预训练的“提取-生成”式答案生成模型,其特征在于,模型整体框架分为三个模块,即数据预处理模块、信息提取模块和答案生成模块,其中信息提取模块的输出是答案生成模块的输入;

所述数据预处理模块对输入的文本进行标记,把和答案相关的输入句子打上标签,形成一个二分类数据集,用于训练信息提取模块,打标签使用相似度匹配算法来实现;

所述信息提取模块提取二分类数据集中与答案有关的句子,同时屏蔽掉文本中的无用信息;包括三个部分,基于BERT的句子编码、基于GCNN的信息提取、以及最终的分类;

所述答案生成模块将Cs作为输入,然后得到最终输出即答案A,该模块使用预训练语言模型UniLM和BERT来进行实现;本质上该模块就是对信息提取模块得到的文本Cs进行整合,去除重复信息并生成具有完整语义的文本。

2.根据权利要求1所述的一种基于预训练的“提取-生成”式答案生成模型,其特征在于,所述信息提取模块首先使用BERT来对数据预处理模块输入的句子进行编码,编码后通过门控卷积神经网络Gated Convolutional Neural Network,GCNN来提取答案句子,其本质上是一个自监督的文本分类任务;在信息提取模块训练阶段,信息提取模块的输入是二分类数据集,输出为学习到的参数;在预测阶段,输入原始上下文文本C及训练学习到的参数,输出Cs,表示原始上下文文本C中所有与答案A相关的句子的集合。

3.一种基于预训练的“提取-生成”式答案生成方法,基于上述权利要求1一种基于预训练的“提取-生成”式答案生成模型实现,其特征在于,具体包括以下步骤:

步骤1:获取文本数据并利用数据预处理模块对其预处理,得到二分类数据集;

步骤2:基于步骤1获得的二分类数据集,利用信息提取模块提取与答案相关的句子Cs;

步骤3:基于步骤2得到的Cs,利用答案生成模块生成最终的答案;

所述答案生成模块包含BERT编码、UniLM建模、Copy机制、损失函数和优化。

4.根据权利要求3所述的一种基于预训练的“提取-生成”式答案生成方法,其特征在于,步骤1具体为:

对输入的文本数据(Context,Answer)进行预处理;首先为了获取更细粒度的文本信息,将按照逗号(,)和句号(。)对Context和Answer进行分句;然后使用相似度匹配算法来对Context中的每一个分句打标签;具体操作流程为:

首先对Answer中的分句按照句子长度排序,找到长度最长的分句,然后遍历Context中的每个分句并计算相似度指标,将最相似的分句打上标签1;循环执行上述过程,直到遍历完Answer中的所有分句;

所述计算相似度指标使用ROUGE指标,具体采用ROUGE-W计算,如下:

对于给定的两个句子X和Y;计算两个句子的加权最长公共子序列Weighted Longest Common Subsequence,WLCS;然后根据WLCS计算ROUGE-W,如下:

$$R_{wlcs}=f^{-1}\left(\frac{WLCS(X,Y)}{f(m)}\right)$$

$$P_{wlcs}=f^{-1}\left(\frac{WLCS(X,Y)}{f(n)}\right)$$

$$F_{wlcs}=\frac{(1+\beta^2)R_{wlcs}P_{wlcs}}{R_{wlcs}+\beta^2P_{wlcs}}$$

其中,R_{wlcs}表示召回率;P_{wlcs}表示准确率;m和n分别表示人工生成的参考摘要和模型生成的自动摘要的长度,f是权重函数,且需要满足f(x+y)>f(x)+f(y),f设为$f(k)=k^2$,f^{-1}是f的逆函数,即$f^{-1}(k)=k^{1/2}$,β设置为1.2,F_{wlcs}即为最终的ROUGE-W得分;x、y和k表示任意参数。

5.根据权利要求3所述的一种基于预训练的“提取-生成”式答案生成方法,其特征在于,步骤2所述信息提取模块包括三个部分,基于BERT的句子编码、基于GCNN的信息提取以及分类。

6.根据权利要求5所述的一种基于预训练的“提取-生成”式答案生成方法,其特征在于,所述基于BERT的句子编码,包含了三部分,即词嵌入Token Embedding、位置嵌入Position Embedding、分段嵌入Segment Embedding;

Token Embedding:词嵌入就是将输入句子的每个词转化成一个768维的向量;首先将输入的句子进行分词,然后将句子转化成BERT词表,最后在句子的开始和结尾处分别添加标识符“[CLS]”和“[SEP]”;

Position Embedding:位置嵌入的作用就是确定每个词在句子中的相对位置,使用正弦和余弦函数来计算;计算公式如下:

$$PE(pos,2i)=\sin(\frac{pos}{10000^{\frac{2i}{d_{model}}}})$$

$$PE(pos,2i+1)=\cos(\frac{pos}{10000^{\frac{2i}{d_{model}}}})$$

其中,pos表示单词的位置,i表示位置向量中每个值的索引,d_{model}表示位置向量的维度;

Segment Embedding:分段嵌入的作用是获取输入文本的句子间的语义关系;

所述基于GCNN的信息提取,具体为进行BERT编码、平均池化、降维、卷积处理得到一层卷积结果;过程如下:

输入文本经过数据预处理模块得到n个句子,即$\{s_1,...s_n\}$,然后经过BERT编码得到n个二维矩阵;

$$E_i=BERT(s_i),\{i=1,...,n\},E\in\mathbb{R}^{L\times768}$$

其中L表示n个句子中最长句子的词的个数,长度不足的句子用0补齐;

接下来,使用平均池化,将二维矩阵变换成一个768维的向量,即句向量;

$$e_i=AveragePooling(E_i),\{i=1,...,n\},e\in\mathbb{R}^{1\times768}$$

将句向量$\{e_i\}_{i=1}^{n}$输入到GCNN前需要通过一个全连接网络层进行降维;

$e_i'=e_iW'+b'$

其中,W′、b′表示可学习参数;

然后进行卷积运算即一层门控卷积神经网络的运算;二分类数据有X个句向量,每个句向量的维度是Y维,组成一个X*Y的矩阵;两个卷积核的大小都为$\frac{X}{2}*Y$,每个卷积核有Y个过滤

器filter,得到Y个特征图map,同时进行卷积运算,得到上下两个X*Y的矩阵,对其中一个矩阵进行sigmoid运算,然后与另一个矩阵按元素相乘,即门控运算,最终得到一层卷积后的结果;数学计算如下:

$$h(E)=(E*W+b)\otimes\sigma(E*V+c)$$

其中E＝X*Y表示输入层,表示上一层的输出或是文本C的句向量矩阵,$W=\frac{X}{2}*Y*Y$,b=Y,$V=\frac{Y}{2}*X*X$, c=Y,表示待学习的参数,σ表示sigmoid函数,⊗表示按元素相乘;

经过sigmoid运算并加入残差结构:

$$h(E)=E+(E*W+b)\otimes\sigma(E*V+c)$$

将上式进行变换得到:

$$h(E)=E\otimes(1-\sigma)+(E*W+b)\otimes\sigma$$

σ＝sigmoid(E*V+c)

向量矩阵所包含的语义信息以1-σ的概率直接通过,以σ的概率经过E*W+b变换后才通过;

所述分类的目的是保留与答案相关的句子,即只需要对一种类别进行建模,因此使用全连接网络将最后一层门控卷积的输出h^L降维至一维;

$h^L=[h_0,...h_{|x|}]$

其中h表示向量,x表示句子,|x|表示句子的长度,$h_{|x|}$表示第x个词的表示向量,h^L表示第L层的表示向量;

再使用sigmoid激活函数最终得到分类概率p;

$$p_i=\text{sigmoid}(h_i^L W''+b'')$$

其中b''、$W''\in\mathbb{R}^{M\times 1}$是待学习参数,M为隐藏特征$h^L$的维度,p的取值范围是(0,1);

损失函数使用二分类交叉熵;

$$\text{Loss}(x)=-\frac{1}{N}\sum_{i=1}^{N}[y_i\log p_i+(1-y_i)\log(1-p_i)]$$

最终输出与答案相关的句子Cs,$C_s=\{\{x_j\}_{j=m_1}^{n_1},...,\{x_j\}_{j=m_N}^{n_N}\}$,$1<=m_1<n_1<...<m_N<n_N<=L_c$,x是一个长为$L_c$的文本序列:$x=\{x_i\}_{i=1}^{Lc}$,m、n分别为起始位置与终点位置;$y_i$表示样本i的label,正类为1,负类为0;$p_i$表示样本i预测为正类的概率,并作为答案生成模块的输入。

7.根据权利要求3所述的一种基于预训练的“提取-生成”式答案生成方法,其特征在于,步骤3具体为:

步骤3.1:基于BERT对信息提取模块的输出Cs进行编码;

答案生成模块的输入是提取模块的输出Cs,再拼接上真实答案A;与提取模块一样,生成模块的编码部分包含词嵌入、分段嵌入以及位置嵌入;

词嵌入Token Embedding:将Token表示成768维的向量;

分段嵌入Segment Embedding:将提取到的文本Cs和真实答案A分割开来,具体表示为:“[CLS],C,[SEP],A”,即将“[CLS],C,[SEP]”的Segment Embedding设置为0,“A”的Segment Embedding设置为1;

Position Embedding:由于Cs和A都为长文本,长度和会超过512,而BERT只能对长度在512以内的文本进行位置编码,为了处理超长文本,采用了层次分解的编码方法;

BERT已经训练好的绝对位置编码向量表示为:$p_1,p_2,...,p_n$,其中n≤512,重新构建一套编码向量$q_1,q_2,...,q_m$,其中m>n;

$q_{(i-1)\times n+j}=\alpha u_i+(1-\alpha)u_j$

其中$u_1,u_2,...,u_n$表示编码变换的基,α∈(0,1),(i-1)×n+j=k;

当编码向量长度小于512时,u和原始位置编码一致,即当i=1时,q=p,因此有:

$q_j=\alpha u_1+(1-\alpha)u_j=p_j$

变换得到:

$$u_j=\frac{p_j-\alpha u_1}{1-\alpha}$$

因为$u_1=p_1$,代入上式到:

$$u_i=\frac{p_i-\alpha p_1}{1-\alpha},i=1,2,...,n$$

将嵌入层Token Embedding、Segment Embedding、Position Embedding按向量中相同位置元素相加相加作为答案生成模块的输入,同时使用BERT预训练好的参数初始化答案生成模块;

步骤3.2:所述UniLM有三种预训练任务分别是:双向语言模型Bidirectional LM、单向语言模型Left-to-Right LM和“Seq-to-Seq”语言模型,具体为:

UniLM首先设计Mask矩阵:将输入文本C_s和目标文本A同时传给BERT,通过返回的Segment_id构建Mask矩阵;

Mask矩阵的每一行表示一个输入,每一列表示一个输出,表示了输入和输出的关联关系,即Attention;具体计算方式如下:

假设输入文本C_s为$\{x_i\}_{i=1}^{|x|}$,经过嵌入层Token Embedding、Segment Embedding、Position Embedding得到UniLM的输入$H^0=[h_1,...h_{|x|}]$,其中h表示向量,x表示句子,|x|表示句子的长度,$h_{|x|}$表示底x个词的表示向量,由于是H^0所以表示底0层的表示向量;

