深度记忆网络研究进展

刘建伟 王园方 罗雄麟
(中国石油大学(北京)自动化系 北京 102249)

摘 要 近年来,随着深度神经网络的快速发展,它在越来越多的领域中有了广泛的应用.深度神经网络模型在处理有序列依赖关系的预测问题时,需要利用之前学习到的信息进行记忆.在一般的神经网络模型中,数据经过多个神经元节点传输会损失很多关键的信息,因此需要具有记忆能力的神经网络模型,我们把它们统称为记忆网络.本文首先介绍了记忆网络的基础模型,包括循环神经网络(RNN)、长短期记忆神经网络(LSTM)、神经图灵机(NTM)、记忆神经网络(MN)和变送器(Transformer).其中,RNN 和 LSTM 是通过隐单元对前一时刻信息的处理来记忆信息,NTM 和 NM 是通过使用外部存储器来进行记忆,而变送器使用注意力机制来选择性记忆.本文对这些模型进了对比,并分析了各个记忆方法的问题和不足.然后根据基础模型的不同,本文对常见的记忆网络模型进行了系统的阐述、分类和总结,包括其模型结构和算法.接着介绍了记忆网络在不同领域和场景下的应用,最后对记忆网络的未来研究方向进行了展望.

关键词 循环神经网络;长短期记忆网络;记忆网络;神经图灵机;自然语言处理
中图法分类号 TP18 **DOI 号** 10.11897/SP.J.1016.2021.01549

Research and Development on Deep Memory Network

LIU Jian-Wei WANG Yuan-Fang LUO Xiong-Lin
(*Department of Automation, China University of Petroleum, Beijing* 102249)

Abstract In recent years, with the rapid development of deep neural networks, it has also made great progress in the fields of natural language processing and computer vision, especially in the areas of text reading, knowledge base question answering systems, and speech recognition and dialogue. However, the existing machine learning methods are difficult to memorize past information when performing tasks such as natural language processing. As the number of training iterations increases, they will gradually forget the long-term knowledge. Therefore, in order for machine learning to complete human-like memory tasks, machine learning models need the ability to remember long-term information. The deep neural network model needs to use the previously learned information to memorize when dealing with prediction problems with a wide range of sequence dependencies. In a general neural network model, data transmission through multiple neuron nodes loses a lot of key information, so a neural network model with memory capabilities is imperatively needed, which we collectively refer to as a memory network. With the development of recurrent neural networks in the 1980 s, there has been a better choice for modeling time series tasks, and the long-term and short-term memory networks expanded on this basis have also greatly improved short-term memory. However, in the face of natural language inference tasks, due to the lack of a "memory" function, these traditional models have defects in varying degrees. In the development of memory neural networks, there are deep and shallow models with

收稿日期:2019-09-10;在线出版日期:2020-02-16.本课题得到中国石油大学(北京)2018 年度前瞻导向及培育项目(2462018QZDX02)资助.**刘建伟**,博士,副研究员,主要研究方向为机器学习、智能信息处理、复杂系统的分析、预测与控制、算法分析与设计.E-mail:liujw@cup.edu.cn.**王园方**,硕士研究生,主要研究方向为机器学习.**罗雄麟**,博士,教授,主要研究领域为智能控制、复杂系统分析、预测与控制.

different depths. Compared with shallow memory networks, deep memory networks have more hidden layers, can extract features with higher levels of abstraction, and can learn more with fewer parameters with limited input samples and neuron nodes. Good fitting function. Therefore, the deep memory network model can show better memory capabilities. In view of the theoretical significance, application value and considerable development potential of memory networks, this article reviews the research progress of memory networks systematically, and establishes a good foundation for further in-depth study of memory network learning mechanisms and development of memory network applications. The article first summarizes the memory network in the first section, clarifies the development process of the memory network, and in the second section introduces the basic model of the memory network, including the recurrent neural network (RNN), the long-short term memory neural network (LSTM), the neural turing machine (NTM), the memory network (NM), and the Transformer. Among them, RNN and LSTM use the cell hidden unit to process the information of the previous moment to remember the information. NTM and NM use the external memory to fulfill remember function, and Transformer uses the attention mechanism to selectively remember the experiences learned in the past. This article compares these models and analyzes the problems and deficiencies of each memory method. The third section points out the various extensions of the previous three basic models of memory networks, and classifies and introduces them. The fourth section introduces the current application scenarios of the memory network, and finally points out the possible future development direction of the memory network in the fifth section. Afterwards, we introduce the extended model of the memory network and its application in different fields and scenarios. Finally, the future research direction of the memory network is prospected.

Keywords recurrent neural network; long short term memory network; memory network; neural turing machine; natural language processing

1 记忆网络概述

随着机器学习领域的发展，机器学习在自然语言处理和计算机视觉等领域也取得了很大的进步，特别是在文本阅读、知识库问答系统、语音识别和对话等方面取得了较大的进展.

但是，现有的机器学习方法在进行自然语言处理等任务时，很难对过去的信息进行记忆，随着训练迭代次数的增加，会逐渐遗忘长久以前的知识. 因此，要想使机器学习完成仿人类的记忆任务，就需要机器学习模型具有记住长期信息的能力.

例如，当机器面对一个问答系统，模型输入的是一个文本序列和一个问题，需要通过对这个文本序列的分析推理来得到这个问题的答案. 但是，通常的机器学习模型随着文本序列的不断输入，会逐渐遗忘过去输入的信息，其中可能包含了答案最关键的信息. 因此对于具有记忆功能的神经网络，可以在文本序列不断输入的过程中，将最可能的关键信息保存下来，在需要回答问题的时候进行调用，这样就避免了关键信息的遗漏.

随着 20 世纪 80 年代循环神经网络开始发展，对时间序列任务的建模有了较好的选择，在此基础上进行扩展的长短期记忆网络也在短期记忆方面有了很大的提高. 但是，面对自然语言的推理任务，由于“记忆”功能的缺乏，这些传统的模型都存在不同程度的缺陷.

在记忆神经网络的发展历程中，具有深层和浅层不同深度的模型. 与浅层记忆网络相比，深度记忆网络具有更多隐层，可以提取抽象层次更高的特征，并且能够在有限的输入样本和神经元节点的情况下，用更少的参数来学习到更好的拟合函数. 因此，深度记忆网络模型能够表现出更好的记忆能力.

鉴于记忆网络的理论意义、所蕴含的应用价值以及可观的发展潜力，本文对记忆网络的研究进展进行了系统性的综述，为进一步深入研究记忆网络

学习机制、开发记忆网络应用潜力确立良好的基础.

文中首先在第一节对记忆网络进行了概述，阐明记忆网络的发展过程，并在第二节着重介绍了记忆网络发展过程中最为重要的五种基础模型. 第三节指出了记忆网络在之前三种基础模型上的各种扩展，并进行了分类和介绍. 第四节介绍了目前记忆网络现有的应用场景，最后在第五节指出了记忆网络未来的可能发展方向.

在 20 世纪 80 年代，循环神经网络(Recurrent Neural Network，RNN)开始发展起来，由于 RNN 是一种对序列建模的人工神经网络，节点之间的连接形成沿着序列的有向图，这使得 RNN 可以表现时间序列的动态时间行为. 但是 RNN 由于其结构的局限性，在对比较长的序列进行预测过程中，其只能学习过去短时时间间隔的记忆. 为了解决这一问题，Hochreiter 和 Schmidhuber 在 1997 年提出了长短期记忆神经网络(Long Short-Term Memory，LSTM)[1]，对 RNN 的隐单元模块进行了扩展，在存储单元上增加了遗忘门和输入门，来对输入和前一时刻的隐状态包含的信息进行选择性记忆，并且因为 LSTM 状态值更新运算是加和的形式，避免了 RNN 在梯度下降时会产生的梯度消失问题，从而在一定程度上解决了学习序列长期依赖问题. 然而，传统深度学习模型如 LSTM 与 RNN 等使用隐单元的方法进行记忆，这种方法在将输入编码成向量时，会丢失很多信息，而且，这种通过隐状态和模型权重编码的记忆容量太小，没有足够容量记忆和区分过去的事实. 因此，在 2014 年，Weston 等人提出了记忆神经网络(Memory Network，MN)的概念[2]，首次把需要记忆的信息存储在单独的存储器中，在需要的时候进行读取和使用. 同年，Google DeepMind 团队提出了神经图灵机(Neural Turing Machine，NTM)[3]，将神经网络与图灵机结合，使用神经网络来实现图灵机计算模型中的读写操作. 在记忆神经网络的基础上，Sukhbaatar 等人提出了不需要监督的端到端记忆网络[4]. 之后，记忆网络领域迎来了快速发展的时期.

2017 年，Google 提出了解决序列到序列机器翻译问题的变送器(Transformer)模型[5]，用全注意力机制的结构代替了 RNN，在翻译任务上取得了更好的预测性能.

这些模型为之后记忆网络的发展做好了前期的理论工作，因此我们首先介绍以下这五种最基础的记忆网络模型.

2　记忆网络基础模型

记忆网络的概念在 2014 年正式提出以前，已经有很多学者对神经网络记忆功能进行了探索. 其中，大多数的模型是来自于循环神经网络和长短期记忆神经网络. 记忆网络基础模型包括循环神经网络、长短期记忆神经网络[1]、神经图灵机[3]、记忆网络[2]和变送器[4-5].

2.1　循环神经网络

循环神经网络是一种对序列建模的人工神经网络，如图 1 所示，是一个典型的 RNN 的展开图. 其节点之间的连接形成沿着序列的有向图.

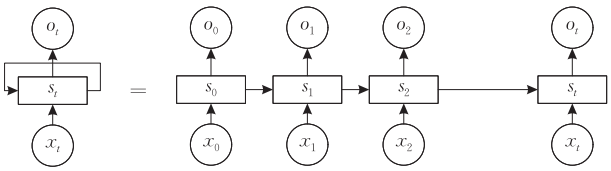

图 1　循环神经网络结构示意图

图中，假定给定序列是四个单词的句子，则 RNN 将展开成四层神经网络，每个单词为一层. 涉及的计算过程如下：

假定 x_t 是 t 时刻的输入. 例如，x_1 可以是与句子的第二个单词对应的单向向量.

s_t 是 t 时刻的隐状态，是网络的“记忆”功能得以实现的载体. s_t 是基于前一层隐状态的输出和当前时刻的输入计算得到的：

$$s_t = f(\boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\boldsymbol{s}_{t-1}) \tag{1}$$

其中函数 f 是作用于向量 $\boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\boldsymbol{s}_{t-1}$ 各个元素上的非线性激活函数，如 tanh 函数或者 ReLU 函数. 第一个节点的隐状态的 s_{-1} 通常被初始化为全零.

$$o_t = \mathrm{Softmax}(Vs_t) \tag{2}$$

这里，o_t 是 t 时刻的输出. 分为训练模式和预测模式，o_t 起着不同的作用. 训练模式下，t 时刻输出的字典中的单词已经给定，需要计算输出该单词概率最大的神经网络的模型参数 V. 模型参数 V 已给定后，如果想用 RNN 预测句子中的下一个可能出现的单词，o_t 是 t 时刻的输出，也就是字典中所有单词出现的概率组成的向量，出现概率最大的单词作为该时刻的输出.

可以将隐状态 s_t 视为网络存储器. s_t 捕获有关所有当前时刻之前步骤中发生的事件信息. t 时刻的输出仅根据时刻 t 的存储器 s_t 进行计算. 在实际中，s_t 通常不能捕获当前时刻之前太多时刻步骤中

的信息.

与传统深度神经网络不同,传统的深度神经网络每一层使用不同的参数,而 RNN 在所有步骤中共享相同的参数(U,V,W).这反映了每一步都执行相同的任务,只是输入不同而已.这大大减少了需要学习的参数个数.

图 1 中每个时刻都有输出,但实际任务中,这可能不是必须的.例如利用 RNN 输出一个句子时,我们关心的是最终整个句子的输出正确与否,而不是只关心每个时刻每个单词的输出.同样,也不需要在每个时间步骤都有输入.RNN 的主要特征是其隐状态能捕获有关序列的信息.

2.2 长短期记忆神经网络

LSTM 是一种特殊的 RNN,适合于处理和预测时间序列中间隔和延迟非常长的重要事件,对于 RNN 来说很难捕获较长序列之前的信息,而且训练的时候可能会出现梯度消失和梯度爆炸的问题,而 LSTM 对 RNN 隐单元的结构进行了修改,避免了这些问题.

LSTM 不断将过去和当前的输入信息传递到存储单元中,使用"门"来处理需要记忆和遗忘的信息,其结构如图 2 所示.