然后计算Transformer的Query、Key、Value(Q、K、V)矩阵:

$Q_1=H^{l-1}W_1^Q,K_1=H^{l-1}W_1^K,V_1=H^{l-1}W_1^V$

其中H^{l-1}表示l-1层的UniLM输入,l表示注意力层数,W_1^Q,W_1^K,W_1^V表示可学习参数;

再计算注意力矩阵A:

$$A_l=\mathrm{softmax}\left(\frac{Q_lK_l^T}{\sqrt{d_k}}+M\right)$$

其中,K_1^T表示K_1的转置,d_k表示矩阵K的维度,M表示掩码矩阵;

$$M_{ij}=\begin{cases}0, & allow\ to\ attend\\ -\infty & prevent\ from\ attending\end{cases}$$

其中$M\in R^{|x|\times|x|}$表示Mask矩阵;

解码器输出的特征矩阵为OUTPUT,计算:

OUTPUT=A*V

其中$A=\{a_t\}_{t=1}^{T}$，a_t表示生成的词向量对t时刻的文本特征向量H的注意力分布，V即为上述计算得到的Value，$OUTPUT=\{output_t\}_{t=1}^{T}$，$output_t$表示t时刻解码器输出的特征向量；

步骤3.3：在UniLM建模的基础上引入Copy机制；

第一步：将上述t时刻解码器输出的特征向量$output_t$，经过线性变换以及softmax函数得到词表分布p_{vocab}；

p_{vocab}＝softmax(W′(W*$output_t$+b)+b′)

其中W,b,W′,b′表示可学习参数；

第二步：计算生成概率p_{gen}，即从词汇表中生成词的概率，那么1-p_{gen}表示从原文中Copy的概率；

p_{gen}＝sigmoid(W[x_t,$output_t$,a_t]+b)

其中x_t表示t时刻的目标向量，W,b表示可学习参数；那么最终的词表概率分布：

p(w)＝p_{gen}*p_{vocab}(w)+(1-p_{gen})*a_t

其中，w表示Token，当词表中没有时，p_{vocab}(w)的值为0，则预测的词从原文中得到，若是原文中没有的词，a_t值为0，预测的词从词表中生成，词表为编码部分分词后生成的；从原文中Copy高概率的词作为生层序列的一部分，能够控制生成结果的准确性；

步骤3.4：计算损失函数；

考虑Copy损失，直接对预测的概率取对数，计算如下所示：

$coploss_t$＝-logp(w_t)

考虑Coverage损失，计算如下所示：

$$c^t=\sum_{i=0}^{t-1}a^i$$

$$covloss_t=\sum_i \min(a_i^t,c_i^t)$$

其中c^t表示Coverage向量，是对先前时间步的注意力权重求和；a_i^t表示t时刻的注意力权重；$covloss_t \le \sum_i a_i^t=1$；

最终损失如所示：

L_t＝$coploss_t$+$ucovloss_t$

步骤3.5：使用Max函数对预测结果进行优化；

使用sparsemax函数代替传统的softmax函数，sparsemax计算如所示：

$$\text{sparsemax}(z)_j=\begin{cases}\dfrac{e^{z_i}}{\sum_{j\in\Omega_k}e^{z_j}}, i\in\Omega_k\\ 0, i\notin\Omega_k\end{cases}$$

其中Ω_k表示$z_1,z_2,\ldots,z_n$从大到小排列后的前k个元素的下标集合，即sparsemax只保留了前k个元素的概率，其余的直接设为0；

交叉熵损失函数表示如所示：

$$L=\log\left(\sum_{j\in\Omega_k}e^{z_j}\right)-z_t$$

其中z_t表示被Mask的原分词，z_j表示预测的结果。

一种基于预训练的“提取-生成”式答案生成模型及方法

技术领域

[0001] 本发明属于自然语言处理领域,具体涉及一种基于预训练的“提取-生成”式答案生成模型及方法。

背景技术

[0002] 随着互联网的高速发展,以及各大应用平台的普及,例如微博、抖音等,互联网世界已经从原来的少部分人创作,发展为人人都是自媒体、人人都是信息的生产者,真正的进入了信息大爆炸时代。互联网上每天都会产生海量的数据,如何利用好这些数据,从中提取有效信息,推动人类社会进步,一直都是自然语言处理(Natural Language Processing,NLP)中十分重要的研究内容。

[0003] 工业界和学术界根据不同的应用场景、特定的研究方法,将自然语言处理细分为多个任务,例如信息抽取(将非结构化或半结构化描述的自然语言文本数据转化成结构化特征数据)、信息检索(对大规模文本进行索引)、文本生成(让计算机能够像人类一样进行写作)等。

[0004] 问答对生成是近年来热门的自然语言处理细分任务之一。虽然互联网上有海量的数据,但大都是陈述式的文本,而与之对应的问题文本却十分匮乏,这导致诸如自动阅读理解、问答系统等任务缺乏足够的训练数据。传统方法往往是通过请专家学者编写问题等人工标注形式来生成问答对,虽然可以得到一些高质量数据,但是经济成本高,并且效率低,得到的数据也很有限,只能满足一些浅层神经网络模型。

[0005] 随着深度学习的发展,模型的规模越来越大,参数越来越多,所需要的训练数据也极其庞大,仅靠人工标注越来越不能满足需求。此外,在一些特定领域根本没有训练数据可用。因此,自然语言生成技术被广泛应用于问答对生成任务中。例如,对于互联网中存在的海量无标注文本,可以通过自然语言处理技术先标记出文本中的答案部分,再根据文本和答案生成对应的问题,从而形成问答对训练数据。根据模型的输入形式可将问题生成任务细分成不同的方向。第一,根据生成问题时是否加入答案信息将问题生成划分为,答案已知问题生成(Answer-aware QG)和答案未知问题生成(Answer-unaware QG)。第二,根据模型输入序列是句子还是段落将问题生成划分为,句子级问题生成(Sentence-Level QG)和段落级问题生成(Paragraph-Level QG)。句子级问题生成模型的输入是一句话,答案是句子中的一个实体或短语。段落级问题生成模型的输入是一个段落,答案是一个长文本,问题的回答需要整个段落的信息。因此问答对生成成为自然语言处理任务中要去解决的一道难题,而在问答对生成中,如何获得答案又变得至关重要。

[0006] 近些年出现了很多优秀的序列到序列(Seq-to-Seq)的模型框架,包括循环神经网络(Recurrent Neural Network,RNN)、长短时记忆神经网络(Long Short-Term Memory,LSTM)、门控循环网络(Gated Recurrent Unit,GRU)等,已经成为问答对生成的基线模型。而BERT等一系列预训练语言模型的提出,使基于预训练模型加微调的问答对生成技术凭借性能优势成为目前主流的研究方向。

[0007] 另外,问答对生成具有广泛的实际应用。(1)教育领域:为阅读理解材料生成问答对,然后将其用于测试人的阅读理解能力,也可为外语初学者提供更多的学习资料。(2)对话系统:问答对生成可以让聊天机器人主动向用户提问以开启一段对话,或者通过追问(即要求用户对问题进行反馈)的方式协助对话任务完成,例如点外卖的时候问用户喜欢吃什么,选择什么价位,从而实现精准推荐。(3)医疗领域:生成的问答对可以用于临床上评估人类的心理健康或提高心理健康水平。(4)标注数据集:因为目前最先进的神经网络模型都需要大规模的训练数据,而传统人工标注的方式代价高、效率低。使用问答对生成技术可以快速标注数据集。

[0008] 综上所述,问答对生成具有极大的科研价值,生成高质量问题有助于推动自然语言处理中其他任务的研究;同时也具有很大的应用价值,可以赋能教育、助力医疗等领域,还能丰富人们的生活。当前,虽然问答对生成已经取得很大进步,但是生成的问题依然存在很多缺点,限制了实际应用的发展。因此研究问答对生成技术,不断提高问答对生成模型的性能具有重要的现实意义。

[0009] 现有的问答对生成模型主要分为三类,即基于规则的生成模型、基于“Seq-to-Seq”的生成模型、基于预训练的生成模型。

[0010] 基于规则的生成模型即设计大量的将陈述句转化成问句的语义模板,其模型的性能完全取决于模板的设计水平,这要求研究者具有深厚的语言功底。为了改进这种纯粹的基于规则的系统,Heilman等人提出一种新的设计,引入了监督学习。该方法的思想就是首先使用基于规则的模型生成大量的问题,然后再使用监督学习的排序器从中选择出排名较高的问题。虽然引入排序算法一定程度上提高了生成质量。但是依然存在以下问题:(1)排序算法对人工设计的训练集有极大的依赖性。同时由于模板所生成的问题大都是输入句子的子集的重排序,所以很容易被回答。(2)基于规则的方法主要利用了句子的句法特征,而忽略了句子的语义特征。(3)基于规则的方法生成的句子的多样性远远不够。

[0011] 基于“Seq-to-Seq”的生成模型将问答对生成任务构建成一个“Seq-to-Seq”的学习任务,即将一个句子从文本段落直接映射到一个问题。具体来说,输入是给定的一段文本c(context),输出是与该文本相关的问题q(question),存在的问题有:将整个段落(包含多个句子)作为信息输入,长文本输入导致生成质量偏低;之前的生成模型生成的问题中往往会包含答案词,导致错误的问题生成等。

[0012] 基于预训练的生成模型的问答对生成思路是“预训练+微调”。不同于事实性问题,开放性问题的答案往往是由多个句子组成,且答案不是连续的文本,可能跨越不同的句子,甚至不同的段落。以往仅使用BERT预测文本跨度的方法并不能获取足够的有效信息,因此存在无法精确地从段落中获取全部的答案信息并将这些信息组织融合成完整语义的长答案等问题。

[0013] 现有的在部分问题以及通用领域达到较好效果的生成模型为基于预训练的答案生成模型。基于预训练语言模型的问答对生成思路是“预训练+微调”。其中“预训练”指的是,通过自监督学习从大规模数据中获得与具体任务无关的预训练模型。体现某一个词在一个特定上下文中的语义表征。“微调”指的是,在面对特定的下游任务时,适当调整模型结构,将上述预训练好的参数作为新模型的初始化状态,然后将该模型在特定任务的数据集上进行训练,此时由于预训练的参数已经获取了文本的语义信息,因此并不需要再花费大

量的资源去训练模型,实现了“微调”。目前基于Transformer的预训练模型,如BERT、ERNIE等,使得自然语言处理的多项任务都得到了质的提升。对于问答对生成任务,由于上述的预训练模型是双向的,不能直接使用。Dong等人提出UniLM(Unified Pre-trained Language Model),将自注意力掩码引入基于Transformer的编码器中,同时结合了单项、双向、“Seq-to-Seq”的语言模型,使得预训练模型可以应用到问答对生成任务中。

[0014] 然而,目前的研究主要集中于限定(事实)性问题的生成,问题的答案是给定文本段落中的一个实体或者短语。主要采用的方法是,以最大似然估计为训练目标,使用语言模型去预测给定文章中的一个跨度,并最终得到的答案就是给定文章中的一个连续的文本。然而在许多领域,开放性问题(描述性问题、解释性问题等)占比很大,而且具有更广泛的意义。例如教育领域,开放性问题可以促进学生获取复杂知识,还能训练其推理能力;在人机交互领域,可以改进搜索引擎、建立开放领域对话系统等。

[0015] 不同于事实性问题,开放性问题的答案往往是由多个句子组成,且答案不是连续的文本,可能跨越不同的句子,甚至不同的段落。以往仅使用BERT预测文本跨度的方法并不能获取足够的有效信息,这导致生成结果不理想。目前主要的开放性问答对生成工作是通过使用各种方法训练深度神经网络从文章中找到候选答案,再根据候选答案生成问题.然而这些方法通常需要复杂的规则和大量的数据训练模型,且可能会出现候选答案与基于候选答案生成的问题的对应答案不一致或基于候选答案生成的问题无法从文中找到对应答案,这种情况称为问答对相关性差.

发明内容

[0016] 针对现有技术的不足,本发明设计一种基于预训练的“提取-生成”式答案生成模型及方法。

[0017] 一种基于预训练的“提取-生成”式答案生成模型是一个端到端的自监督训练模型,模型整体框架分为三个模块,即数据预处理模块、信息提取模块和答案生成模块,其中信息提取模块的输出是答案生成模块的输入;

[0018] 所述数据预处理模块对输入的文本进行标记,把和答案相关的输入句子打上标签,形成一个二分类数据集,用于训练信息提取模块,打标签使用相似度匹配算法来实现;

[0019] 所述信息提取模块提取二分类数据集中与答案有关的句子,同时屏蔽掉文本中的无用信息;包括三个部分,基于BERT的句子编码、基于GCNN的信息提取、以及最终的分类;

[0020] 信息提取模块首先使用BERT来对数据预处理模块输入的句子进行编码,编码后通过门控卷积神经网络Gated Convolutional Neural Network,GCNN来提取答案句子,其本质上是一个自监督的文本分类任务;在信息提取模块训练阶段,信息提取模块的输入是二分类数据集,输出为学习到的参数;在预测阶段,输入原始上下文文本C及训练学习到的参数,输出Cs,表示原始上下文文本C中所有与答案A相关的句子的集合;

[0021] 所述答案生成模块将Cs作为输入,然后得到最终输出即答案A,该模块使用预训练语言模型UniLM和BERT来进行实现;本质上该模块就是对信息提取模块得到的文本Cs进行整合,去除重复信息并生成具有完整语义的文本;

[0022] 一种基于预训练的“提取-生成”式答案生成方法,基于上述一种基于预训练的“提取-生成”式答案生成模型实现,具体包括以下步骤:

[0023] 步骤1:获取文本数据并利用数据预处理模块对其预处理,得到二分类数据集;

[0024] 对输入的文本数据(Context,Answer)进行预处理;首先为了获取更细粒度的文本信息,将按照逗号(,)和句号(。)对Context和Answer进行分句;然后使用相似度匹配算法来对Context中的每一个分句打标签;具体操作流程为:

[0025] 首先对Answer中的分句按照句子长度排序,找到长度最长的分句,然后遍历Context中的每个分句并计算相似度指标,将最相似的分句打上标签1;循环执行上述过程,直到遍历完Answer中的所有分句;

[0026] 所述计算相似度指标使用ROUGE指标,具体采用ROUGE-W计算,如下:

[0027] 对于给定的两个句子X和Y;计算两个句子的加权最长公共子序列Weighted Longest Common Subsequence,WLCS;然后根据WLCS计算ROUGE-W,如下:

[0028] $$R_{wlcs}=f^{-1}\left(\frac{WLCS(X,Y)}{f(m)}\right)$$

[0029] $$P_{wlcs}=f^{-1}\left(\frac{WLCS(X,Y)}{f(n)}\right)$$

[0030] $$F_{wlcs}=\frac{(1+\beta^2)R_{wlcs}P_{wlcs}}{R_{wlcs}+\beta^2P_{wlcs}}$$

[0031] 其中,R_{wlcs}表示召回率;P_{wlcs}表示准确率;m和n分别表示人工生成的参考摘要和模型生成的自动摘要的长度,f是权重函数,且需要满足f(x+y)>f(x)+f(y),f设为$f(k)=k^2$,f^{-1}是f的逆函数,即$f^{-1}(k)=k^{1/2}$,β设置为1.2,F_{wlcs}即为最终的ROUGE-W得分;x、y和k表示任意参数;

[0032] 步骤2:基于步骤1获得的二分类数据集,利用信息提取模块提取与答案相关的句子Cs;

[0033] 所述信息提取模块包括三个部分,基于BERT的句子编码、基于GCNN的信息提取以及分类;

[0034] 所述基于BERT的句子编码,包含了三部分,即词嵌入Token Embedding、位置嵌入Position Embedding、分段嵌入Segment Embedding;

[0035] Token Embedding:词嵌入就是将输入句子的每个词转化成一个768维的向量;首先将输入的句子进行分词,然后将句子转化成BERT词表,最后在句子的开始和结尾处分别添加标识符“[CLS]”和“[SEP]”;

[0036] Position Embedding:位置嵌入的作用就是确定每个词在句子中的相对位置,使用正弦和余弦函数来计算;计算公式如下:

[0037] $$PE(pos,2i)=\sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

[0038] $$PE(pos,2i+1)=\cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

[0039] 其中,pos表示单词的位置,i表示位置向量中每个值的索引,d_{model}表示位置向量的维度;

[0040] Segment Embedding:分段嵌入的作用是获取输入文本的句子间的语义关系;

[0041] 所述基于GCNN的信息提取,具体为进行BERT编码、平均池化、降维、卷积处理得到一层卷积结果;过程如下:

[0042] 输入文本经过数据预处理模块得到n个句子,即$\{s_1,...s_n\}$,然后经过BERT编码得到n个二维矩阵;

[0043] $$E_i = BERT(s_i), \{i = 1,...,n\}, E \in \mathbb{R}^{L\times 768}$$

[0044] 其中L表示n个句子中最长句子的词的个数,长度不足的句子用0补齐;

[0045] 接下来,使用平均池化,将二维矩阵变换成一个768维的向量,即句向量;

[0046] $$e_i = AveragePooling(E_i), \{i = 1,...,n\}, e \in \mathbb{R}^{1\times 768}$$

[0047] 将句向量$\{e_i\}_{i=1}^{n}$输入到GCNN前需要通过一个全连接网络层进行降维;

[0048] $e_i' = e_i W' + b'$

[0049] 其中,W'、b'表示可学习参数;

[0050] 然后进行卷积运算即一层门控卷积神经网络的运算;二分类数据有X个句向量,每个句向量的维度是Y维,组成一个X*Y的矩阵;两个卷积核的大小都为$\frac{X}{2}*Y$,每个卷积核有Y个过滤器filter,得到Y个特征图map,同时进行卷积运算,得到上下两个X*Y的矩阵,对其中一个矩阵进行sigmoid运算,然后与另一个矩阵按元素相乘,即门控运算,最终得到一层卷积后的结果;数学计算如下:

[0051] $$h(E) = (E*W + b) \otimes \sigma(E*V + c)$$

[0052] 其中E=X*Y表示输入层,表示上一层的输出或是文本C的句向量矩阵,$W=\frac{X}{2}*Y*Y$,b=Y,$V=\frac{Y}{2}*X*X$,c=Y,表示待学习的参数,σ表示sigmoid函数,⊗表示按元素相乘;

[0053] 经过sigmoid运算并加入残差结构:

[0054] $$h(E) = E + (E*W + b) \otimes \sigma(E*V + c)$$

[0055] 将上式进行变换得到:

[0056] $$h(E) = E \otimes (1-\sigma) + (E*W + b) \otimes \sigma$$

[0057] σ=sigmoid(E*V+c)

[0058] 向量矩阵所包含的语义信息以1-σ的概率直接通过,以σ的概率经过E*W+b变换后才通过;

[0059] 所述分类的目的是保留与答案相关的句子,即只需要对一种类别进行建模,因此使用全连接网络将最后一层门控卷积的输出h^L降维至一维;

[0060] $h^L = [h_0,...h_{|x|}]$

[0061] 其中h表示向量,x表示句子,|x|表示句子的长度,$h_{|x|}$表示第x个词的表示向量,h^L表示第L层的表示向量;

[0062] 再使用sigmoid激活函数最终得到分类概率p;

[0063] $$p_i = \text{sigmoid}(h_i^L W'' + b'')$$

[0064] 其中b''、$W'' \in \mathbb{R}^{M\times 1}$是待学习参数,M为隐藏特征$h^L$的维度,p的取值范围是(0,1);

[0065] 损失函数使用二分类交叉熵;

[0066] $$\mathrm{Loss}(x)=-\frac{1}{N}\sum_{i=1}^{N}[y_i\log p_i+(1-y_i)\log(1-p_i)]$$

[0067] 最终输出与答案相关的句子Cs,

$C_s=\{\{x_j\}_{j=m_1}^{n_1},...,\{x_j\}_{j=m_N}^{n_N}\},1<=m_1<n_1<...<m_N<n_N<=L_c$,x是一个长为$L_c$的文本序列:$x=\{x_i\}_{i=1}^{Lc}$,m、n分别为起始位置与终点位置;$y_i$表示样本i的label,正类为1,负类为0;$p_i$表示样本i预测为正类的概率,并作为答案生成模块的输入;

[0068] 步骤3:基于步骤2得到的Cs,利用答案生成模块生成最终的答案;

[0069] 所述答案生成模块包含BERT编码、UniLM建模、Copy机制、损失函数和优化;

[0070] 步骤3.1:基于BERT对信息提取模块的输出Cs进行编码;

[0071] 答案生成模块的输入是提取模块的输出Cs,再拼接上真实答案A;与提取模块一样,生成模块的编码部分包含词嵌入、分段嵌入以及位置嵌入;

[0072] 词嵌入Token Embedding:将Token表示成768维的向量;

[0073] 分段嵌入Segment Embedding:将提取到的文本Cs和真实答案A分割开来,具体表示为:“[CLS],C,[SEP],A”,即将“[CLS],C,[SEP]”的Segment Embedding设置为0,“A”的Segment Embedding设置为1;

[0074] Position Embedding:由于Cs和A都为长文本,长度和会超过512,而BERT只能对长度在512以内的文本进行位置编码,为了处理超长文本,采用了层次分解的编码方法;

[0075] BERT已经训练好的绝对位置编码向量表示为:$p_1,p_2,...,p_n$,其中n≤512,重新构建一套编码向量$q_1,q_2,...,q_m$,其中m>n;

[0076] $q_{(i-1)\times n+j}=\alpha u_i+(1-\alpha)u_j$

[0077] 其中$u_1,u_2,...,u_n$表示编码变换的基,α∈(0,1),(i-1)×n+j=k;

[0078] 当编码向量长度小于512时,u和原始位置编码一致,即当i=1时,q=p,因此有:

[0079] $q_j=\alpha u_1+(1-\alpha)u_j=p_j$

[0080] 变换得到:

[0081] $$u_j=\frac{p_j-\alpha u_1}{1-\alpha}$$

[0082] 因为$u_1=p_1$,代入上式到:

[0083] $$u_i=\frac{p_i-\alpha p_1}{1-\alpha},i=1,2,...,n$$

[0084] 将嵌入层Token Embedding、Segment Embedding、Position Embedding按向量中相同位置元素相加相加作为答案生成模块的输入,同时使用BERT预训练好的参数初始化答案生成模块;

[0085] 步骤3.2:所述UniLM有三种预训练任务分别是:双向语言模型Bidirectional LM、单向语言模型Left-to-Right LM和“Seq-to-Seq”语言模型,具体为:

[0086] UniLM首先设计Mask矩阵:将输入文本C_s和目标文本A同时传给BERT,通过返回的Segment_id构建Mask矩阵;

[0087] Mask矩阵的每一行表示一个输入,每一列表示一个输出,表示了输入和输出的关

联关系,即Attention;具体计算方式如下:

[0088] 假设输入文本C_s为$\{x_i\}_{i=1}^{|x|}$,经过嵌入层Token Embedding、Segment Embedding、Position Embedding得到UniLM的输入$H^0=[h_1,...h_{|x|}]$,其中h表示向量,x表示句子,|x|表示句子的长度,$h_{|x|}$表示底x个词的表示向量,由于是H^0所以表示底0层的表示向量;

[0089] 然后计算Transformer的Query、Key、Value(Q、K、V)矩阵:

[0090] $Q_1=H^{l-1}W_1^Q,K_1=H^{l-1}W_1^K,V_1=H^{l-1}W_1^V$

[0091] 其中H^{l-1}表示l-1层的UniLM输入,l表示注意力层数,W_1^Q,W_1^K,W_1^V表示可学习参数;

[0092] 再计算注意力矩阵A:

[0093] $$A_l=\mathrm{softmax}\left(\frac{Q_lK_l^T}{\sqrt{d_k}}+M\right)$$

[0094] 其中,K_1^T表示K_1的转置,d_k表示矩阵K的维度,M表示掩码矩阵;

[0095] $$M_{ij}=\begin{cases}0, & allow\ \ to\ \ attend\\ -\infty & prevent\ \ from\ \ attending\end{cases}$$

[0096] 其中$M\in R^{|x|\times|x|}$表示Mask矩阵;

[0097] 解码器输出的特征矩阵为OUTPUT,计算:

[0098] OUTPUT=A*V

[0099] 其中$A=\{a_t\}_{t=1}^T$,a_t表示生成的词向量对t时刻的文本特征向量H的注意力分布,V即为上述计算得到的Value,$OUTPUT=\{output_t\}_{t=1}^T$,$output_t$表示t时刻解码器输出的特征向量;

[0100] 步骤3.3:在UniLM建模的基础上引入Copy机制;

[0101] 第一步:将上述t时刻解码器输出的特征向量$output_t$,经过线性变换以及softmax函数得到词表分布p_{vocab};

[0102] $p_{vocab}=\mathrm{softmax}(W'(W*output_t+b)+b')$

[0103] 其中W,b,W',b'表示可学习参数;

[0104] 第二步:计算生成概率p_{gen},即从词汇表中生成词的概率,那么1-p_{gen}表示从原文中Copy的概率;

[0105] $p_{gen}=\mathrm{sigmoid}(W[x_t,output_t,a_t]+b)$

[0106] 其中x_t表示t时刻的目标向量,W,b表示可学习参数;那么最终的词表概率分布:

[0107] $p(w)=p_{gen}*p_{vocab}(w)+(1-p_{gen})*a_t$

[0108] 其中,w表示Token,当词表中没有时,$p_{vocab}(w)$的值为0,则预测的词从原文中得到,若是原文中没有的词,a_t值为0,预测的词从词表中生成,词表为编码部分分词后生成的;从原文中Copy高概率的词作为生层序列的一部分,能够控制生成结果的准确性;

[0109] 步骤3.4:计算损失函数;

[0110] 考虑Copy损失,直接对预测的概率取对数,计算如下所示:

[0111] $coploss_t=-logp(w_t)$

[0112] 考虑Coverage损失,计算如下所示:

[0113] $$c^t=\sum_{i=0}^{t-1}a^i$$

[0114] $$covloss_t = \sum_i \min(a_i^t, c_i^t)$$

[0115] 其中c^t表示Coverage向量,是对先前时间步的注意力权重求和;a_i^t表示t时刻的注意力权重;$covloss_t \le \sum_i a_i^t = 1$;

[0116] 最终损失如所示:

[0117] $L_t = coploss_t + ucovloss_t$

[0118] 步骤3.5:使用Max函数对预测结果进行优化;

[0119] 使用sparsemax函数代替传统的softmax函数,sparsemax计算如所示:

[0120] $$\text{sparsemax}(z)_j = \begin{cases} \dfrac{e^{z_i}}{\sum_{j\in\Omega_k} e^{z_j}}, i \in \Omega_k \\ 0, i \notin \Omega_k \end{cases}$$

[0121] 其中Ω_k表示$z_1, z_2, \ldots, z_n$从大到小排列后的前k个元素的下标集合,即sparsemax只保留了前k个元素的概率,其余的直接设为0;

[0122] 交叉熵损失函数表示如所示:

[0123] $$L = \log\left(\sum_{j\in\Omega_k} e^{z_j}\right) - z_t$$

[0124] 其中z_t表示被Mask的原分词,z_j表示预测的结果。

[0125] 本发明有益技术效果:

[0126] 本发明提出了“提取-生成”两阶段模型,该模型摒弃了传统的只预测一个文本跨度作为答案的方法,而是将预测文本跨度与文本生成相结合。

[0127] 具体来说,“提取”阶段主要作用是获取与答案相关的信息,同时过滤掉大量的对答案生成无用的信息,从而提升答案生成的精确性。该阶段使用门控卷积神经网络(Gated Convolutional Neural Networks,GCNN)处理分类任务。首先用BERT预训练模型对输入文本进行向量化表示,然后输入到多层GCNN中进行二分类,最终得到与答案相关的句子。

[0128] “生成”阶段,主要作用是对“提取”阶段得到的相关句子进行整合、去重,从而得到语义完整、语句通顺的长答案。该阶段使用统一语言模型(Unified Language Model,UniLM)处理生成任务,首先将“提取”阶段输出的句子进行BERT向量化表示,然后输入到UniLM中进行编码、解码,生成最终的答案。为了确保生成的答案对原文本的忠诚度,在UniLM的基础上引入了Copy机制。此外,为了增强生成效果,在UniLM的解码阶段对传统的softmax函数进行了改进。

附图说明

[0129] 图1本发明实施例一种基于预训练的“提取-生成”式答案生成方法流程图;

[0130] 图2本发明实施例信息提取模块框架示意图;

[0131] 图3本发明实施例BERT编码层框架示意图;

[0132] 图4本发明实施例门控卷积运算示意图;

[0133] 图5本发明实施例门控卷积运算的信息流向示意图;

[0134] 图6本发明实施例基于UniLM的答案生成框架示意图;

[0135] 图7本发明实施例基于BERT的分层编码示意图;

[0136] 图8本发明实施例UniLM的单向语言模型示意图;

[0137] 图9本发明实施例UniLM的seq-to-seq结构图;

[0138] 图10本发明实施例用于做seq-to-seq任务的Mask矩阵示意图;

[0139] 图11本发明实施例Copy机制示意图。

具体实施方式

[0140] 下面结合附图和实施例对本发明做进一步说明;

[0141] 一种基于预训练的“提取-生成”式答案生成模型是一个端到端的自监督训练模型,模型整体框架分为三个模块,即数据预处理模块、信息提取模块和答案生成模块,其中信息提取模块的输出是答案生成模块的输入;信息提取模块框架示意图如附图2所示;

[0142] 所述数据预处理模块对输入的文本进行标记,把和答案相关的输入句子打上标签,形成一个二分类数据集,用于训练信息提取模块;只用于训练阶段,预测阶段不再执行;打标签具体使用相似度匹配算法来实现;

[0143] 所述信息提取模块提取二分类数据集的有效信息,同时屏蔽掉文本中的无用信息;包括三个部分,基于BERT的句子编码、基于GCNN的信息提取、以及最终的分类;

[0144] 首先,输入是一个句子序列,需要将其表示成向量才能送入后续的网络中,信息提取模块使用BERT来对数据预处理模块输入的句子进行编码,编码后通过门控卷积神经网络Gated Convolutional Neural Network,GCNN来提取与答案相关的句子,其本质上是一个自监督的文本分类任务;在信息提取模块训练阶段,信息提取模块的输入是二分类数据集,输出为学习到的参数;在预测阶段,输入是上下文文本C及训练学习到的参数,输出为C的一个子集,记为Cs,$C_s = \{\{x_j\}_{j=m_1}^{n_1},...,\{x_j\}_{j=m_N}^{n_N}\}, 1 <= m_1 < n_1 < ... < m_N < n_N <= L_c$;Cs表示C中所有与答案A相关的句子的集合,x是一个长为L_c的文本序列:$x = \{x_i\}_{i=1}^{Lc}$,m、n分别为起始位置与终点位置;

[0145] 所述答案生成模块将Cs作为输入,然后得到最终输出,即答案A,该模块使用预训练语言模型UniLM和BERT来进行实现;本质上该模块就是对信息提取模块得到的文本Cs进行整合,去除重复信息并生成具有完整语义的文本;

[0146] 一种基于预训练的“提取-生成”式答案生成方法,基于上述一种基于预训练的“提取-生成”式答案生成模型实现,如附图1所示,具体包括以下步骤:

[0147] 步骤1:获取文本数据并利用数据预处理模块对其预处理,得到二分类数据集;

[0148] 在正式提取文本有效信息之前,需要对输入的文本数据(Context,Answer)进行预处理;首先为了获取更细粒度的文本信息,将按照逗号(,)和句号(。)对Context和Answer进行分句;

[0149] 因为信息提取模块的任务是得到与答案信息相关的文本,它只是一个中间结果,还需要将得到的文本输入到答案生成模块中,因此在该过程中全面去提取Context中的与Answer相关的分句;

[0150] 然后使用相似度匹配算法来对Context中的每一个分句打标签;具体操作流程是:

[0151] 首先对Answer中的分句按照句子长度排序,找到长度最长的分句,然后遍历

Context中的每个分句并计算相似度指标,将最相似的分句打上标签1;循环执行上述过程,直到遍历完Answer中的所有分句;

[0152] 所述计算相似度指标使用ROUGE指标,ROUGE是机器翻译、摘要生成、问答生成等领域常用的评价指标,通过将模型生成的文本序列与标准文本序列进行比较计算,得到对应的得分。ROUGE常用的有四种计算方式,即ROUGE-L、ROUHE-N、ROUGE-S、ROUGE-W。本节使用ROUGE-W来计算,其目的是让连续匹配的词得到更多的分数。具体采用ROUGE-W计算,如下:

[0153] 对于给定的两个句子X和Y;计算两个句子的加权最长公共子序列Weighted Longest Common Subsequence,WLCS;如下所示:

```
(1)For(i=0;i<=m;i++)
        c(i,j)=0//初始化 c 表
        w(i,j)=0//初始化 w 表
(2)For(i=1;i<=m;i++)
  For(j=1;j<=m;j++)
     If xi=yj Then
```

[0154]

```
            //位置 i-1 和 j-1 连续匹配的长度
            k=w(i-1,j-1)
            c(i,j)=c(i-1,j-1)+f(k+1)-f(k)
           //记住位置 i,j 连续匹配的长度
            w(i,j)=k+1
         Otherwise
            Ifc(i-1,j)>c(i,j-1)Then
               c(i,j)=c(i-1,j)
           w(i,j)=0//在 i,j 处不匹配
            Elsec(i,j)=c(i,j-1)
```

[0155]

```
            w(i,j)=0//在 i,j 处不匹配
(3)WLCS(X,Y)=c(m,n)
```

[0156] 其中,c为动态规划表,c(i,j)存储X中结束于单词xi处的WLCS得分以及Y中结束于单词yi处的WLCS得分,w(i,j)用于存储结束于表c位置i和j的连续匹配长度,f是表c(i,j)处连续匹配的函数;通过提供不同的权重函数f,参数化WLCS算法,为连续的序列匹配分配不同的分数;

[0157] 然后根据WLCS计算ROUGE-W,计算公式如下:

[0158] $$R_{wlcs}=f^{-1}\left(\frac{WLCS(X,Y)}{f(m)}\right)$$

[0159] $$P_{wlcs}=f^{-1}\left(\frac{WLCS(X,Y)}{f(n)}\right)$$

[0160] $$F_{wlcs}=\frac{(1+\beta^2)R_{wlcs}P_{wlcs}}{R_{wlcs}+\beta^2P_{wlcs}}$$

[0161] 其中,R_{wlcs}表示召回率;P_{wlcs}表示准确率;m和n分别表示人工生成的参考摘要和模型生成的自动摘要的长度,f是权重函数,且需要满足f(x+y)>f(x)+f(y),f设为$f(k)=k^2$,f^{-1}是f的逆函数,即$f^{-1}(k)=k^{1/2}$,β设置为1.2,F_{wlcs}即为最终的ROUGE-W得分;x、y和k表示任意参数;

[0162] 步骤2:基于步骤1获得的二分类数据集,利用信息提取模块提取与答案相关的句子Cs;

[0163] 所述信息提取模块包括三个部分,基于BERT的句子编码、基于GCNN的信息提取、以及最终的分类;

[0164] 所述基于BERT的句子编码,包含了三部分,即词嵌入TokenEmbedding、位置嵌入PositionEmbedding、分段嵌入SegmentEmbedding;如附图3所示;

[0165] 输入是经过预处理模块处理得到的context分句,需要将其表示成向量才能进行后续的处理,因为计算机更擅长处理数值类型的信息。自然语言处理中经常使用深度学习算法来获取和分析文本信息,这就需要先将句子的每个词映射到一个连续的低维空间中。

[0166] 使用BERT来对句子进行编码,BERT是Google提出的一个预训练语言模型,该模型从提出到现在一直是NLP领域研究的热点。BERT提供了多种不同训练规格的预训练模型,本节选用的是参数适中的BERTBase。

[0167] TokenEmbedding:词嵌入就是将输入句子的每个词转化成一个768维的向量;首先将输入的句子进行分词(不同的分词算法分词的粒度不同),然后将句子转化成BERT词表,最后在句子的开始和结尾处分别添加标识符“[CLS]”和“[SEP]”;例如,输入的句子是“Ilikecat”,处理后就得到{“[CLS]”,“I”,“like”,“cat”,“[SEP]”},即一共五个Token,然后再将这五个Token转化成一个5×768的矩阵。

[0168] PositionEmbedding:位置嵌入的作用就是确定每个词在句子中的相对位置,使用正弦和余弦函数来计算;计算公式如下:

[0169] $$PE(pos,2i)=\sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

[0170] $$PE(pos,2i+1)=\cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

[0171] 其中,pos表示单词的位置,i表示位置向量中每个值的索引,d_{model}表示位置向量的维度;

[0172] 如图7所示,为一个具体的嵌入示例。

[0173] 输入:Itseeseveryobjectasdistinctfromallotherobjects

[0174] 分词:[CLS]Itseeseveryobjectasdistinctfromallotherobjects[SEP]

[0175] 输出:e={e0,...e11},eE11x768即为BERT向量化后的句子,将用于后续的处理。

[0176] Segment Embedding:分段嵌入的作用是获取输入文本的句子间的语义关系;例如

输入“Ilike cat.I like doraemon”那么分词后为{“[CLS]”,“I”,“like”,“cat”,“[SEP]”,“I”,“like”,“doraemon”},一共八个Token,此时位置编码的前五个Token编码为0,后三个Token编码为1,即{0,0,0,0,0,1,1,1},并将其嵌入到8×768的矩阵中。

[0177] 所述基于GCNN的信息提取,具体为:

[0178] 门控卷积神经网络(GCNN)与传统卷积神经网络的主要区别点是门控机制,门控机制已被证明是循环神经网络(如LSTM、GRU)达到最先进性能的必要条件,在GCNN中,门控单元通过为梯度提供线性路径,同时保留非线性能力,减少了深度架构的梯度消失问题。在语言建模过程中,卷积神经网络是通过叠层,一层一层获取更长文本且更抽象特征的语义信息,这种分层分析类似于典型的语法形式主义,即逐级递增的去构建语法树,例如句子是由名词短语和动词短语组成,每个短语又包含进一步的内部结构。

[0179] 所述基于GCNN的信息提取,具体为进行BERT编码、平均池化、降维、卷积处理得到一层卷积结果;如附图4所示,过程如下:

[0180] 进行BERT编码、平均池化、降维、卷积处理得到一层卷积结果;过程如下:

[0181] 输入文本经过数据预处理模块得到n个句子,即{s1,...sn},然后经过BERT编码得到n个二维矩阵;

[0182] $E_i = BERT(s_i), \{i = 1,\dots,n\}, E \in \mathbb{R}^{L\times 768}$

[0183] 其中L表示六个句子中最长句子的词的个数,长度不足的句子用0补齐;

[0184] 接下来,使用平均池化,将二维矩阵变换成一个768维的向量,即句向量;

[0185] $e_i = AveragePooling(E_i), \{i = 1,\dots,n\}, e \in \mathbb{R}^{1\times 768}$

[0186] 将句向量$\{e_i\}_{i=1}^{n}$输入到GCNN前需要通过一个全连接网络层进行降维;

[0187] $e_i' = e_i W' + b'$

[0188] 其中,W′、b′表示可学习参数;

[0189] 然后进行卷积运算即一层门控卷积神经网络的运算;二分类数据有X个句向量,每个句向量的维度是Y维,组成一个X*Y的矩阵;两个卷积核,卷积核的大小都为$\frac{X}{2}*Y$,每个卷积核有Y个过滤器filter,得到Y个特征图map,同时进行卷积运算,得到上下两个X*Y的矩阵,对其中一个矩阵进行sigmoid运算,然后与另一个矩阵按元素相乘,即门控运算,最终得到一层卷积后的结果;数学计算如下:

[0190] $h(E) = (E*W + b) \otimes \sigma(E*V + c)$

[0191] 其中E=6x5表示输入层,其要么是上一层的输出,要么是一个样本的句向量矩阵;

[0192] W=3x5x5,b=5,V=3x5x5,c=5,表示待学习的参数,σ表示sigmoid函数,⊗表示按元素相乘;

[0193] 经过sigmoid运算并加入残差结构:

[0194] $h(E) = E + (E*W + b) \otimes \sigma(E*V + c)$

[0195] 将上式进行变换得到:

[0196] $h(E) = E \otimes (1-\sigma) + (E*W + b) \otimes \sigma$

[0197] σ=sigmoid(E*V+c)

[0198] 向量矩阵所包含的语义信息以1—σ的概率直接通过,以σ的概率经过E*W+b变换后

才通过,如图5所示,可清晰的看到卷积过程中的信息流向;

[0199] 所述分类的目的是尽可能保留多的与答案相关的句子,即只需要对一种类别进行建模,因此使用全连接网络将最后一层门控卷积的输出h^L降维至一维,再使用sigmoid激活函数最终得到分类概率p;

[0200] $h^L=[h_0,...h_{|x|}]$

[0201] 其中h表示向量,x表示句子,|x|表示句子的长度,$h_{|x|}$表示第x个词的表示向量,h^L表示第L层的表示向量;

[0202] $$p_i=\text{sigmoid}(h_i^L W''+b'')$$

[0203] 其中b''、$W''\in\mathbb{R}^{M\times 1}$是待学习参数,M为隐藏特征$h^L$的维度,p的取值范围是(0,1);

[0204] 损失函数使用二分类交叉熵;

[0205] $$\text{Loss}(x)=-\frac{1}{N}\sum_{i=1}^{N}[y_i\log p_i+(1-y_i)\log(1-p_i)]$$

[0206] 最终输出与答案相关的句子Cs,

$C_s=\{\{x_j\}_{j=m_1}^{n_1},...,\{x_j\}_{j=m_N}^{n_N}\},1<=m_1<n_1<...<m_N<n_N<=L_c$,x是一个长为$L_c$的文本序列:$x=\{x_i\}_{i=1}^{Lc}$,m、n分别为起始位置与终点位置;$y_i$表示样本i的label,正类为1,负类为0;$p_i$表示样本i预测为正类的概率,并作为答案生成模块的输入;

[0207] 步骤3:基于步骤2得到的Cs,利用答案生成模块生成最终的答案;

[0208] 文本生成的本质就是使用句子之前的信息去预测下一个词的概率分布,一般方法就是构建“Seq-to-Seq”模型对文本进行编码、解码、计算概率。主要选用UniLM和BERT来构建如图6所示,所述答案生成模块包含BERT编码、UniLM建模、Copy机制、损失函数和优化;

[0209] 步骤3.1:基于BERT对信息提取模块的输出Cs进行编码;

[0210] 答案生成模块的输入是提取模块的输出Cs,再拼接上真实答案A(此处的A指的是SQuAD、NewsQA和DuReader几个数据集$D_{train}=\{(c^{(i)},x^{(i)},y^{(i)}):i=1,...,N\}$中的x,即真实的答案);与提取模块一样,生成模块的编码部分包含词嵌入、分段嵌入以及位置嵌入;

[0211] 词嵌入TokenEmbedding:将Token表示成768维的向量;

[0212] 分段嵌入SegmentEmbedding:将提取到的文本C′和真实答案A分割开来,具体表示为:“[CLS],C,[SEP],A”,即将“[CLS],C,[SEP]”的SegmentEmbedding设置为0,“A”的SegmentEmbedding设置为1;

[0213] PositionEmbedding:由于C′和A都为长文本,长度和会超过512,而BERT只能对长度在512以内的文本进行位置编码,为了处理超长文本,采用了层次分解的编码方法;如图7所示:

[0214] BERT已经训练好的绝对位置编码向量表示为:$p_1,p_2,...,p_n$,其中n≤512,重新构建一套编码向量$q_1,q_2,...,q_m$,其中m>n;

[0215] $q_{(i-1)\times n+j}=\alpha u_i+(1-\alpha)u_j$

[0216] 其中$u_1,u_2,...,u_n$表示编码变换的基,$\alpha\in(0,1)$,$(i-1)\times n+j=k$;

[0217] 当编码向量长度小于512时,u和原始位置编码一致,即当i=1时,q=p,因此有:

[0218] $q_j=\alpha u_1+(1-\alpha)u_j=p_j$

[0219] 变换得到：

[0220] $$u_j = \frac{p_j - \alpha u_1}{1-\alpha}$$

[0221] 因为$u_1 = p_1$，代入上式到：

[0222] $$u_i = \frac{p_i - \alpha p_1}{1-\alpha}, i = 1,2,...,n$$

[0223] 默认情况下$\alpha = 0.4$。

[0224] 如果n＝512，那么i＝{1,2,3,4}，j＝{1,2,...,512}，可以处理2048的长度。

[0225] 当$\alpha < 0.5$时，式(3.17)中的$(1-\alpha)u_j$占主导地位，位置编码的区分度更好(j的取值有512种)，模型收敛更快；

[0226] 将嵌入层TokenEmbedding、SegmentEmbedding、PositionEmbedding按位置相加作为答案生成模块的输入，同时使用BERT预训练好的参数初始化答案生成模块；

[0227] 步骤3.2：所述UniLM有三种预训练任务分别是：双向语言模型Bidirectional LM、单向语言模型Left-to-Right LM和“Seq-to-Seq”语言模型，具体为：

[0228] BERT是多层Transformer结构堆叠而成，其核心是自注意力机制(Self-attention)，该机制使得每个位置(Position)上的Token都能够获取上下文的所有信息，这是不符合文本生成任务逻辑的。文本生成具有一定的依赖关系，即生成xt+1需要获取x<t的所有信息，同时又不能看到x>t的信息(因为这是未来的信息)。如果使用这种Self-attention来训练模型的生成能力，模型一开始就知道的所有答案，训练就没有意义。

[0229] BERT提出之前，一般使用“Seq-to-Seq”的模型来完成文本生成任务，例如RNN、LSTM、GRU等，其模型一般是由编码器(Encoder)和解码器(Decoder)构成。在解码器中信息传递是单向的，每个输出Token只取决于之前的所有输入Token，如图8所示；

[0230] 微软提出的统一语言模型(UniLM)将BERT巧妙的应用到了文本生成任务中，只需要单个BERT就可以完成“Seq-to-Seq”任务，无需区分编码器和解码器。UniLM将“Seq-to-Seq”任务当成“句子补全”任务来做。例如输入句子是：“whereareyoufrom”，输出句子是：“NEU”，UniLM将输入和输出的内容拼接在一起，即“[CLS]whereareyoufrom[SEP]NEU[SEP]”。然后训练语言模型，通过“[CLS]whereareyoufrom[SEP]”，逐个预测“N”、“E”、“U”，直到出现“[SEP]”截止。

[0231] UniLM认为Self-attention会泄露未来信息，因此使用Mask方式将预测部分的Token的注意力选择性的屏蔽掉，让他只能获取到上文的信息，而对于训练的内容应该获取全部的信息，如图9所示，为UniLM做“Seq-to-Seq”任务的结构图，训练部分为双向注意力，获取到上下文所有信息，而预测部分是单向注意力，只能得到上文的信息。

[0232] 具体来说，UniLM通过设计Mask矩阵来屏蔽对应信息，Mask矩阵取决于输入和输出，其信息来源于SegmentEmbedding，因此只需要将输入文本C_s和目标文本A同时传给BERT，通过返回的Segment_id构建Mask矩阵，如图10所示，就是根据上述实例构建的Mask矩阵；

[0233] Mask矩阵的每一行表示一个输出，每一列表示一个输出，矩阵就表示了输入和输出的关联关系，即Attention；图10中灰色表示有关联，白的表示没有关联，例如输出“N”只与“[CLS]whereareyoufrom[SEP]”有关联，即“N”是由“[CLS]whereareyoufrom[SEP]”的整

个信息预测得到的。而“E”的预测不仅取决于“[CLS]whereareyoufrom[SEP]”还包含了上一个预测“N”的信息。这完全符合“Seq-to-Seq”任务的生成逻辑。具体计算方式如下：

[0234] 假设输入文本C_s为$\{x_i\}_{i=1}^{|x|}$，经过嵌入层TokenEmbedding、SegmentEmbedding、PositionEmbedding得到UniLM的输入$H^0=[h_1,...h_{|x|}]$，其中h表示向量，x表示句子，|x|表示句子的长度，$h_{|x|}$表示底x个词的表示向量，由于是H^0所以表示底0层的表示向量；然后计算Transformer的Query、Key、Value(Q、K、V)矩阵：

[0235] $Q_l=H^{l-1}W_l^Q,K_l=H^{l-1}W_l^K,V_l=H^{l-1}W_l^V$

[0236] 其中H^{l-1}表示l-1层的UniLM输入，l表示注意力层数，W_l^Q,W_l^K,W_l^V表示可学习参数；再计算注意力矩阵A：

[0237] $$A_l=\mathrm{softmax}\left(\frac{Q_lK_l^T}{\sqrt{d_k}}+M\right)$$

[0238] 其中，K_l^T表示K_l的转置，d_k表示矩阵K的维度，M表示掩码矩阵；

[0239] $$M_{ij}=\begin{cases}0, & allow\ \ to\ \ attend\\ -\infty & prevent\ \ \ from\ \ attending\end{cases}$$

[0240] 其中$M\in R^{|x|\times|x|}$表示Mask矩阵；

[0241] 解码器输出的特征矩阵为OUTPUT，计算：

[0242] OUTPUT=A*V

[0243] 其中$A=\{a_t\}_{t=1}^T$，a_t表示生成的词向量对t时刻的文本特征向量H的注意力分布，V即为上文计算得到的Value，$OUTPUT=\{output_t\}_{t=1}^T$，$output_t$表示t时刻解码器输出的特征向量；

[0244] 步骤3.3:在UniLM建模的基础上引入Copy机制；

[0245] Copy机制常用于“Seq-to-Seq”模型中，是文本生成软约束(softconstraint)非常有效的方案，其目的是使生成的文本更加忠于原文。Copy机制在计算当前要预测Token的概率分布时，不仅考虑解码器中输出的上下文表示向量，同时还需要考虑使用编码器中原始文本构建的注意力来计算概率分布，最后通过权重综合两个概率分布，得到最终的分布。如图11所示，具体计算规则如下。

[0246] 第一步:将上述t时刻解码器输出的特征向量$output_t$，经过线性变换以及softmax函数得到词表分布p_{vocab}；

[0247] $p_{vocab}=\mathrm{softmax}(W'(W*output_t+b)+b')$

[0248] 其中W,b,W',b'表示可学习参数；

[0249] 第二步:计算生成概率p_{gen}，即从词汇表中生成词的概率，那么$1-p_{gen}$表示从原文中Copy的概率；

[0250] $p_{gen}=\mathrm{sigmoid}(W[x_t,output_t,a_t]+b)$

[0251] 其中x_t表示t时刻的目标向量，W,b表示可学习参数；那么最终的词表概率分布：

[0252] $p(w)=p_{gen}*p_{vocab}(w)+(1-p_{gen})*a_t$

[0253] 其中，w表示Token，当词表中没有时，$p_{vocab}(w)$的值为0，则预测的词从原文中得到，若是原文中没有的词，a_t值为0，预测的词从词表中生成，词表为编码部分分词后生成的；从原文中Copy高概率的词作为生层序列的一部分，能够控制生成结果的准确性；

[0254] 步骤3.4:计算损失函数;

[0255] 一个是Copy损失,直接对预测的概率取对数,计算如下所示:

[0256] $coloss_t = -logp(w_t)$

[0257] 另一个是Coverage损失,计算如下所示:

[0258] $$c^t = \sum_{i=0}^{t-1} a^i$$

[0259] $$covloss_t = \sum_i \min(a_i^t, c_i^t)$$

[0260] 其中c^t表示Coverage向量,是对先前时间步的注意力权重求和;a_i^t表示t时刻的注意力权重;$covloss_t \le \sum_i a_i^t = 1$;

[0261] 最终损失如所示:

[0262] $L_t = coloss_t + ucovloss_t$

[0263] 步骤3.5:使用Max函数进行优化;

[0264] 使用sparsemax函数代替传统的softmax函数,sparsemax计算如所示:

[0265] $$\text{sparsemax}(z)_j = \begin{cases} \dfrac{e^{z_i}}{\sum_{j\in\Omega_k} e^{z_j}}, i \in \Omega_k \\ 0, i \notin \Omega_k \end{cases}$$

[0266] 其中Ω_k表示$z_1, z_2, \ldots, z_n$从大到小排列后的前k个元素的下标集合,即sparsemax只保留了前k个元素的概率,其余的直接设为0;

[0267] 交叉熵损失函数表示如所示:

[0268] $$L = \log\left(\sum_{j\in\Omega_k} e^{z_j}\right) - z_t$$

[0269] 其中z_t表示被Mask的原分词,z_j表示预测的结果。

[0270] 本发明一种“提取-生成”式答案生成模型包括三个模块:数据处理模块、信息提取模块以及答案生成模块。数据预处理模块介绍了使用相似度匹配的方法对数据进行预处理,得到标记数据;信息提取模块介绍了使用门控卷积神经网络进行文本分类,包含了句子编码、网络结构、解码优化等内容;答案生成模块介绍了使用统一语言模型进行答案生成,包含了编码、网络结构、Copy机制、Max函数优化等内容。

[0271] 针对开放性问题,本文提出了“提取-生成”式两阶段答案生成模型。“提取”阶段,使用门控卷积神经网络(GatedConvolutionalNeuralNetwork,GCNN)提取与答案相关的信息,同时过滤掉无用信息,提升答案生成的精确性;“生成”阶段,将提取的相关信息作为输入,使用统一语言模型(UnifiedLanguageModel,UniLM)进行整理、去重,得到语义完整、语句通顺的长答案。为了确保生成的结果对输入文本的忠诚度,在统一语言模型的基础上引入了Copy机制。此外,为了增强生成效果,在统一语言模型的解码过程中对传统的softmax函数进行了改进。

[0272] 为了更加公平的评估两个模块的性能,采用的数据集、评估方法与已有成熟的问答对生成模型的设置相同。从以下几个方面详细介绍实验,首先是实验准备,包括实验数据集、评估方法、实验所需软硬件环境等,其次介绍每个模型的细节设置,最后设计对比实验