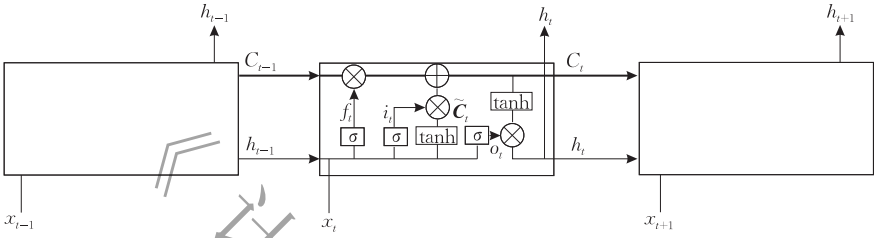

图 2 长短期记忆神经网络结构示意图

LSTM 的关键是单元状态,即贯穿图 2 顶部的水平线.单元状态有点像传送带.它穿越整条链,只有一些附带的线性交互作用,信息很容易以不变的方式流过.

LSTM 通过设计的称作为"门"的结构来去除或者增加信息到单元状态.门是一种让信息选择性通过的方式,由 sigmoid 型神经网络层和逐点乘法运算组成.sigmoid 层将输出 0 到 1 之间的数字,代表信息应该通过多少.值为 0 表示不让任何信息通过,而值为 1 表示让所有的信息通过.一个 LSTM 有三个这些门,用于保存和控制单元状态.

LSTM 首先是决定从单元状态中去除哪些信息.这个决定由叫"遗忘门"的 sigmoid 层来完成,如图 3 所示.它读取 h_{t-1} 和 x_t,并计算遗忘门 f_t 的值:

$$f_t=\sigma(W_{t-1}\cdot[h_{t-1},x_t])+b_f \tag{3}$$

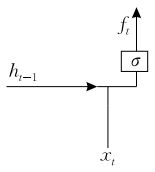

图 3 遗忘门结构示意图

其次是控制在单元状态中存储哪些新的信息.这由两部分组成,如图 4 所示,首先,一个 sigmoid 层计算"输入门".接下来,tanh 层计算一个状态门候选值向量 $\tilde{C}_t$.

$$i_t=\sigma(W_{t-1}\cdot[h_{t-1},x_t]+b_i) \tag{4}$$

$$\tilde{C}_t=\tanh(W_c\cdot[h_{t-1},x_t]+b_c) \tag{5}$$

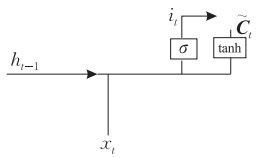

图 4 输入门结构示意图

接着是将旧的单元状态 $C_{t-1}\in(0,1)$ 更新为新的单元状态 C_t,如图 5 所示.将旧的单元状态 C_{t-1} 与 i_t 相乘,$f_t\times C_{t-1}$ 得到遗忘门 f_t 决定忘记的状态信息.然后,结合输入门 i_t 和 $\tilde{C}_t$ 来对单元状态进行更新.$f_t\times C_{t-1}$ 加上 $i_t\times\tilde{C}_t$,得到新的单元状态值 C_t:

$$C_t=f_t\times C_{t-1}+i_t\times\tilde{C}_t \tag{6}$$

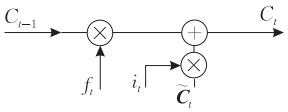

图 5 单元状态更新示意图

最后一步便是决定输出的内容,基于单元状态 C_t 计算输出,如图 6 所示.首先使用一个 sigmoid 层 o_t 决定输出单元状态值 C_t 的哪些部分.然后通过 tanh 层将单元状态值 C_t 压缩在 -1 到 1 之间,并将其与 o_t 相乘,得到当前时刻的隐状态 h_t.

$$o_t=\sigma(W_o\cdot[h_{t-1},x_t]+b_o) \tag{7}$$

$$h_t=o_t\times\tanh(C_t) \tag{8}$$

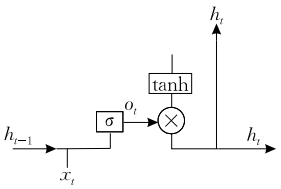

图 6　输出结构示意图

对上式总结，得到 LSTM 存储单元 c_t 和隐状态 h_t 随时间 t 的更新方程：

$$\begin{bmatrix} i_t \\ f_t \\ o_t \\ \hat{c}_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{bmatrix} W \cdot [h_{t-1}, x_t] \tag{9}$$

$$c_t = f_t \times c_{t-1} + i_t \times \hat{c}_t \tag{10}$$

$$h_t = o_t \times \tanh(c_t) \tag{11}$$

而且根据 Greff 等人的研究[6]，上述 LSTM 在各种数据集上有很好的性能，其中遗忘门、输出门和激活函数是 LSTM 最关键的成分. 研究还发现了使用随机梯度下降训练 LSTM 时，学习率是最重要的超参数，其次是网络规模，即神经元的个数.

2.3　神经图灵机

2014 年，Google DeepMind 提出神经图灵机[3]，首次使用外部存储器来增强神经网络的记忆能力，并使用了注意力机制. 神经图灵机模型结构类似于图灵在 1936 年提出的计算模型——图灵机，模拟人类进行数学计算的过程.

NTM 的结构如图 7 所示，由两部分组成：神经网络控制器和存储器. 控制器通过对输入和输出向量的读写，使用两个读写头有选择地对存储器进行读写操作，从而实现存储器中记忆内容的更新和拾取.

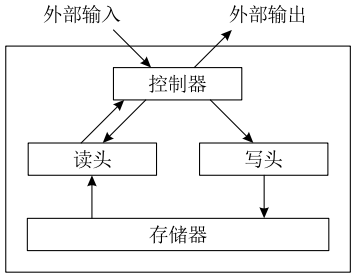

图 7　神经图灵机结构示意图

NTM 模型由控制器对存储器的读取和写入实现. NTM 的寻址机制和控制器的运算包含以下四个部分：

(1) 读取

$\boldsymbol{M}_t \in \mathbb{R}^{N \times M}$ 是 t 时刻存储器中的记忆矩阵，其中 N 是存储器位置的个数，M 是每个位置存储的向量的维数. $\boldsymbol{w}_t \in \mathbb{R}^N$ 是在 t 时刻由读头(read head)输出的 N 个位置上的加权归一化向量，即 $\boldsymbol{w}_t$ 满足 $\sum_i \boldsymbol{w}_t(i)=1, 0 \le \boldsymbol{w}_t(i) \le 1$，$\boldsymbol{w}_t$ 为 N 个位置 i 上的每一个读写位置分配一个权值 $\boldsymbol{w}_t(i)$.

读头返回的维数为 M 的读取向量 $\boldsymbol{r}_t$，是记忆矩阵行向量 $\boldsymbol{M}_t(i)$ 的凸组合：

$$\boldsymbol{r}_t \leftarrow \sum_i \boldsymbol{w}_t(i) \boldsymbol{M}_t(i) \tag{12}$$

这样定义的记忆向量 $\boldsymbol{r}_t$ 和权重向量 $\boldsymbol{w}_t$ 明显是可微分的.

(2) 写入

与 LSTM 的输入门和遗忘门类似，将写入操作分为两个部分：擦除和添加.

给定写头(write head)在 t 时刻的权重向量 $\boldsymbol{w}_t$ 和擦除向量 $\boldsymbol{e}_t$，其中权重向量 $\boldsymbol{w}_t$ 和擦除向量 $\boldsymbol{e}_t$ 的所有 M 个元素的取值范围为(0,1)之间. 那么前一时刻的记忆向量 $\boldsymbol{M}_{t-1}$ 被改写为

$$\widetilde{\boldsymbol{M}}_t(i) \leftarrow \boldsymbol{M}_{t-1}(i)[\boldsymbol{1} - \boldsymbol{w}_t(i)\boldsymbol{e}_t] \tag{13}$$

其中 $\boldsymbol{1}$ 是所有元素为 1 的行向量，并且 $\boldsymbol{M}_{t-1}(i)$ 与 $[\boldsymbol{1} - \boldsymbol{w}_t(i)\boldsymbol{e}_t]$ 的乘法为逐元素相乘，乘积结果为向量，而非标量. 因此，只有当第 i 个位置存储的权值 $\boldsymbol{w}_t(i)$ 和擦除向量 $\boldsymbol{e}_t$ 的分量都为 1 时，存储器单元中的元素才会被复位到 0. 如果权值 $\boldsymbol{w}_t(i)$ 或者擦除向量 $\boldsymbol{e}_t$ 为 $\boldsymbol{0}$ 时，那么存储器中的记忆向量保持不变.

每个写头还产生一个维数为 M 的附加向量 $\boldsymbol{a}_t$，在擦除之后附加到存储器中：

$$\boldsymbol{M}_t(i) \leftarrow \widetilde{\boldsymbol{M}}_t(i) + \boldsymbol{w}_t(i)\boldsymbol{a}_t \tag{14}$$

在 t 时刻，所有写头的擦除和附加操作运算结果即为存储器最后存储的内容. 其中，擦除和添加向量都具有 M 个独立分量，允许神经图灵机模型对 N 个位置中的每个存储器位置中的记忆矩阵行向量 $\boldsymbol{M}_t(i)$ 的每个元素进行细粒度控制.

(3) 寻址机制

为了得到权重向量 $\boldsymbol{w}_t$，NTM 使用了基于内容的寻址和基于位置的寻址两种寻址机制，其寻址流程图如图 8 所示.

基于内容的寻址. 第一种是基于记忆内容的寻址，控制器将注意力集中在基于当前记忆向量 $\boldsymbol{M}_t(i)$ 和控制器的键向量 $\boldsymbol{k}_t$ 值的相似度高的位置上. 时刻 t 控制器为每个读写头产生键向量 $\boldsymbol{k}_t$ 并与存储器 $\boldsymbol{M}_t$ 中的每个记忆向量 $\boldsymbol{M}_t(i)$ 计算相似度，归一化后得到读写权重 w_t^c：

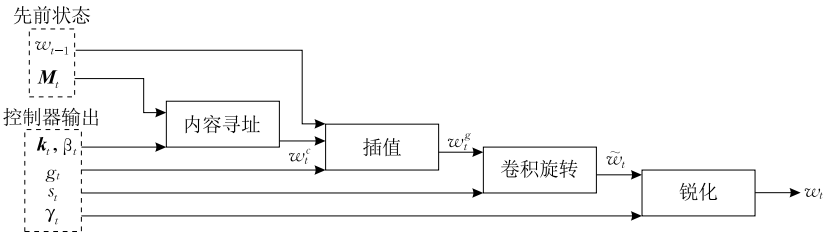

图 8 神经图灵机寻址机制流程图

$$w_t^c(i) \leftarrow \frac{\exp(\beta_t K[\boldsymbol{k}_t, \boldsymbol{M}_t(i)])}{\sum_j \exp(\beta_t K[\boldsymbol{k}_t, \boldsymbol{M}_t(j)])} \tag{15}$$

其中 K 是相似度函数，一般选择余弦相似度：

$$K[u,v] = \frac{u \cdot v}{\|u\| \cdot \|v\|} \tag{16}$$

β_t 是神经网路连接边参数，记忆内容的寻址方式的优点是检索简单，仅需要计算控制器键向量 $\boldsymbol{k}_t$ 与存储器 $\boldsymbol{M}_t$ 中的每个记忆向量 $\boldsymbol{M}_t(i)$ 的相似度，然后与存储器进行比较，产生精确的存储值.

基于位置的寻址. 第二种是基于位置的寻址，基于位置的寻址可实现对任意变量的寻址和表示，基于位置的寻址可以增强模型的适用范围和泛化效果.

基于位置的寻址机制通过实施加权的变换移位来实现跨存储器位置的简单迭代和随机跳转访问. 在变换移位之前，每个头产生范围为(0,1)的标量插值门 g_t，用来对前一时刻的头部权重 w_{t-1} 与当前时刻的权重 w_t^c 进行加权混合，得到门控权重 w_t^g：

$$w_t^g \leftarrow g_t w_t^c + (1-g_t) w_{t-1} \tag{17}$$

将 N 个存储器的位置从 0 索引到 $N-1$，引入 s_t 对 w_t^g 进行移位变换，表示为经如下循环卷积，得到移位后的权值 $\tilde{w}_t(i)$：

$$\tilde{w}_t(i) \leftarrow \sum_{j=0}^{N-1} w_t^g(j) s_t(i-j) \tag{18}$$

由于卷积操作会使权重随时间衰减和发散，导致本来集中于单个位置的焦点出现发散现象，因此需要对结果进行锐化(sharpening)操作，对每个头附加一个标量 γ_t 作为锐化因子：

$$\boldsymbol{w}_t(i) \leftarrow \frac{\tilde{w}_t(i)^{\gamma_t}}{\sum_j \tilde{w}_t(i)^{\gamma_t}} \tag{19}$$

这样就得到了新的权重向量，可以根据该向量进行相应的读写操作，对记忆进行修改.