并分析结果。

[0273] 选取大多数问答对生成模型广泛采用的问答对生成任务数据集:SQuAD、NewsQA和DuReader。

[0274] 根据本模型的设置,其问题的答案应该是文档的子集,且一个问答对对应唯一的文档。因此对于DuReader数据集,过滤掉答案不属于文档的数据项。对于SQuAD数据集中一个长文档对应多个问答对,将长文档拆分成子文档,保证每个子文档包含一个问答对。整个答案生成过程,分为两个过程,即提取模块和生成模块。如下表所示为这两个模块的主要实验参数。

[0275]

模块	参数	值
提取模块	字符最大长度	512
	编码向量维度	768
	初始参数	BERTBase
	GCNN 隐藏层维度	384
	GCNN 层数	6
	Dropout	0.1
	Threshold	0.2
	优化器	Adam
	Batch_size	64
	Epochs	20
生成模块	字符最大长度	1024
	编码向量维度	768
	初始参数	BERTBase
	优化器	AdamEMA
	学习率	2E-5
	Batch_size	8
	Epochs	50

[0276] 对于提取模块,设置了文本的最大句子个数为512,超过的部分直接截掉。使用BERTBase对输入句子的每个词进行编码,BERT输出的句子编码向量维度是768维。GCNN层数设为6层,其隐藏层维度是384维,Dropout设为0.1,使用Adam函数进行优化。为了尽可能多的获取相关信息Threshold设置为0.2。Batch_size设置为64,Epochs设置为20。

[0277] 对于生成模块,设置最长输入文本为1024,超过的部分被截掉。使用BERTBase对输入句子进行编码,输出的句子编码向量维度是768维。模型的初始化参数为BERTBase。使用AdamEMA进行优化,ema_momentum设为0.9999,学习率设为2E-5。Batch_size设置为8,Epochs设置为50。

[0278] 为了方便标记使用EGAG("Extraction Generation"Answer Generation)代替“提取-生成”式答案生成

[0279] 对“提取-生成”式答案生成模型EGAG进行了实验。通过与已有的仅基于预训练模型加微调的方法对比,测评了各模型的BLEU-1、Rouge-L以及METEOR分数,验证了EGAG的有效性。实验结果表明,见下表,本发明提出的EGAG模型对比基于BERT的答案生成模型有很大

的提升,在有些指标甚至超过了基于T5模型的答案生成模型。还可以看出EGAG模型更擅长处理长文本。通过消融实验可以看出sparsemax函数对EGAG模型的帮助最大,同时Copy机制也在使得生成结果更忠于原文。

[0280]

Dataset	Model	BLEU-1	Rouge-L	METEOR
SQuAD	BERT-AG	19.33	51.68	18.50
	UniLM-AG	21.40	52.21	19.66
	OneStop	24.14	52.28	22.42
	T5-AG	22.91	53.22	20.80
	EGAG	23.22	53.51	21.15
NewsQA	BERT-AG	26.15	53.37	35.44
	UniLM-AG	27.25	54.82	36.66
	OneStop	28.70	53.93	39.15
	T5-AG	28.15	56.09	37.77
	EGAG	28.05	56.40	37.80

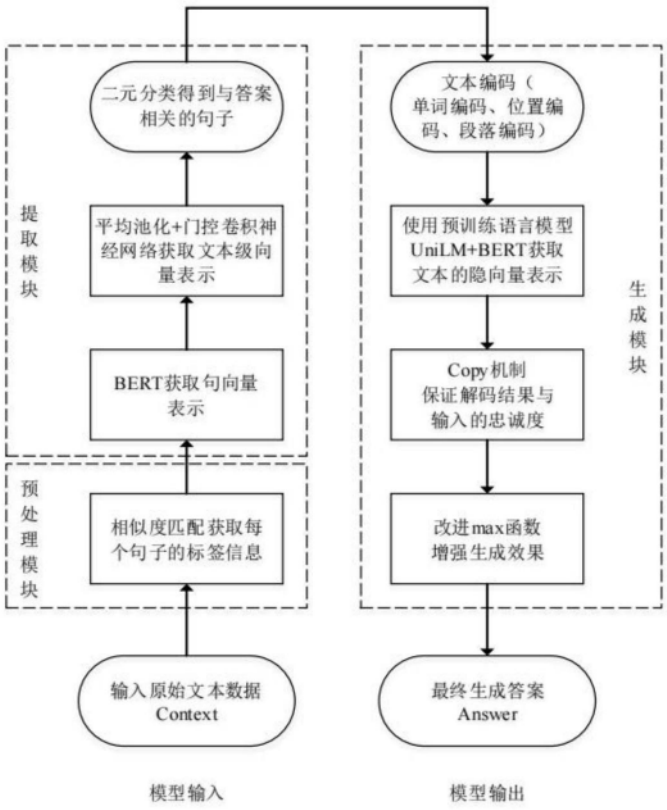
二元分类得到与答案相关的句子
平均池化+门控卷积神经网络获取文本级向量表示
BERT获取句向量表示
提取模块
相似度匹配获取每个句子的标签信息
预处理模块
输入原始文本数据 Context
文本编码（单词编码、位置编码、段落编码）
使用预训练语言模型UniLM+BERT获取文本的隐向量表示
Copy机制保证解码结果与输入的忠诚度
改进max函数增强生成效果
生成模块
最终生成答案 Answer
模型输入
模型输出

图1

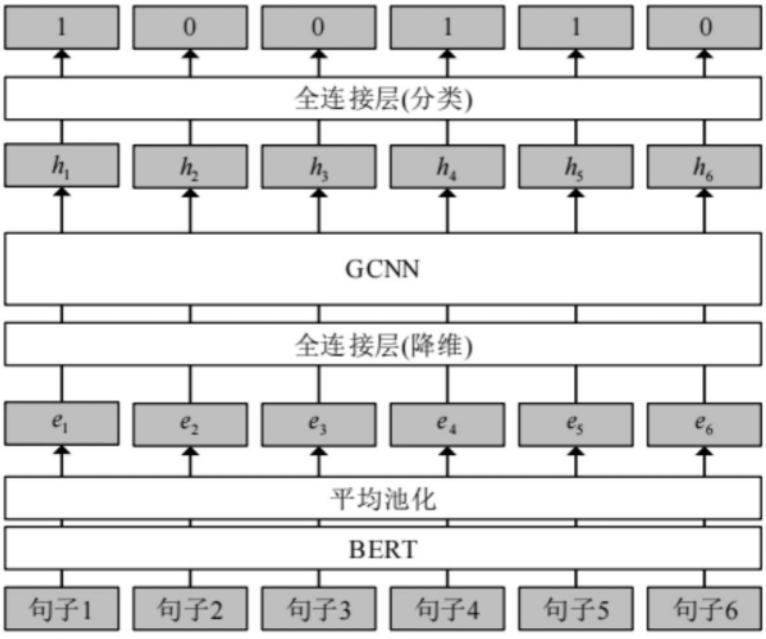
1
0
0
1
1
0
全连接层(分类)
h_1
h_2
h_3
h_4
h_5
h_6
GCNN
全连接层(降维)
e_1
e_2
e_3
e_4
e_5
e_6
平均池化
BERT
句子1
句子2
句子3
句子4
句子5
句子6

图2

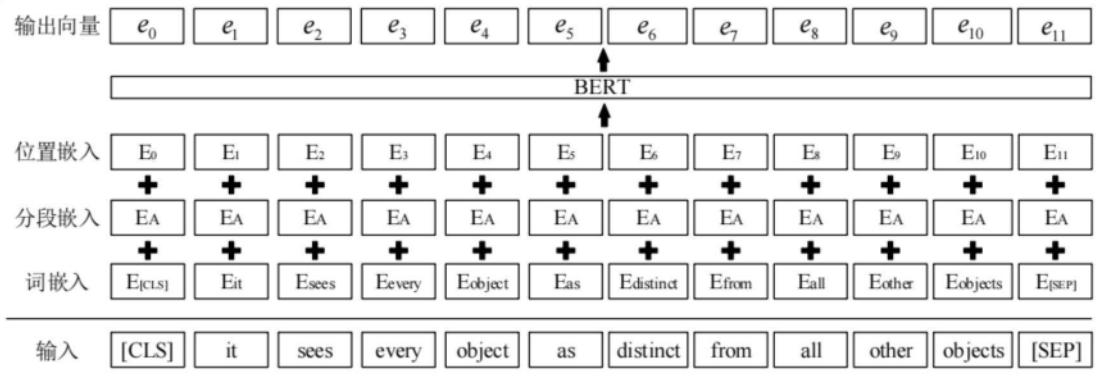
输出向量
e_0 e_1 e_2 e_3 e_4 e_5 e_6 e_7 e_8 e_9 e_{10} e_{11}
BERT
位置嵌入
E_0 E_1 E_2 E_3 E_4 E_5 E_6 E_7 E_8 E_9 E_{10} E_{11}
分段嵌入
E_A E_A E_A E_A E_A E_A E_A E_A E_A E_A E_A E_A
词嵌入
$E_{[CLS]}$ E_{it} E_{sees} E_{every} E_{object} E_{as} $E_{distinct}$ E_{from} E_{all} E_{other} $E_{objects}$ $E_{[SEP]}$
输入
[CLS] it sees every object as distinct from all other objects [SEP]

图3

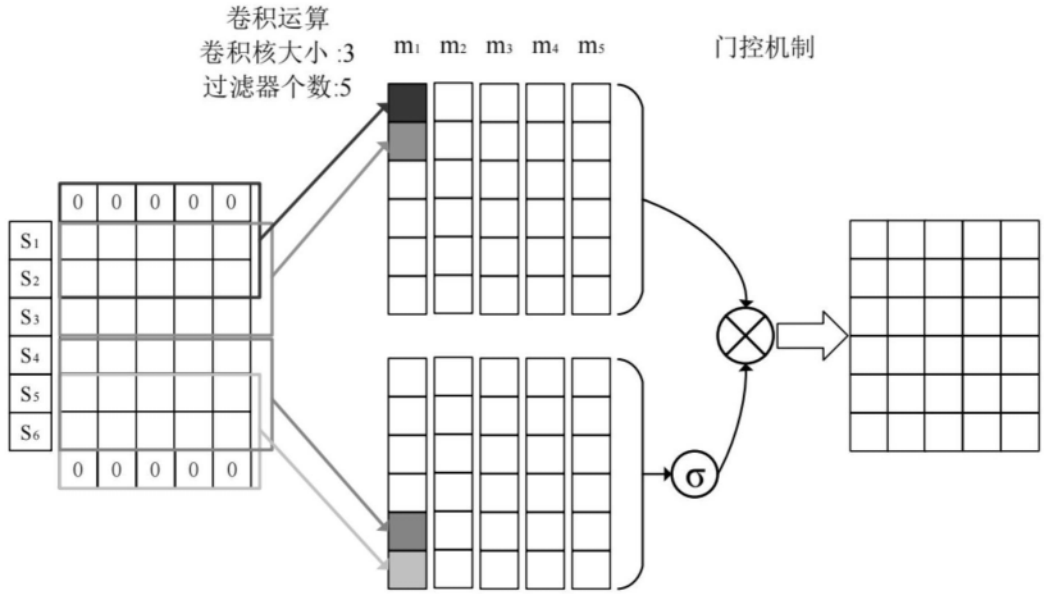
卷积运算
卷积核大小 :3
过滤器个数:5
m_1 m_2 m_3 m_4 m_5
门控机制
S_1 S_2 S_3 S_4 S_5 S_6
σ