(4) 控制器

NTM 的控制器可以选择 LSTM 或者前馈神经网络实现. 如果选择 LSTM，其隐状态可以作为存储器中记忆矩阵 $\boldsymbol{M}_t$ 的补充，并允许控制器使用混合跨越多个时间步的信息. 如果选择前馈神经网络，可以通过在不同的时刻读取内存中相同的位置信息来模拟递归网络的特性，但是前馈网络的局限性在于读写头数目有限，每个读写头在每个时间步只能操作一个内存向量.

2.4 记忆神经网络

记忆神经网络是指具有独立可读写操作的记忆模块的神经网络. MN 用于问题回答学习场景(Question Answers，QA).

MN 由一个记忆数组 m 和输入 I、泛化 G、输出 O 和回答 R 四个模块组成，结构如图 9 所示.

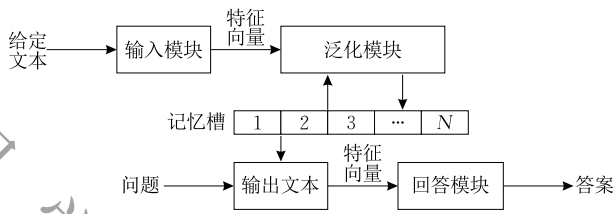

图 9 记忆网络结构示意图

这四个模块的执行过程如下：

(1) 当输入给定的文本 x，输入模块 I 使用标准预处理技术或编码方式，将其表示成内部特征 $I(x)$，然后通过泛化模块把 $I(x)$ 存储到存储器 m 的单元中.

(2) 根据新的要回答的问题文本输入 x，输出模块使用 N 个记忆槽中得分最高的 k 个记忆单元的索引值计算输出特征向量.

例如，当 $k=1$ 时，使用 N 个记忆槽中得分最高的一个记忆单元的索引值：

$$o_1 = O_1(x,m) = \arg\max_{i=1,\cdots,N} s_O(x, m_i) \tag{20}$$

其中 s_O 是输出特征向量的打分函数，由 o_1 索引值得到输出特征向量 $\boldsymbol{m}_{o_1}$.

当 $k=2$ 时，利用当前输入 x 和记忆单元 $\boldsymbol{m}_{o_1}$ 计算第二个记忆单元的索引值：

$$o_2 = O_2(x,m) = \arg\max_{i=1,\cdots,N} s_O([x, \boldsymbol{m}_{o_1}], m_i) \tag{21}$$

由 o_2 索引值得到输出特征向量 $\boldsymbol{m}_{o_2}$.

以此类推，最终输出序列 $x, \boldsymbol{m}_{o_1}, \boldsymbol{m}_{o_2}, \cdots, \boldsymbol{m}_{o_k}$.

(3) 输出响应模块可以把 $\boldsymbol{m}_{o_k}$ 作为输出 r，或者

选择字典里匹配得分最高的单词作为输出 r：

$$r=\arg\max_{w\in W} s_R([x,\boldsymbol{m}_{o_1},\boldsymbol{m}_{o_2}],w) \tag{22}$$

其中 $w\in W$ 是字典 W 中的单词，s_R 和 s_O 是具有相同形式的打分函数，形式如下：

$$s(x,y)=\Phi_x(x)^{\mathrm{T}}\boldsymbol{U}^{\mathrm{T}}\boldsymbol{U}\Phi_y(y) \tag{23}$$

其中，$\boldsymbol{U}$ 是一个 $n\times D$ 维的矩阵，D 表示特征个数，n 是嵌入的维数. Φ_x 和 Φ_y 的作用是把原始文本映射到 D 维的特征空间，特征空间选择用单词袋子表示.

在文本问题问答训练过程中，MN 使用大间隔排序损失函数(margin ranking loss)：

$$\begin{aligned}&\sum_{\bar{f}\neq\boldsymbol{m}_{o_1}}\max(0,\gamma-s_O(x,\boldsymbol{m}_{o_1})+s_O(x,\bar{f}))+\\&\sum_{\bar{f}'\neq\boldsymbol{m}_{o_2}}\max(0,\gamma-s_O([x,\boldsymbol{m}_{o_1}],\boldsymbol{m}_{o_2})+s_O([x,\boldsymbol{m}_{o_1}],\bar{f}'))+\\&\sum_{\bar{r}\neq r}\max(0,\gamma-s_R([x,\boldsymbol{m}_{o_1},\boldsymbol{m}_{o_2}],r)+s_R([x,\boldsymbol{m}_{o_1},\boldsymbol{m}_{o_2}],\bar{r}))\end{aligned} \tag{24}$$

完成文本问题问答功能. 其中，$\bar{f}$，$\bar{f}'$，$\bar{r}$ 是所有的错误答案，即训练目标是要求正确答案得分比错误答案的得分至少高一个 $\gamma>0$. 大间隔排序损失函数使用随机梯度下降训练算法学习问题问答.

MN 的实验数据集选择了 Fader 等人提出的大规模 QA 数据集[7]. 该数据集由 14M 语句组成，存储形式为(主体，关系，对象)三元组，例如(米尔恩，作者，小熊维尼)，并存储到 MN 模型的存储器模块中. 训练结合了由问题和相关的三元组构成的带有伪标签的 QA 对，以及 WikiAnswers 中的释义问题，例如“谁写了小熊维尼这本书?”.

该实验按照 Bordes 等人的方法[8]，通过在多个 $F1$ 分数测试系统上对返回的最佳答案进行重新排名的任务上进行了实验，并设置 MN 的记忆单元个数为 $k=1$.

Fader 等人在实验中设置初始权重向量为 0，迭代次数为 20，并将训练数据分成 10 个部分，还限制了每个系统只返回每个测试语句的前 100 个数据库查询结果. 所有输入的单词都是小写的并且提取了词干或者词根.

Bordes 等人在实验中设置单词和概念的维数为 20，评价函数的嵌入维数为 200，滑动窗口宽度为 13(即中心词的两边各有 6 个单词).

实验结果如表 1 所示，MN 模型的性能与 Bordes 的方法相当. 在添加了单词袋子表示后模型的得分有一定的提高，结果显示 MN 是一种完成大规模 QA 任务的可行方法.

表 1　记忆网络在大规模 QA 数据集上的实验结果对比

模型	$F1$
Fader 等人[7]	0.54
Bordes 等人[8]	0.73
MN(只使用单词嵌入表示)	0.72
MN(使用单词袋子表示)	0.82

实验任务还使用了自己建立的 QA 模型：具有四个角色，三个物体和五个房间，人物在房间内移动并拾取或者放下物体. 该任务的难度在于，当询问物体在哪里时，必须使用多个语句进行推理. 通过设置过去的时间步数(1 或 5)来控制任务难度，每个难度下包括了只提及角色移动的问题和角色携带物体的问题.

记忆网络在该实验任务上与 RNN 和 LSTM 进行对比. 对于基准 RNN 和 LSTM，通过使用基于时间的反向传播来建立语言模型[9]，并优化了超参数. 对于 MN 模型，在所有实验中将嵌入维数设置为固定值 100，学习率为 0.01，将裕度 γ 设置为 0.1 和 10 个训练时期. 实验结果如表 2 所示.

表 2　记忆网络在自己建立的 QA 任务下的实验结果对比

模型	难度 1		难度 5	
	角色/%	角色+物体/%	角色/%	角色+物体/%
RNN	60.9	27.9	23.8	17.8
LSTM	64.8	49.1	35.2	29.0
MN $k=1$	31.0	24.0	21.9	18.5
MN $k=1$+time	60.2	42.5	60.8	44.4
MN $k=2$+time	100.0	100.0	100.0	99.9

根据实验结果可以看到，RNN 和 LSTM 很难解决具有角色携带物体转移的任务，更是难以解决难度为 5 的问题. 这表明 RNN 无法实现长期记忆编码，随着过去的时间步数的增加效果会更差. LSTM 比 RNN 更好，因为设计有更复杂的记忆模型，但仍然难以记住过去的句子. 而对于 MN 来说，$k=2$ 的实验结果表明使用两个得分最高的记忆单元的索引值来计算问题的回答能够提高模型预测答案的准确度.

2.5　变送器

神经网络中的注意力机制是模拟人类视觉处理过程，其本质是模拟人的视觉以“高分辨率”聚焦在图像的某个区域，同时以“低分辨率”来感知周围不那么重要的图像，而且焦点随着时间的推移进行调整.

对于之前的序列模型，都需要通过包括编码器和解码器的 RNN 或者 CNN 进行处理，性能最好的模型需要通过注意力机制连接编码器和解码器. 对

此，Vaswani 等人提出了完全基于注意力机制的网络结构变送器（Transformer）[5]，无需使用复杂的 RNN 或者 CNN 就能在机器翻译任务中获得更好的性能.

变送器模型使用了编码器-解码器结构，编码器将输入的序列（x_1，…，x_n）映射到连续表示的序列（z_1，…，z_n），然后通过解码器生成预测输出（y_1，…，y_m）. 模型整体结构如图 10 所示.

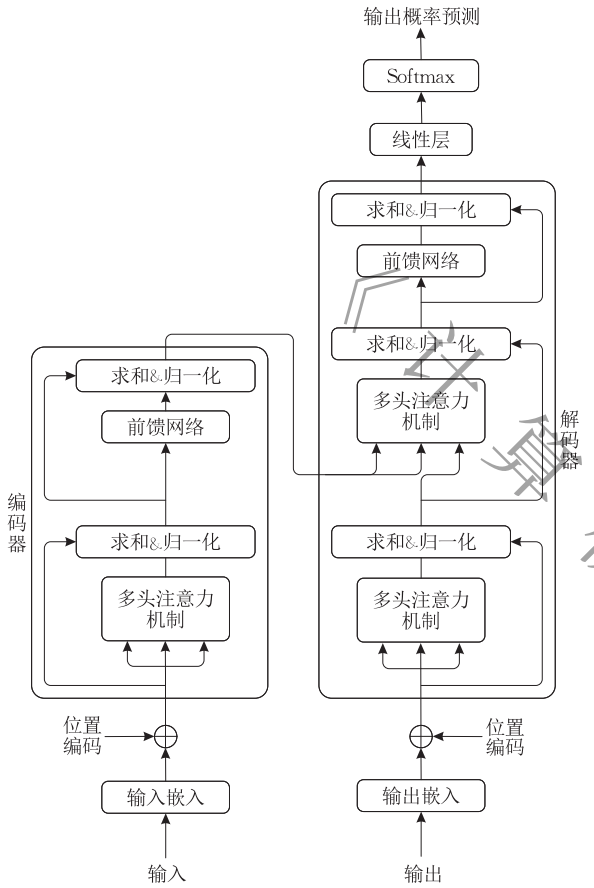

图 10 变送器网络结构示意图

该模型的左半部分表示编码器结构，包含了多头注意力机制（即自注意力机制）和前馈神经网络两个子层. 模型的右半部分表示解码器结构，除了与编码器相同的两子层外，还额外增加了专门处理编码器输出的多头注意力机制（即编码器-解码器注意力机制）. 这两部分的每个子层都使用了残差连接（residual connection），用来解决深度学习中随着深度加深无法训练的问题. 同时，在每个子层的预激活函数后，激活函数前使用批归一化处理.

对于机器翻译任务来说，解码器部分的两个注意力机制分别用来计算输入和输出的权重. 其中编码器-解码器注意力机制表示当前翻译目标语言隐状态向量与编码器隐状态的特征向量之间的关系，而自注意力机制表示当前源语言某个时刻隐状态与源语言句子中其余时刻隐状态之间的关系.

自注意力机制. 对于输入序列的每个单词，通过不同的嵌入向量得到对应的查询向量 $\boldsymbol{q}$、键向量 $\boldsymbol{k}$ 和值向量 $\boldsymbol{v}$，然后计算键向量和查询向量的点积，并除以 $\sqrt{d_k}$，其中 d_k 是键向量的维数，最后使用 Softmax 函数获得权重，计算该单词的自注意力函数：

$$\text{Attention}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V})=\text{Softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^{\mathrm{T}}}{\sqrt{d_k}}\right)\boldsymbol{V} \tag{25}$$

其中 $\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}$ 是句子中每个单词的查询向量 $\boldsymbol{q}$、键向量 $\boldsymbol{k}$ 和值向量 $\boldsymbol{v}$ 组成的矩阵.

对于编码器-解码器注意力机制，其中的查询向量 $\boldsymbol{q}$ 来自解码器隐状态表示，而键向量 $\boldsymbol{k}$ 和值向量 $\boldsymbol{v}$ 来自编码器的输出隐表示.

多头注意力机制. 将 h 个不同的自注意力机制使用不同的映射矩阵进行集成，拼接成一个大的特征矩阵，然后通过全连接层得到输出：

$$\begin{aligned}&\text{MultiHead}(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V})=\text{Concat}(head_1,\cdots,head_h)\boldsymbol{W}^O,\\&head_i=\text{Attention}(\boldsymbol{Q}\boldsymbol{W}_i^{\boldsymbol{Q}},\boldsymbol{K}\boldsymbol{W}_i^{\boldsymbol{K}},\boldsymbol{V}\boldsymbol{W}_i^{\boldsymbol{V}})\end{aligned} \tag{26}$$

其中 $\boldsymbol{W}^O,\boldsymbol{W}_i^{\boldsymbol{Q}},\boldsymbol{W}_i^{\boldsymbol{K}},\boldsymbol{W}_i^{\boldsymbol{V}}$ 是不同的映射矩阵.

位置编码（Position Embedding）. 由于变送器抛弃了序列结构的 RNN，因此没有捕捉顺序序列的能力，也就是说，如果输入句子的单词顺序改变，模型也会得到同样的结果，这显然是不合理的. 因此，为了区分不同位置的单词，在编码词向量的时候使用不同频率的正弦和余弦函数，加入了位置编码的特征：

$$\begin{aligned}PE_{(pos,2i)}&=\sin(pos/10\,000^{2i/d_{\text{model}}})\\PE_{(pos,2i+1)}&=\cos(pos/10\,000^{2i/d_{\text{model}}})\end{aligned} \tag{27}$$

其中，pos 是单词的位置，i 表示单词的维数，d_{model} 是位置编码特征向量的长度. 这样位置编码的每个维度对应正弦曲线的值，而且由于正余弦函数的特性，位置 $pos+k$ 的位置向量可以用 pos 的位置向量线性表示出来：

$$\begin{aligned}\sin(pos+k)&=\sin(pos)\cos(k)+\cos(pos)\sin(k)\\\cos(pos+k)&=\cos(pos)\cos(k)-\sin(pos)\sin(k)\end{aligned} \tag{28}$$

这样除了单词的绝对位置，单词之间的相对位置关系可以通过位置向量捕获到.

在标准的 WMT 2014 英语-德语数据集和 WMT 2014 英语-法语数据集对变送器模型进行了评价. 模型使用 Adam 优化器进行训练[10]，其参数设置为 $\beta_1=0.9$，$\beta_2=0.98$ 以及 $\varepsilon=10^{-9}$，并根据下面的公式在训练过程中改变学习率：

$$\begin{aligned}l_{\text{rate}}=d_{\text{model}}^{-0.5}\cdot\min(&step\_num^{-0.5},\\&step\_num\cdot warmup\_steps^{-1.5})\end{aligned} \tag{29}$$

该公式线性增加第一个 *warmup_steps* 训练步的学习率，之后与步数的平方根成反比地降低学习率，并设置 *warmup_steps* 为 4000.

模型还在每个子层的输出端使用了残差网络和 Dropout 方法[11]，并使用了层归一化技术. 对基本模型使用 $P_{drop}=0.1$. 模型在训练过程中使用标签平滑技术[12]，其参数为 $\varepsilon_{ls}=0.1$.

变送器模型选择了以下 5 个语言模型进行对比实验.

(1) ByteNet 模型. 在 WMT 2014 英语-德语数据集上进行实验. 由于德语的句子往往比英语长，因此设置模型的输出序列长度上限为输入的 1.2 倍. 该模型还使用了空洞卷积(Dilated Convolution)来增加目标网络的接受域，其网络的扩张速率从 1 开始，每层增加一倍，最终达到最大速率 16.

(2) Deep-Att＋PosUnk 模型. 在 WMT 2014 英语-法语数据集上进行实验. 对于源语言和目标语言都使用 256 维的单词嵌入维数，所有的 LSTM 层都具有 512 个存储单元，输出层大小与目标词汇表的大小相等.

(3) GNMT＋RL 模型. 在 WMT 2014 英语-法语和 WMT 2014 英语-德语两个数据集上进行实验. 模型由 8 个编码器层和 8 个解码器层组成，每个编码器和解码器层使用 1024 个 LSTM 节点. 其中注意力网络是一个简单的前馈网络，是一个具有 1024 个节点的隐层.

(4) ConvS2S 模型. 在 WMT 2014 英语-法语和 WMT 2014 英语-德语两个数据集上进行实验. 模型的编码器和解码器使用了 512 个隐单元，而且所有的嵌入维数为 512. 该模型在训练过程使用加速梯度的方法，设置动量值为 0.99，并在梯度的范数超过 0.1 的情况下将其归一化，学习率设置为 0.25.

(5) MoE 模型. 在 WMT 2014 英语-法语和 WMT 2014 英语-德语两个数据集上进行实验. 该模型基于 GNMT 模型进行改进，为了减少计算复杂性，将编码器和解码器中的 LSTM 层数分别设置为 3 和 2，并在编码器和解码器的最后一层前面插入了 MoE 层. 每个 MoE 层最多包含 2048 个专家，每个专家具有约 200 万个参数.

实验结果将获得的 BLEU 分数和训练费用与介绍的对比模型进行比较，如表 3 所示，可以看出变送器模型在翻译任务上的效果以及所耗费的训练成本都是最优的.

基于变送器模型，研究人员在此基础上进行了研究和改进，他们的研究成果总结如表 4 所示.

表 3　变送器与其他模型的实验结果对比

模型	BLEU 分数		训练费用(浮点运算)	
	EN-DE	EN-FR	EN-DE	EN-FR
ByteNet[13]	23.75			
Deep-Att＋PosUnk[14]		39.20		$1.0\cdot10^{20}$
GNMT＋RL[15]	24.60	39.92	$2.3\cdot10^{19}$	$1.4\cdot10^{20}$
ConvS2S[16]	25.16	40.46	$9.6\cdot10^{18}$	$1.5\cdot10^{20}$
MoE[17]	26.03	40.56	$2.0\cdot10^{19}$	$1.2\cdot10^{20}$
Deep-Att＋PosUnk Ensemble[14]		40.40		$8.0\cdot10^{20}$
GNMT＋RL Ensemble[15]	26.30	41.16	$1.8\cdot10^{20}$	$1.1\cdot10^{21}$
ConvS2S Ensemble[16]	26.36	**41.29**	$7.7\cdot10^{19}$	$1.2\cdot10^{21}$
Transformer (base model)	27.30	38.10	**$3.3\cdot10^{18}$**	
Transformer (big)	**28.40**	**41.00**	$2.3\cdot10^{19}$	

表 4　基于变送器的记忆网络模型

模型	时间	核心思想	作用	应用
DTN[18]	2017 年	从数据中学习局部图片的形状和大小	恢复空间对应关系	图像分割
WTN[19]	2017 年	使用多个自注意力分支替代多头注意力	训练更快、性能更好	机器翻译
AAN[20]	2018 年	平均注意力机制	加速解码器	机器翻译
BlendCNN[21]	2018 年	变送器作为教学模型训练 CNN	大规模并行计算并减少内存占用	文本分类
Action Transformer[22]	2018 年	使用变送器来聚合行为人的特征	对时空背景下的行为人进行分类	视频行为识别
Universal Transformers[23]	2018 年	使用循环结构的变送器网络	弥补了变送器不能捕捉序列依赖关系的缺点	语言理解任务
Evolved Transformer[24]	2019 年	神经网络结构搜索	改进前馈序列模型	机器翻译和语言建模
Set Transformer[25]	2019 年	使用注意力机制来编码和聚合特征	模拟输入集合元素之间的复杂交互	涉及数据点集的多个任务
Transformer-XL[26]	2019 年	在自注意力机制引入循环概念和新的位置编码方法	对长期大范围依赖关系建模	语言建模

2.6 小结与分析

本节主要详细介绍了记忆网络发展过程中最重要,也是最基础的模型:RNN、LSTM、NTM、MN 和变送器,以及它们的模型结构和实现过程.通过这几种模型的对比分析,如表 5 所示,可以看出:

(1)RNN 是处理序列模型的神经网络,通过学习序列前后之间的依赖关系来保留部分过去的信息,具有很差的记忆功能.

(2)LSTM 是在 RNN 的基础上对内部隐状态节点的修改,增加了遗忘门和记忆门,从而能够捕获到更有用的记忆信息.

(3)NTM 同样是在 RNN 基础上进行的改进,借用了图灵机的思想,使用神经网络来实现图灵机计算模型中的读写操作,并使用了外部存储矩阵来增强记忆.

(4)MN 与 NTM 都使用了外部存储器来实现模型的记忆功能,但 NTM 的模型结构固化,只能学习较为简单的算法,而 MN 具有更好的变种和泛化能力.

(5)变送器抛弃了 RNN 的基础结构,仅使用注意力机制和编码器-解码器结构,通过构造整个输入序列的隐状态序列与输出序列当前单个输出单词的对齐权,计算当前目标语言输出单词对应的输入源语言序列上下文向量,用上下文向量来预测当前目标语言输出单词的条件后验概率,从而完成机器翻译任务.这里的编码器-解码器注意力机制,与 MN 中的打分函数有着相似的作用,都是用来评估输入序列与候选输出之间的关系.二者最大的不同在于,变送器中本身没有处理序列结构的能力,也使模型丧失了捕捉局部特征的能力,但却能够并行计算,有利于现有的 GPU 训练环境.而 MN 中的输入向量按照序列保存在存储器中,在回答问题时找出与问题最相关的单词.

因此,我们将上述具有记忆功能的神经网络模型统称为记忆网络.在下一节,将在此基础上进行扩展的各种记忆网络模型和应用进行分类介绍.

表 5 记忆网络基础模型对比

基础模型	时间	记忆方式	优点	缺点
循环神经网络(RNN)	1986 年	前后节点传递信息	可以表现时间序列的动态时间行为	只能学习短时间间隔的记忆
长短期记忆网络(LSTM)	1997 年	遗忘门和记忆门	两个门可以控制信息的传递	对于长距离大量记忆有局限性
神经图灵机(NTM)	2014 年	外部存储器	使用外部存储矩阵来增强记忆	只能学习比较简单的算法
记忆网络(MN)	2014 年	外部存储器	具有很好的变种和泛化能力	应用场景单一
变送器(Transformer)	2017 年	注意力机制	模块化,并行计算适应 GPU 环境	无法捕捉局部特征和位置信息

3 记忆网络模型扩展

上一节介绍了记忆网络的基础模型,包括了 RNN、LSTM、NTM、MN 和变送器.除此之外,还有很多学者在这些基础模型上进行了扩展和改进,其中主流的记忆网络有基于 RNN 的扩展模型、基于 LSTM 的扩展模型和基于 MN 的扩展模型.

3.1 基于 RNN 的扩展模型

基于 RNN 的扩展模型有很多,最典型的基于模型结构改进的扩展模型有基于注意力机制的记忆选择 RNN、具有长期记忆的 RNN 和用于单样本学习的 RNN.基于 RNN 的应用类扩展模型太过繁杂,这一部分就不再赘述.

3.1.1 基于注意力机制的记忆选择 RNN

由于 RNN 的记忆单元具有固定的大小,不能存储句子中的所有信息,在预测时容易忽略有用的长期依赖信息.而基于注意力机制的记忆选择 RNN(Attention-based Memory Selection Recurrent Network,AMSRN)[27],通过注意力机制在存储器中查找记忆相关信息的时间,计算注意力权重并提取信息.

AMSRN 模型主要由传统 LSTM 和注意力机制模块两部分组成.LSTM 读取输入单词序列,并存储每个时间步生成的隐层输出.注意力机制模块将存储的隐层输出信息作为输入,生成与下一个单词的条件后验概率分布.

(1)LSTM.LSTM 的输入是单词序列$\{x_1,x_2,\cdots,x_t,\cdots\}$,每个 x_t 由 N 位编码(1-of-N)独热表示.每个时间步 LSTM 的隐层输出是一个 d 维向量 $\boldsymbol{h}_t\in R^d$,因此 t 时刻 LSTM 计算得到的信息是 $M_t=[h_0,h_1,\cdots,h_{t-1}]$.

(2)注意力机制.记忆选择模块从当前隐状态 $\boldsymbol{h}_t$ 生成两个 d 维向量 $\boldsymbol{w}_{h1}$ 和 $\boldsymbol{w}_{h2}$,用于选择 LSTM 存储信息 $M_t=[h_0,h_1,\cdots,h_{t-1}]$.并从当前隐状态 $\boldsymbol{h}_t$ 生成 d 维向量 $\boldsymbol{k}_t$ 作为注意力机制的键:

$$\boldsymbol{k}_t=\boldsymbol{W}_{kh}\boldsymbol{h}_t+\boldsymbol{b}_k \tag{30}$$

其中,矩阵 $\boldsymbol{W}_{kh}\in\mathbb{R}^{d\times d}$ 和向量 $\boldsymbol{b}_k\in\mathbb{R}^d$ 是注意力机制模型需要学习的参数.