图4

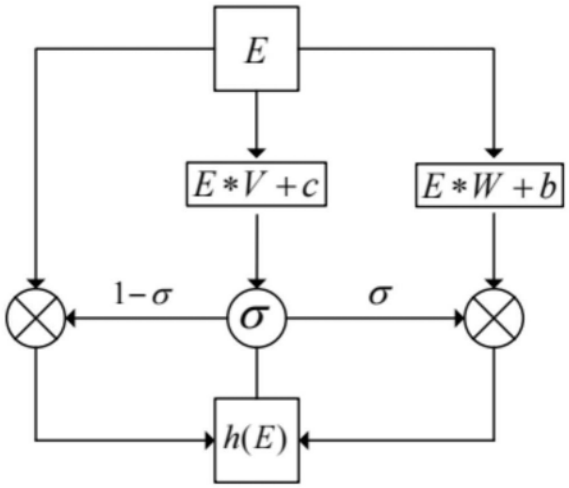
E
E*V+c
E*W+b
1−σ
σ
σ
h(E)

图5

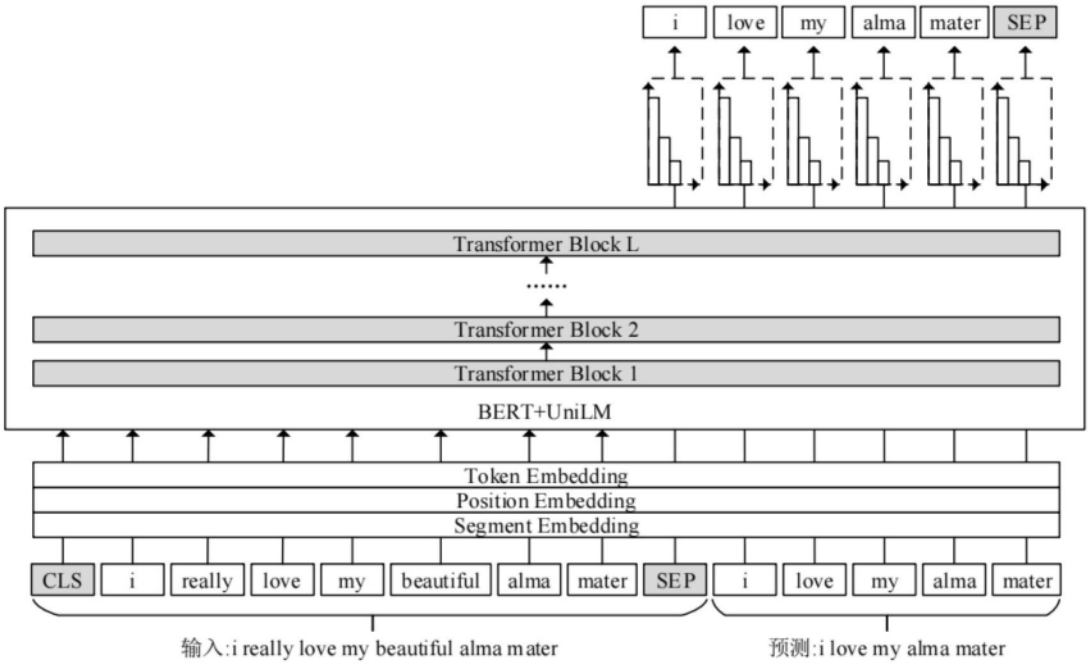
i
love
my
alma
mater
SEP
Transformer Block L
.....
Transformer Block 2
Transformer Block 1
BERT+UniLM
Token Embedding
Position Embedding
Segment Embedding
CLS
i
really
love
my
beautiful
alma
mater
SEP
i
love
my
alma
mater
输入:i really love my beautiful alma mater
预测:i love my alma mater

图6

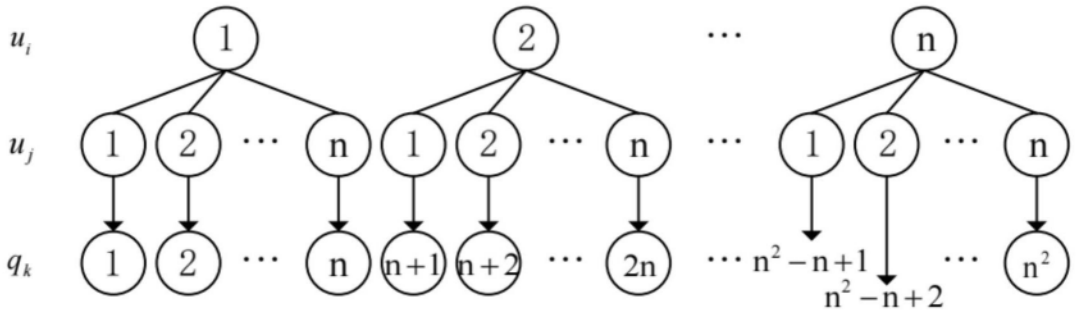
u_i
u_j
q_k
1
2
n
n+1
n+2
2n
n^2-n+1
n^2-n+2
n^2

图7

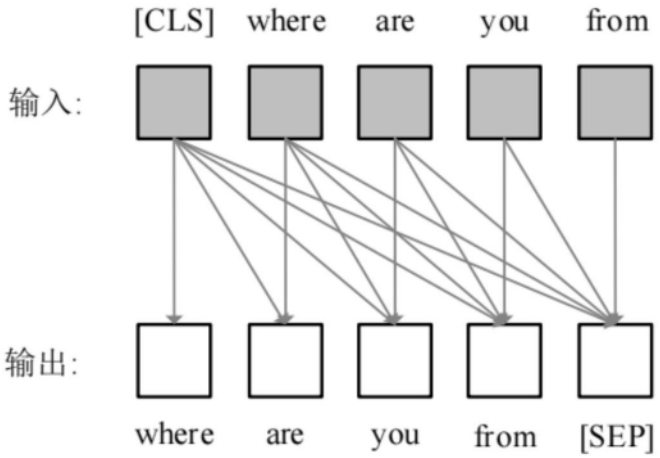
[CLS] where are you from
输入:
输出:
where are you from [SEP]

图8

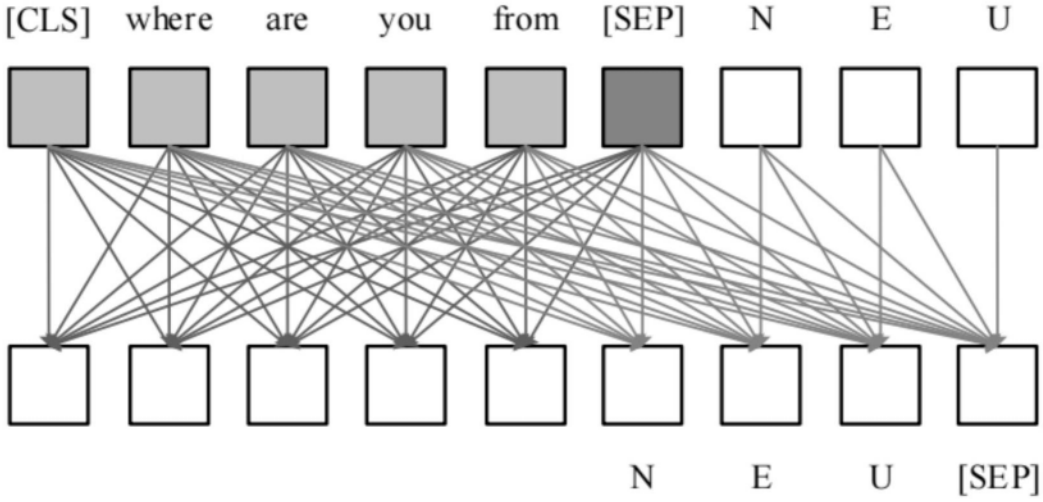
[CLS] where are you from [SEP] N E U
N E U [SEP]

图9

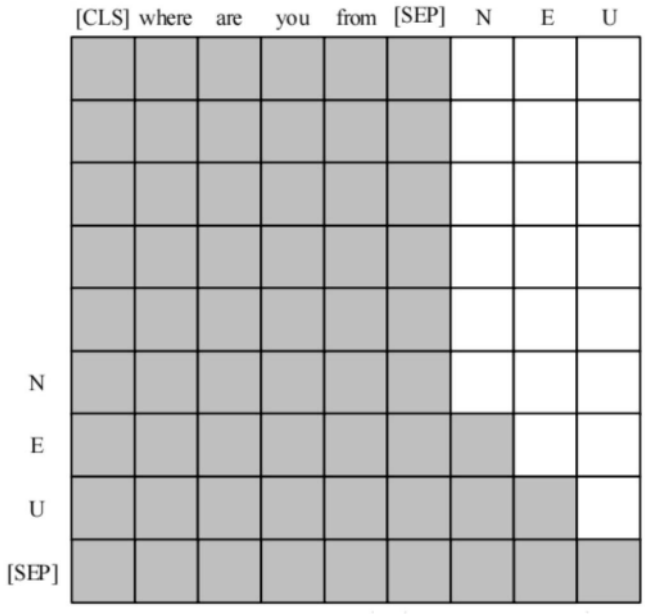
[CLS] where are you from [SEP] N E U
N
E
U
[SEP]

图10

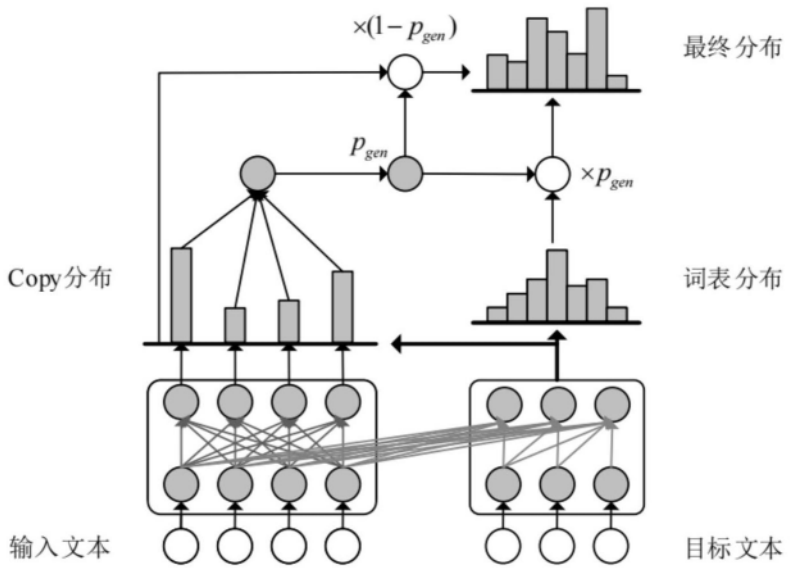
$\times(1-p_{gen})$
最终分布
p_{gen}
$\times p_{gen}$
Copy分布
词表分布
输入文本
目标文本

图11