然后计算键 $\boldsymbol{k}_t$ 与 LSTM 存储信息 $M_t=[h_0,$

$h_1,\cdots,h_{t-1}]$中每个 h_i的内积相似度 e_{ti}：

$$e_{ti}=(h_i\circ \boldsymbol{w}_{h1})\cdot \boldsymbol{k}_t \tag{31}$$

其中“∘”表示逐元素乘法，“·”表示内积. 通过 $h_i\circ \boldsymbol{w}_{h1}$对 LSTM 存储信息 $M_t=[h_0,h_1,\cdots,h_{t-1}]$每个元素的每一维用当前隐状态 $\boldsymbol{h}_t$对应的 $\boldsymbol{w}_{h1}$的每一维进行比例变换，然后计算经过调整后的多个时刻的隐状态 $h_i\circ \boldsymbol{w}_{h1}$与键 $\boldsymbol{k}_t$的相似度.

然后用 Softmax 函数将相似度 e_{ti}归一化得到注意力权重 α_{ti}：

$$\alpha_{ti}=\frac{\exp(e_{ti})}{\sum_{i=0}^{t-1}\exp(e_{ti})} \tag{32}$$

注意力机制也用 $\boldsymbol{w}_{h2}$对 h_i进行选择得到 h_i'，式(72)用于确定 h_i的每个分量与相关向量 $\boldsymbol{r}_t$的关联程度. 相关向量 $\boldsymbol{r}_t$通过注意力权重 α_{ti}与 h_i'的加权和得到：

$$h_i'=h_i\circ \boldsymbol{w}_{h2} \tag{33}$$

$$\boldsymbol{r}_t=\sum_{i=0}^{t-1}\alpha_{ti}h_i' \tag{34}$$

最后，用注意力相关向量 $\boldsymbol{r}_t$和隐状态 $\boldsymbol{h}_t$来预测下一个单词的条件概率分布：

$$P_w=\mathrm{Softmax}(W_{ph}\boldsymbol{h}_t+W_{pr}\boldsymbol{r}_t+b_p) \tag{35}$$

其中 W_{ph}，W_{pr}，b_p都是模型需要学习的参数. 在训练过程中，目标损失函数选用预测的单词条件概率分布 P_w和训练集中所有单词的参考分布之间的交叉熵函数.

AMSRN 模型在两个英文数据集和一个中文数据集上进行实验，第一个英文数据集 Penn Treebank Corpus[28]是一种广泛使用的数据集，用来评估语言模型的有效性. 第二个英文数据集 Switchboard Corpus[29]是一个电话语音语料库，收集了美国电话用户之间的双向对话内容. 还有一个中文数据集 Gigaword[30]包含了 25 K 的中文新闻文章. 表 6 总结了这三个数据集在实验中的统计信息，并在测试集上使用困惑度对模型进行评价.

表 7 显示了不同模型在不同数据集上的实验结果. 实验是逐步进行的，首先训练典型的 LSTM 语言模型，然后在 LSTM 中添加注意力机制模块，可以发现注意力机制模块对 Penn Treebank 和 Switchboard 两个数据集都有帮助，但不能改善在 Gigaword 上的实验效果. 而且记忆选择对于注意力机制至关重要，能够极大改进模型的性能.

表 6　AMSRN 模型使用的数据集统计

数据集	语言	训练集大小/K	验证集大小/K	测试集大小/K	每句的平均单词数量	字典大小
Penn Treebank	英语	40	3	4.0	21.10	9999
Switchboard	英语	945	10	5.2	10.39	47238
Gigaword	汉语	531	165	260.0	10.79	5123

表 7　AMSRN 模型困惑度对比实验结果

模型	Penn Treebank	Switchboard	Gigaword
LSTM	143.31	93.90	94.03
LSTM+注意力机制	134.09	93.74	102.04
LSTM+注意力机制+记忆选择机制	133.36	92.49	86.85
RMN	123.32	64.41	121.28
RMR	134.30	71.04	145.24

除此之外，还将提出的模型与另外两个基于注意力机制的语言模型进行比较，即 RMN(Recurrent Memory Network)和 RMR(Recurrent Memory Recurrent)[31]. AMSRN 在两个英语数据集上的效果不如 RMN 和 RMR，但是在中文数据集上却有很好的效果. 这可能是 RMN 和 RMR 仅在英文数据集上得到验证，而对于中文数据集需要考虑特别的技巧. 总的来说，提出的 AMSRN 在不同数据集上的效果有一定的提升，但是改进有限.

3.1.2　具有长期记忆的 RNN

Mikolov 等人对简单循环神经网络结构进行修改[32]，提出了一种具有上下文特征的 RNN，使循环权重矩阵能够缓慢改变隐单元的状态，形成一种长期记忆，具有与复杂的 LSTM 相近的性能，其结构如图 11 所示.

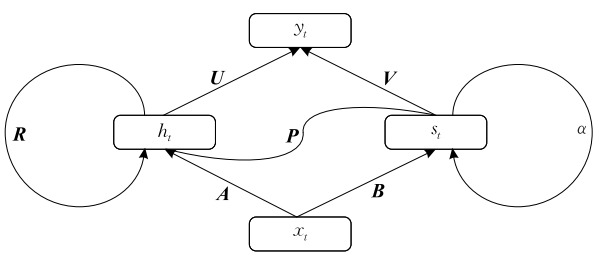

图 11　具有上下文特征的 RNN

对于简单循环神经网络，隐状态 $\boldsymbol{h}_t$ 是基于前一时刻的 h_{t-1} 和当前输入 x_t 进行更新：

$$\boldsymbol{h}_t=\sigma(\boldsymbol{A}x_t+\boldsymbol{R}h_{t-1}) \tag{36}$$

具有上下文特征的 RNN，增加了一个用于获取长期依赖关系的隐层 s_t：

$$\begin{aligned} s_t&=(1-\alpha)\boldsymbol{B}x_t+\alpha s_{t-1}\\ \boldsymbol{h}_t&=\sigma(\boldsymbol{P}s_t+\boldsymbol{A}x_t+\boldsymbol{R}h_{t-1})\\ y_t&=f(\boldsymbol{U}\boldsymbol{h}_t+\boldsymbol{V}s_t) \end{aligned} \tag{37}$$

这里，$\boldsymbol{A},\boldsymbol{R},\boldsymbol{B},\boldsymbol{P},\boldsymbol{U},\boldsymbol{V}$ 是适当维数的连接边权值矩阵，$0\leqslant\alpha\leqslant1$. 为了获取不同时间延迟上的上下文的关系，可以使用元素均为 0 或 1 的对角阵 $\boldsymbol{Q}$ 来代替 α，选择当前时刻 t 之前的 k 个时刻的长期依赖关系的隐层 s_k，$k=t-1,\cdots,t-K$.

该模型在 Penn Treebank Corpus 和 Text8 数据集上进行实验，并与标准 LSTM 和 RNN 比较. 实验结果表明，该模型可以在具有相对较少参数的小型数据集上实现与 LSTM 相当的性能.

3.1.3 用于单样本学习的 RNN

单样本学习打破了传统神经网络需要大量迭代训练才能进行学习的传统，当模型有新的数据输入时，使用记忆增强的神经网络可以充分利用新的数据，同时不会对现有模型造成干扰，在几个样本训练之后就能做出准确的预测[33].

(1) 元学习方法

对于一般的学习任务，通常使学习的损失函数 L 最小化来选择模型参数 θ. 而对于元学习分类任务，如图 12 所示，需要选择能够使得损失函数 $L(D;\theta)$ 在数据 $D=\{x_i,y_i\}_{i=1}^{t}$ 的概率分布 $p(D)$ 上的数学期望最小的参数：

$$\theta^*=\arg\min_{\theta}E_{D:p(D)}[L(D;\theta)] \tag{38}$$

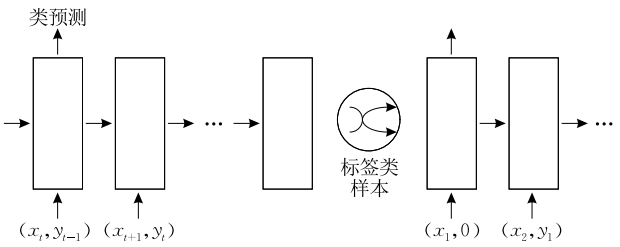

图 12 元学习分类任务示意图

在 t 时刻，模型的输入是先前的数据样本的正确标签 y_{t-1} 和新的问题 x_t. 这样的元学习模型将数据表示与标签联系起来，并采用通用的方法将这种关系表示映射到合适的类或者用于预测的函数值上.

(2) 记忆增强模型

选择神经图灵机作为记忆增强模型，因为它可以通过缓慢更新权重进行长期记忆，并使用外部存储器模块进行短期记忆，是元学习和少样本预测的理想选择. 控制器选择 LSTM 或前馈网络，使用读写头与外部存储器模块交互.

但是，传统的神经图灵机对于独立序列的编码任务不是最佳的，因此在写入存储器时需要使用新设计的访问模块，即近期最少使用访问模块(Least Recently Used Access，LRUA). 这个模块使用纯基于内容的存储写入，把需要存储的信息写入最少使用或者最近使用的存储器位置. LRUA 注重最近相关信息的准确编码，以及纯基于内容的索引. 模块写入新的信息到很少使用的位置，保留最近编码的信息，或者写入到最后使用的位置，用最可能相关的新的信息作为存储器的更新. 这两种写入方法通过在先前读写权重和使用权重 w_t^u 的比例变换之间插值来实现. 在每个时刻通过衰减先前的使用权重 w_{t-1}^u 并添加当前的读 w_t^r 和写 $\boldsymbol{w}_t^w$ 来更新使用权重 w_t^u：

$$w_t^u\leftarrow\gamma w_{t-1}^u+w_t^r+\boldsymbol{w}_t^w \tag{39}$$

其中 γ 是一个衰减因子，w_t^r 通过归一化控制器的键 $\boldsymbol{k}_t$ 和存储器矩阵 $\boldsymbol{M}_t$ 的余弦相似度得到：

$$K(\boldsymbol{k}_t,\boldsymbol{M}_t(i))=\frac{\boldsymbol{k}_t\cdot\boldsymbol{M}_t(i)}{\|\boldsymbol{k}_t\|\|\boldsymbol{M}_t(i)\|} \tag{40}$$

$$w_t^r(i)\leftarrow\frac{\exp(K(\boldsymbol{k}_t,\boldsymbol{M}_t(i)))}{\sum_j\exp(K(\boldsymbol{k}_t,\boldsymbol{M}_t(i)))} \tag{41}$$

使用 sigmoid 函数计算先前读取权重 w_{t-1}^r 和最少使用权重 w_{t-1}^{lu} 的凸组合，得到写入权重：

$$\boldsymbol{w}_t^w\leftarrow\sigma(\alpha)w_{t-1}^r+(1-\sigma(\alpha))w_{t-1}^{lu} \tag{42}$$

其中 α 是在权重之间插值的标量参数. 最少使用权重 w_{t-1}^{lu} 是一个位置指示函数，当 $w_t^u(i)$ 小于向量 $\boldsymbol{w}_t^w$ 中的最小元素时，设为 1，否则设为 0.

最后用写入权重和控制器的键 $\boldsymbol{k}_t$ 来更新存储器：

$$\boldsymbol{M}_t(i)\leftarrow\boldsymbol{M}_{t-1}(i)+\boldsymbol{w}_t^w(i)\boldsymbol{k}_t \tag{43}$$

因此，记忆增强模型将记忆写入设为零或者先前使用的存储器中，其中使用最少的记忆会被覆盖.

因为记忆是 RNN 学习中最重要的机制之一，使用良好的记忆机制，长期的历史信息可以与当前信息融合，从而改善 RNN 的学习. Wang 等人提出了一种新的 RNN 记忆机制[34]，使用一种特殊的辅助记忆单元，明确地分离了存储器和输出的计算过程，并采用错误流截断技术开发了一种有效的学习算法. 该方法克服了学习存储器和输出之间的冲突问题和梯度消失问题，可以在很长的时间范围内学

习和保持稳定的记忆能力.

除了上述几种方法以外，还有梯度截断和信息流正则化[35]的方法来优化 RNN 模型，以便学习长期依赖关系.

梯度截断. 当目标函数的参数梯度特别大的时候，基于梯度下降的神经网络的连接边权值矩阵和偏置参数更新很大，偏离当前解很远. 因此神经网络的连接边权值矩阵和偏置参数更新量必须足够小，以避免偏离当前解过分远.

梯度截断有两种方法，一种选择是在参数更新之前，逐元素地截断小批次更新产生的参数梯度. 另一种是在参数更新之前，截断梯度 g 的范数 $\|g\|$，即当 $\|g\|>v$ 的时候：

$$g \leftarrow \frac{gv}{\|g\|} \tag{44}$$

其中 v 是范数上界，截断梯度 g 用来更新参数. 因为所有参数(包括不同的参数组，如权重和偏置)的梯度被单个缩放因子联合重整化，所以后一种方法具有的优点是保证了每个参数更新步骤仍然是沿着梯度方向更新的. 虽然参数更新与真实梯度具有相同的方向梯度，经过梯度范数截断，更新的参数的向量范数现在变得有界. 这种有界梯度能避免梯度爆炸问题.

传统的随机梯度下降方法得到的结果为梯度的无偏估计，而使用范数截断的梯度下降方法引入了基于经验的启发式截断偏置设置方法. 通过逐元素截断，更新的方向与真实梯度或小批量训练过程的梯度不再具有无偏估计关系，但仍然是下降方向.

信息流正则化. 梯度截断有助于避免梯度爆炸，但它无助于处理梯度消失问题. 为了解决梯度消失问题并使得学习的模型更好地捕获长期序列依赖关系，可以在模型中引入参数正则化项或参数约束，以迫使损失函数的梯度按照我们的要求改变. 即使在损失函数中，只对学习序列的尾部对应的隐向量引入正则化项，也希望梯度向量 $\nabla_{h^{(t)}} L$ 在反向传播的时候能保持其范数大小不过分上下波动，因此 Pascanu 等人[36]提出了在损失函数中引入正则化项：

$$\Omega=\sum_t \left(\frac{\left\| (\nabla_{h^{(t)}} L)\frac{\partial h^{(t)}}{\partial h^{(t-1)}} \right\|}{\nabla_{h^{(t)}} L} - 1 \right)^2 \tag{45}$$

使用该正则化项与标准的启发式梯度截断方法(处理梯度爆炸)相结合，可以显著增加 RNN 学习序列依赖关系的长度范围.

3.2　基于 LSTM 的扩展模型

基于 LSTM 的扩展模型主要包括了两个改进方向，分别是基于模型结构改进的 LSTM，以及基于应用而改进的 LSTM.

前者包括了关联 LSTM 和树 LSTM 两个基于 LSTM 结构改进的模型. 后者包括了 LSTM 在用于机器阅读和单样本学习的两个应用方向的改进模型.

3.2.1　关联 LSTM

LSTM 被广泛应用于序列预测、语音识别和机器翻译中，但 LSTM 仍存在两个缺点. 一个是记忆单元数量与递归权重矩阵包含的元素个数相关，具有 N_h 个记忆单元的 LSTM 需要递归权重矩阵包含的元素个数是 $O(N_h^2)$. 另一个是 LSTM 缺少读写时对存储器进行索引的机制，不能很好地表示数据结构. 因此 Danihelka 等人提出了关联长短期记忆网络(Associative Long Short-Term Memory，ALSTM)[37]，将 LSTM 与全息缩减表示(Holographic Reduced Representations，HRR)相结合，利用键-值对进行存储，使用冗余存储增加存储器的容量，并减少检索过程中的噪声，在面对多个记忆任务时有更快的学习速度.

HRR 是一种用固定长度的向量来表示键-值对的关联数组的方法. 每个独立的键值对和整个关联数组的大小相同，数组由所有对的和进行表示. 假设一个复数向量键 $r=(a_r[1]e^{i\phi_r[1]}, a_r[2]e^{i\phi_r[2]}, \cdots)$，该键-值对通过逐元素复数乘法绑定在一起：

$$y=r\otimes x= (a_r[1]a_x[1]e^{i(\phi_r[1]+\phi_x[1])}, a_r[2]a_x[2]e^{i(\phi_r[2]+\phi_x[2])}, \cdots) \tag{46}$$

给定键 r_1, r_2, r_3 和输入向量 $\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{x}_3$，关联数组为 $c=r_1\otimes\boldsymbol{x}_1+r_2\otimes\boldsymbol{x}_2+r_3\otimes\boldsymbol{x}_3$，把 c 称为记忆迹(memory trace). 为了检索与键 r_k 相关的项，将 r_k^{-1} 与记忆迹 c 逐个相乘，就得到了 $\boldsymbol{x}_k$ 的噪声项. 例如 $\boldsymbol{x}_2$ 的噪声项计算方式为

$$\begin{aligned} r_2^{-1}\otimes c &= r_2^{-1}\otimes(r_1\otimes\boldsymbol{x}_1+r_2\otimes\boldsymbol{x}_2+r_3\otimes\boldsymbol{x}_3) \\ &= \boldsymbol{x}_2+r_2^{-1}\otimes(r_1\otimes\boldsymbol{x}_1+r_3\otimes\boldsymbol{x}_3) \\ &= \boldsymbol{x}_2+noise \end{aligned} \tag{47}$$

模型将 HRR 与 LSTM 结合，需要计算复数向量乘法. 对于复向量 $\boldsymbol{z}=h_{\text{real}}+ih_{\text{imaginary}}$，其形式为

$$h=\begin{bmatrix} h_{\text{real}} \\ h_{\text{imaginary}} \end{bmatrix} \tag{48}$$

其中 $h\in\mathbb{R}^{N_h}$，$h_{\text{real}}, h_{\text{imaginary}}\in\mathbb{R}^{N_h/2}$.

首先，使用 LSTM 中的方法计算门控变量，并同时定义关联键的参数 $\hat{r}_i, \hat{r}_o$，实部和虚部使用相

同的门：

$$\hat{g}_f,\hat{g}_i,\hat{g}_o,\hat{r}_i,\hat{r}_o=W_{xh}x_t+W_{hh}h_{t-1}+b_h \tag{49}$$

$$\hat{u}=W_{xu}x_t+W_{hu}h_{t-1}+b_u \tag{50}$$

定义一个上界函数，将复数的实部和虚部的模限制在 0 和 1 之间：

$$\text{bound}(h)=\begin{bmatrix} h_{\text{real}}/d \\ h_{\text{imaginary}}/d \end{bmatrix} \tag{51}$$

$$d=\max(1,\sqrt{h_{\text{real}}\circ h_{\text{real}}+h_{\text{imaginary}}\circ h_{\text{imaginary}}})\in\mathbb{R}^{N_h/2} \tag{52}$$

其中“/”表示对应元素除法，“∘”表示对应元素乘法. d 表示当模大于 1 时，对每个复数的模进行归一化. 然后使用上界函数来得到更新函数 u、输入键函数 r_i 和输出键函数 r_o 的值：

$$\begin{aligned} u&=\text{bound}(\hat{u}) \\ r_i&=\text{bound}(r_i) \\ r_o&=\text{bound}(r_o) \end{aligned} \tag{53}$$

其中 $r_i\in\mathbb{R}^{N_h}$ 是输入键，作为关联数组中的存储键，$r_o\in\mathbb{R}^{N_h}$ 是输出键，对应查找键. 更新函数 u 和输入门 g_i 相乘得到要存储的值.

为了减少检索噪声，增加了具有随机排列的冗余副本. 对于每个副本通过 $s=\{1,\cdots,N_{\text{copies}}\}$ 进行索引，并将相同的键-值对添加到复位单元和记忆单元上：

$$r_{i,s}=\begin{bmatrix} \boldsymbol{P}_s & 0 \\ 0 & \boldsymbol{P}_s \end{bmatrix}r_i \tag{54}$$

$$c_{s,t}=g_f\circ c_{s,t-1}+r_{i,s}\otimes(g_i\circ u) \tag{55}$$

其中，$r_{i,s}$ 是变换后的输入键，$\boldsymbol{P}_s\in\mathbb{R}^{N_h/2\times N_h/2}$ 是一个取值为常数的随机排列矩阵，对应于第 s 个副本. “⊗”表示元素复数乘法，定义为

$$r\otimes u=\begin{bmatrix} r_{\text{real}}\circ u_{\text{real}}-r_{\text{imaginary}}\circ u_{\text{imaginary}} \\ r_{\text{real}}\circ u_{\text{imaginary}}+r_{\text{imaginary}}\circ u_{\text{real}} \end{bmatrix} \tag{56}$$

而每个副本的输出键 $r_{o,s}$ 与输入键 $r_{i,s}$ 格式相同：

$$r_{o,s}=\begin{bmatrix} \boldsymbol{P}_s & 0 \\ 0 & \boldsymbol{P}_s \end{bmatrix}r_o \tag{57}$$

最后通过对每个副本求均值，得到记忆迹单元的隐状态表示：

$$\boldsymbol{h}_t=g_o\circ\text{bound}\left(\frac{1}{N_{\text{copies}}}\sum_{s=1}^{N_{\text{copies}}}r_{o,s}\otimes c_{s,t}\right) \tag{58}$$

3.2.2 树状 LSTM

Zhang 等人提出的树状长短期记忆网络[38]（Tree Long Short-Term Memory Networks，TLSTM），基于 LSTM 预测树状序列数据，通过估计依赖树的生成概率来定义句子单词出现的概率. 模型基于生成子树的表示，生成依赖树节点，还能够通过明确表示左右依赖之间的相关性，进一步增强 TLSTM 的建模能力.

依赖路径的定义. 依赖路径 $D(w)$ 是树根和当前节点 w 之间的路径，由路径上的节点和连接它们的边组成. w_0 表示树中的节点，$w_1,w_2,\cdots,w_n$ 表示其左依赖. 令 LEFT 表示 w_0 和第一个左依赖之间的边. w_k 表示 w_0 的次左依赖，从 w_{k-1} 到 w_k 的边记为 NX-LEFT，如图 13 所示的左依赖路径. 并用类似的方法定义右依赖路径 RIGHT 和 NX-RIGHT.

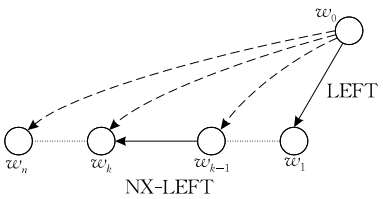

图 13 左依赖路径示意图

依赖树概率的定义. 语言建模问题的核心是，在给定输入树 T 和条件概率 $P(S|T)$ 的情况下，计算输出句子 S 的概率. 把依赖树的概率计算过程看作生成模型的过程，假设依赖树 T 以广度优先（Breadth-First Search，BFS）的方式自上而下构建，从 ROOT 节点开始. 对于同一层的节点，先是左依赖由近到远生成，然后以同样的方式生成右依赖，其模型结构如图 14 所示.

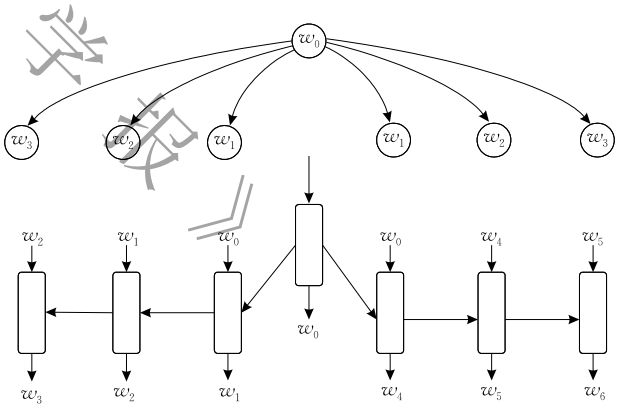

图 14 TLSTM 模型结构示意图

假设依赖树中的单词 w 仅以依赖路径为条件，给定依赖树 T，句子 S 的条件后验概率可表示为

$$P(S|T)=\prod_{w\in BFS(T)\backslash ROOT}P(w|D(w)) \tag{59}$$

模型结构. 依赖路径 $D(w)$ 表示成〈单词，边类型〉元组的序列子树，使用四个 LSTM 进行学习，这四个 LSTM 分别表示四种类型的边（LEFT，NX-LEFT，RIGHT，NX-RIGHT）. 在每个时刻，根据边的类型选择 LSTM，然后将一个单词作为输入并生成其依赖节点或兄弟节点，上述生成过程，可以看作对树添加边和节点的过程.

在时刻 $t\in[1,n]$，令$\langle w_{t'},z_{t'}\rangle$表示 $D(\boldsymbol{w}_t)$中的上一个元组，下标 t 和 t'表示 w 的广度优先搜索顺序，$z_t\in\{\text{LEFT},\text{RIGHT},\text{NX-LEFT},\text{NX-RIGHT}\}$是边的类型. 为了减少参数的数量，令这四个 LSTM 共享相同的嵌入矩阵 $\boldsymbol{W}$ 和隐状态 H，并忽略所有的偏差项. 那么在时刻 t 时，用 $H[:,t]$表示依赖路径 $D(w)$的计算过程为

$$\begin{aligned}x_t&=\boldsymbol{W}_e\cdot e(w_{t'})\\ \boldsymbol{h}_t&=\text{LSTM}^{z_t}(x_t,H[:,t'])\\ H[:,t']&=\boldsymbol{h}_t\\ y_t&=\boldsymbol{W}_{ho}\cdot\boldsymbol{h}_t\end{aligned}\tag{60}$$

其中 $\boldsymbol{W}_e\in\mathbb{R}^{s\times|V|}$ 表示单词嵌入矩阵，$\boldsymbol{W}_{ho}\in\mathbb{R}^{|V|\times d}$ 表示模型的输出矩阵，其中 V 是字典中包含单词的个数，s 是嵌入子空间的维数，d 是隐单元的维数. $H\in\mathbb{R}^{d\times(n+1)}$ 表示共享的隐状态，$e(\boldsymbol{w}_t)$是 $\boldsymbol{w}_t$的独热编码表示. LSTM^{z_t}为深度 LSTM 网络. 在 t 时刻，输入 x_t 和前一时刻的隐状态 $h_{t'}$，输出当前时刻的隐状态 $\boldsymbol{h}_t$.

模型基于边的类型选择 LSTM，那么依赖路径 $D(\boldsymbol{w}_t)$的选择概率通过 Softmax 函数进行计算：

$$P(\boldsymbol{w}_t\mid D(\boldsymbol{w}_t))=\frac{\exp(y_t,\boldsymbol{w}_t)}{\sum_{k'=1}^{|V|}\exp(y_t,k')}\tag{61}$$

依赖树的改进方法. TLSTM 忽略了同一层中左右依赖之间的相互作用，因此在上述模型中增加一个 LSTM，从左依赖的远端到近端，称为左依赖树，如图 15 所示.

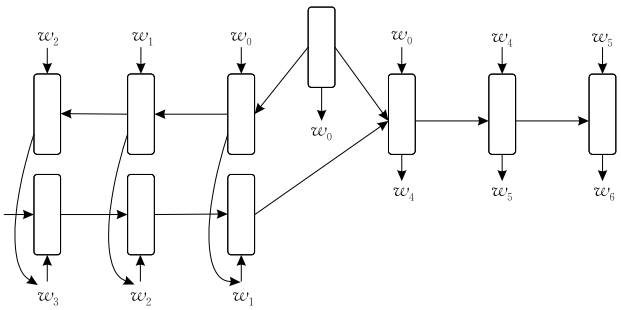

图 15　改进后的依赖树模型示意图

左依赖树学习根节点 w_0的所有左依赖表示，预测根节点下的第一右依赖，其余的右依赖可以通过其隐状态进行学习.

令 v_t为左依赖序列，长度为 K(在图 20 中$K=3$，$v_t=(w_3,w_2,w_1)$)，那么在每个时间步 k 的隐状态 q_k(所有左依赖的表示)为

$$m_k=\boldsymbol{W}_e\cdot e(v_t,k)\tag{62}$$

$$q_k=\text{LSTM}^{L_D}(m_k,q_{k-1})\tag{63}$$

那么当前隐状态计算过程为

$$\boldsymbol{r}_t=\begin{bmatrix}\boldsymbol{W}_e\cdot e(w_{t'})\\ q_k\end{bmatrix}\tag{64}$$

$$\boldsymbol{h}_t=\text{LSTM}^{\text{GEN-R}}(\boldsymbol{r}_t,H[:,t'])\tag{65}$$

其中 q_k作为 RIGHT 边的 LSTM 的附加输入.

具有依赖树的树状 LSTM 在 Microsoft Research 句子完成挑战任务中获得了 60.67%的准确率，要高于 57.02%准确率的 LSTM，以及 RNN 和 skip-gram 组合模型获得的最先进方法(58.9%)，这表明了具有依赖树的树状 LSTM 的性能要优于其他模型.

除了上述基于 LSTM 的扩展模型，Graves 等人[39]结合了 LSTM 和双向 RNN 的优点，得到了双向 LSTM. 双向 LSTM 由一个正向 LSTM 和一个反向 LSTM 组成，其中正向 LSTM 用来学习输入序列的正向依赖关系，而反向 LSTM 用来学习输入序列的反向依赖关系，而且每一时刻的隐状态是由这两个 LSTM 的隐状态拼接而成，分别包含了前向和后向依赖信息.

对于 LSTM 来说，由于按照时间顺序处理输入序列，因此 LSTM 的输出通常是基于前文的内容得到的，很难利用输入序列的后向依赖关系. 所以，将双向结构引入 LSTM，使得 LSTM 能够有效利用上下文的依赖信息，在实验中获得比单向 LSTM 更好的效果.

3.2.3　用于机器阅读的 LSTM

传统的 LSTM 在处理序列文本时具有两个问题：(1)记忆压缩，在记忆压缩过程中哪些部分被记忆是未知的；(2)结构化输入，没有明确的机制推理输入单词之间的相互关系. 针对上述两个问题，文献[40]提出了用于机器阅读问题的长短期记忆网络(Long Short-Term Memory Network，LSTMN).

LSTMN 结构如图 16 所示，其中 x_t表示当前

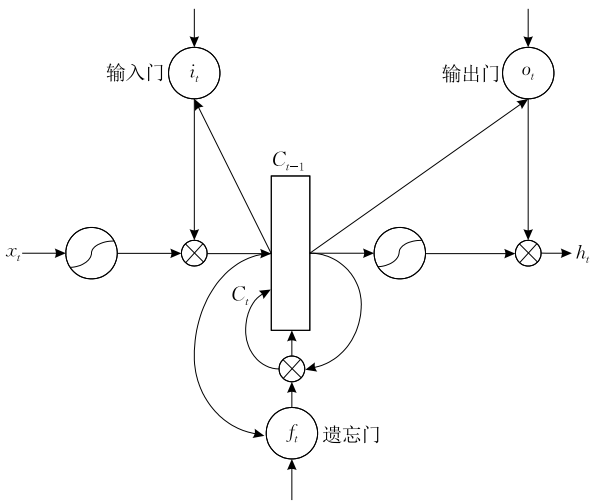

图 16　LSTMN 结构示意图

输入，模型存储着前一时刻隐状态 H_{t-1} 和记忆向量 $\boldsymbol{C}_{t-1}$：

(1) 与环境交互的隐状态 $H_{t-1}=(h_1,\cdots,h_{t-1})$；

(2) 表示实际存储内容的记忆 $\boldsymbol{C}_{t-1}=(c_1,\cdots,c_{t-1})$. 每个输入都与一个隐状态向量和一个记忆向量相关联.

在 t 时刻，通过一个注意力机制层计算 x_t 与之前所有单词的相互关系：

$$a_i^t=\boldsymbol{v}^{\mathrm{T}}\tanh(W_h h_i+W_x x_t+W_{\tilde{h}}\tilde{h}_{t-1}) \tag{66}$$

$$s_i^t=\mathrm{Softmax}(a_i^t) \tag{67}$$

然后计算隐状态向量和记忆向量的自适应求和向量：

$$\begin{bmatrix}\tilde{h}_t\\ \tilde{c}_t\end{bmatrix}=\sum_{i=1}^{t-1}s_i^t\begin{bmatrix}\boldsymbol{h}_t\\ \boldsymbol{c}_t\end{bmatrix} \tag{68}$$

最后更新整个 LSTM 各个门的状态：

$$\begin{bmatrix}i_t\\ f_t\\ o_t\\ \hat{c}_t\end{bmatrix}=\begin{bmatrix}\sigma\\ \sigma\\ \sigma\\ \tanh\end{bmatrix}W\cdot[\tilde{h}_t,x_t] \tag{69}$$

$$\boldsymbol{c}_t=f_t\times\tilde{c}_t+i_t\times\hat{c}_t \tag{70}$$

$$\boldsymbol{h}_t=o_t\times\tanh(\boldsymbol{c}_t) \tag{71}$$

自然语言处理任务涉及到对两个序列进行建模，常用的方法是编码器-解码器架构. 对此文献[40]提出了浅融合模型和深融合模型：

(1) 浅融合模型结构如图 17 所示，用 LSTMN 模块替代标准编码器-解码器中的 LSTM 模块，编码器和解码器都被建模为具有自注意力机制(intra-attention)的 LSTMN. 同时，当解码器读取目标时则使用编码器和解码器隐向量之间的相互注意力机制(inter-attention).

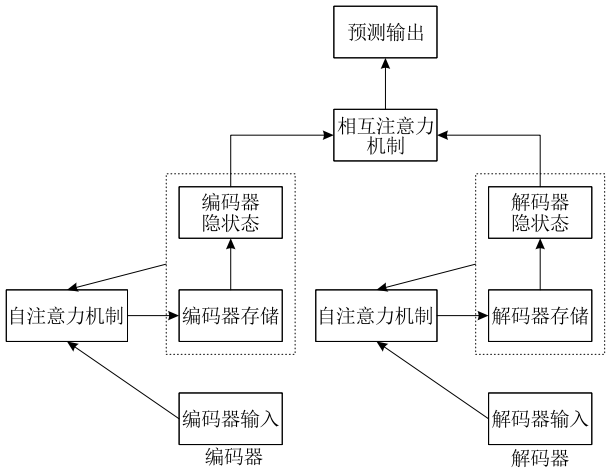

图 17 浅融合模型表示

(2) 深融合模型结构如图 18 所示，将相互注意力机制和自注意力机制结合在一起，由解码器输出更新状态. 假设记忆向量为 $\boldsymbol{A}=[\alpha_1,\cdots,\alpha_m]$，隐状态向量为 $\boldsymbol{Y}=[\gamma_1,\cdots,\gamma_m]$，其中 m 是条件序列的长度. 计算时刻 t 的输入和整个输入序列间的相互注意力机制权：

$$p_j^t=\mathrm{Softmax}(\boldsymbol{u}^{\mathrm{T}}\tanh(W_\gamma\gamma_j+W_x x_t+W_{\tilde{\gamma}}\tilde{\gamma}_{t-1})) \tag{72}$$

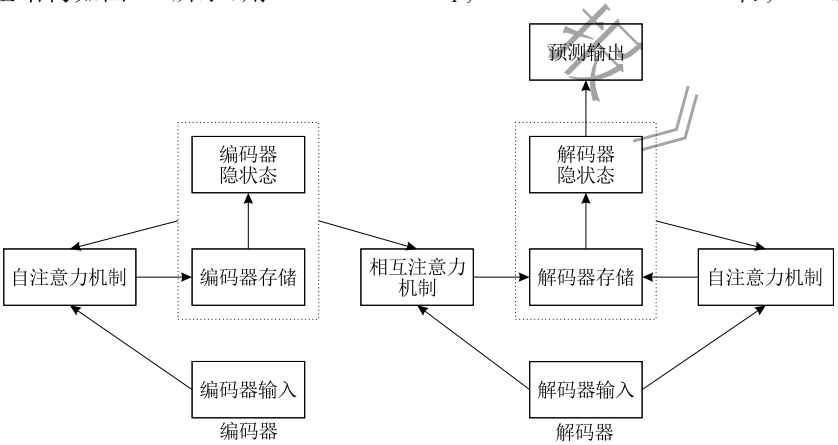

图 18 深融合模型表示

然后计算自适应表示：

$$\begin{bmatrix}\tilde{\gamma}_t\\ \tilde{\alpha}_t\end{bmatrix}=\sum_{j=1}^{m}p_j^t\cdot\begin{bmatrix}\gamma_j\\ \alpha_j\end{bmatrix} \tag{73}$$

然后用一个门函数 $\boldsymbol{r}_t$ 将自适应表示变换到目标存储器：

$$\boldsymbol{r}_t=\sigma(W_r\cdot[\tilde{\gamma}_t,x_t]) \tag{74}$$

那么当前时刻更新后的存储器向量 $\boldsymbol{c}_t$ 和隐状态向量 $\boldsymbol{h}_t$ 为

$$\boldsymbol{c}_t=\boldsymbol{r}_t\times\tilde{\alpha}_t+f_t\times\tilde{c}_t+i_t\times\hat{c}_t \tag{75}$$

$$\boldsymbol{h}_t=o_t\times\tanh(\boldsymbol{c}_t) \tag{76}$$

该模型在 Penn Treebank 数据集进行了语言建模实验，并与 RNN、LSTM 以及多个改进的 LSTM 模型进行对比，都获得了最低的困惑度. 还在 Stanford Sentiment Treebank 数据集进行了情绪分析实验，与基线模型 LSTM 和多个最先进的方法进行对比，实验结果表明，该模型均优于 LSTM 基准模型，并

且与效果最佳的 T-CNN 模型相当.

3.2.4　用于单样本学习的 LSTM 匹配网络

在机器学习研究中，从少量样本中进行学习是一个巨大的挑战. 在视觉、文本的处理上，传统的有监督的深度学习方法不能够从有限的数据中有效地学习新的概念. 受到深度神经网络学习图像、文本特征的启发，通过拓展外部信息加强记忆网络的学习效果，文献[41]提出了针对小样本学习的匹配网络模型.

模型使用非参数方法解决小样本学习的问题. 给定一个具有 k 个样本的图像-标签支持集 $S=\{(x_i,y_i)\}_{i=1}^{k}$，通过学习到的分类器 $c_S(\hat{x})$ 对测试样例 $\hat{x}$ 进行分类，得到 $\hat{x}$ 的类别 $\hat{y}$. 那么 $S\rightarrow c_S(\hat{x})$ 的映射表示为条件后验概率 $P(\hat{y}|\hat{x},S)$，$P(\hat{y}|\hat{x},S)$ 模型参数通过神经网络学习得到，神经网络映射方式为学习到的模型.

(1) 模型算法

输入测试样例 $\hat{x}$，计算输出 $\hat{y}=\sum_{i=1}^{k}a(\hat{x},x_i)y_i$ 的条件后验概率 $P(\hat{y}|\hat{x},S)$，得到该样例所属的类别 $c_S(\hat{x})$. 其中，x_i，y_i 是支持集中的样例和标签，a 是注意力对齐函数，用来度量测试样例 $\hat{x}$ 和训练样例 x_i 的匹配程度. 注意力函数 a 的形式是余弦距离函数 c 的 Softmax 函数，用其它所有参考集样本归一化：

$$a(\hat{x},x_i)=e^{c(f(\hat{x}),g(x_i))}\Big/\sum_{j=1}^{k}e^{c(f(\hat{x}),g(x_i))} \tag{77}$$

其中 f，g 是测试样例和参考样例的特征嵌入表示函数，在图像任务中使用深度卷积神经网络实现，在语言任务中使用单词嵌入表示. c 是余弦距离，用来计算训练样例和测试样例的匹配度.

(2) 对训练集进行编码

对支持集进行编码的函数为 $g(x_i,S)$，其形式为双向 LSTM：$g(x_i,S)=\overrightarrow{h}_i+\overleftarrow{h}_i+g'(x_i)$. 这个双向 LSTM 的输入序列为支持集 S 中的各个样本，$g'(x_i)$ 是神经网络的输入样例 x_i 的编码.

其中，

$$\begin{aligned}\overrightarrow{h}_i,\overrightarrow{c}_i&=\mathrm{LSTM}(g'(x_i),\overrightarrow{h}_{i-1},\overrightarrow{c}_{i-1})\\ \overleftarrow{h}_i,\overleftarrow{c}_i&=\mathrm{LSTM}(g'(x_i),\overleftarrow{h}_{i+1},\overleftarrow{c}_{i+1})\end{aligned} \tag{78}$$

其中隐编码 h_i 和记忆单元 c_i 都是 LSTM 的输出，反向隐编码 $\overleftarrow{h}$ 的输入序列从 $i=|S|$ 开始.

(3) 对测试集进行编码

对于测试样例 $\hat{x}$ 的嵌入函数 $f(\hat{x},S)$ 是带注意力机制的 LSTM 网络 attLSTM$(\cdot,\cdot,\cdot)$，表示如下：

$$f(\hat{x},S)=\mathrm{attLSTM}(f'(\hat{x}),g(S),K) \tag{79}$$

其中 f' 与 g' 的作用类似，是对输入到 LSTM 的测试样例自身进行的编码，K 表示 LSTM 迭代的步数，编码器 $g(S)$ 表示从集合 S 到每个元素 x_i 的嵌入函数.

经过 k 步迭代后，状态如下：

$$\begin{aligned}&\hat{h}_k,c_k=\mathrm{LSTM}(f'(\hat{x}),[h_{k-1},r_{k-1}],c_{k-1})\\ &h_k=\hat{h}_k+f'(\hat{x})\\ &r_{k-1}=\sum_{i=1}^{|S|}a(h_{k-1},g(x_i))g(x_i)\\ &a(h_{k-1},g(x_i))=\mathrm{Softmax}(\boldsymbol{h}_{k-1}^{\mathrm{T}}g(x_i))\end{aligned} \tag{80}$$

其中，LSTM(x,h,c) 中的 x 是输入，h 是输出隐状态，c 是记忆单元状态. a 表示基于内容的注意力机制. 从 $g(S)$ 中得到的 r_{k-1} 与 h_{k-1} 叠加形成增广向量. 最后一步的编码输出为带注意力机制的 LSTM 网络 attLSTM$(f'(\hat{x}),g(S),K)=h_K$.

3.3　基于 MN 的扩展模型

基于 MN 的扩展模型主要包括了两个改进方向，分别是基于模型结构改进的 MN，以及基于应用而改进的 MN.

其中，基于模型结构改进的 MN 包括了端到端记忆网络、门控端到端记忆网络、动态记忆网络(Dynamic Memory Networks，DMN)、键-值记忆网络(Key-Value Memory Networks，KV-MemNN)、分层记忆网络(Hierarchical Memory Network，HMN)和基于二叉树的分层注意力记忆网络(Hierarchical Attentive Memory，HAM). 而基于应用而改进的模型包括了应用于大规模问答系统的 MN、用于视觉和文本回答系统的 DMN、应用于视频描述的 KV-MemNN 和用于未知单词答案选择的 HMN.

3.3.1　端到端记忆网络

MN 的局限性在于需要在网络的每层上进行监督学习，不容易实现反向传播训练过程. 因此提出了端到端记忆网络[4]，模型能够在输出之前从外部存储器反复读取，适用于更多不能进行监督学习的任务.

端到端记忆网络假定模型的输入为存储在存储器中的 $x_1,\cdots,x_n$ 和问题 $\boldsymbol{q}$，模型的输出为答案 a. 模型将所有的 x 写入固定大小的存储器，并使用嵌入矩阵得到 x 和 $\boldsymbol{q}$ 的特征表示，通过多层处理来输出答案 a.

(1)单层模型. 单层模型如图 19 所示. 对输入的句子$\{x_i\}$使用不同的嵌入矩阵进行编码,分别得到输入编码$\{m_i\}$和传递到输出编码$\{\boldsymbol{c}_i\}$. 通过计算输入记忆模块的$\{m_i\}$与问题编码表示 $\boldsymbol{u}$ 的内积,并使用归一化指数函数,得到每个句子与问题的相关性概率 p_i. 用该相关性概率 p_i 与输出记忆模块$\{c_i\}$加权求和,得到输出向量 $\boldsymbol{o}$. 最后输出向量 $\boldsymbol{o}$ 加上问题表示 $\boldsymbol{u}$,通过与权重矩阵 $\boldsymbol{W}$ 相乘,得到预测标签 $\hat{a}$.

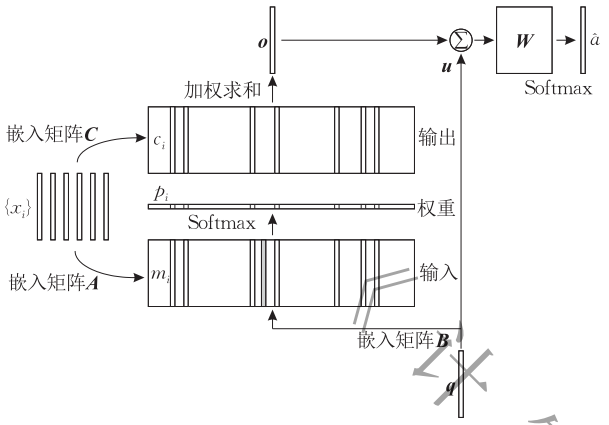

图 19 单层模型示意图

(2)多层模型. 把前面的单层模型,扩展到多层模型,如图 20 所示.

将单层模型进行堆叠,上一层的输入 u^{k+1} 是前一层的输出 o^k 和 u^k 的和,并使用线性映射 H 计算得到 u^{k+1}:

$$u^{k+1}=Hu^k+o^k \tag{81}$$

每层都有不同的嵌入矩阵 $\boldsymbol{A}^k$,$\boldsymbol{C}^k$ 用于嵌入输入$\{x_i\}$,也可以使每一层的嵌入矩阵都相等,以简化训练并减少参数数量.

在模型顶层,$\boldsymbol{W}$ 的输入也组合了顶层存储器的输入和输出,然后就得到了多层模型的预测标签 $\hat{a}$:

$$\hat{a}=\mathrm{Softmax}(\boldsymbol{W}u^{k+1})=\mathrm{Softmax}(\boldsymbol{W}(o^k+u^k)) \tag{82}$$

端到端记忆网络句子单词的编码方法有:

单词袋子表示(Bag of Words,BoW). 输入句子表示为 x_{ij},代表第 i 个句子的 j 个单词,使用独热向量表示. 那么对于第 i 个句子 $x_i=\{x_{i1},x_{i2},\cdots,x_{in}\}$,嵌入后的向量为:$\boldsymbol{m}_i=\sum_j \boldsymbol{A}x_{ij}$,$\boldsymbol{c}_i=\sum_j \boldsymbol{C}x_{ij}$. 使用同样的方法表示问题的输入向量 $\boldsymbol{u}=\sum_j \boldsymbol{B}q_j$.

位置编码(Position Encoding,PE). 不同位置的单词的权重是不一样的,对各个单词的词向量按照不同位置权重进行加权求和得到句子表示 $\boldsymbol{m}_i=\sum_j \boldsymbol{l}_j\cdot\boldsymbol{A}x_{ij}$. 对于问题、记忆输入和输出有同样的表示. 其中,

$$l_{kj}=(1-j/J)-(k/d)(1-2j/J) \tag{83}$$

这里,$\boldsymbol{l}_j$ 是 l_{kj} 组成的列向量,表示位置信息. J 是句子中的单词数量,d 是单词嵌入的维数.

时间编码. 将时序信息编码在 $\boldsymbol{T}_A$ 和 $\boldsymbol{T}_C$ 两个矩阵里面,得到对应的存储器 $\boldsymbol{m}_i$ 和输出向量 $\boldsymbol{c}_i$ 的表达式:

$$\boldsymbol{m}_i=\sum_j \boldsymbol{l}_j\cdot\boldsymbol{A}x_{ij}+\boldsymbol{T}_A(i) \tag{84}$$

$$\boldsymbol{c}_i=\sum_j \boldsymbol{C}x_{ij}+\boldsymbol{T}_C(i) \tag{85}$$

在训练期间,端到端记忆网络使用随机梯度下降过程进行训练,通过最小化预测标签 $\hat{a}$ 和真实标签 a 之间的交叉熵损失,来共同学习所有的嵌入矩阵,即模型参数.

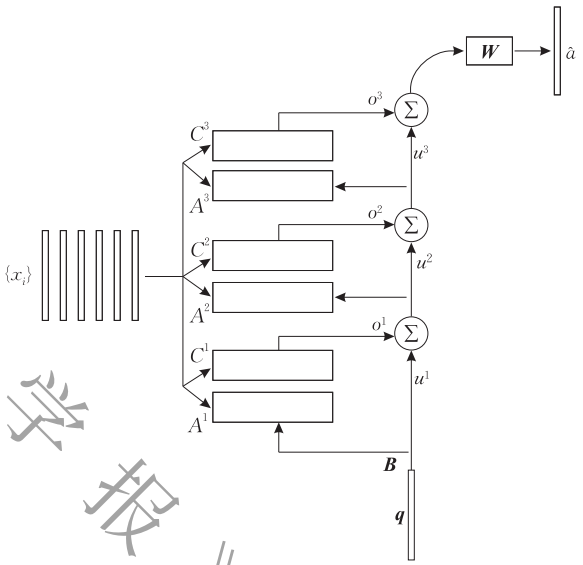

图 20 多层模型示意图

该模型在数据集 Penn Treebank 和 Text8 上,与 RNN、LSTM 和结构化约束循环网络(Structurally Constrained Recurrent Nets,SCRN)进行了比较. 实验结果显示,该模型在这两个数据集上实现了较低的困惑度.

3.3.2 门控端到端记忆网络

端到端记忆网络在面对简单自然语言推理任务时有很好的效果,但是其存储器和控制器模块之间需要更复杂的交互,很难完成多事实问答、位置推理或对话任务. 因此,文献[42]提出了门控端到端记忆网络,可以通过存储器访问层和控制器堆栈之间的门控连接来实现,使模型能够动态确定跳过基于内存的推理过程的时间和方式.

将端到端记忆网络中的 $u^{k+1}=u^k+o^k$ 看作残差的一种形式,把高速公路网络自适应选通机制的思

想，应用到端到端记忆网络中，得到了修改后的 u^{k+1} 的更新过程：

$$T^k(u^k)=\sigma(\boldsymbol{W}_T^k u^k+\boldsymbol{b}_T^k) \tag{86}$$

$$u^{k+1}=o^k\cdot T^k(u^k)+u^k\cdot(1-T^k(u^k)) \tag{87}$$

其中 $\boldsymbol{W}_T^k$，$\boldsymbol{b}^k$ 分别是第 k 层的参数矩阵和偏差向量项，T^k 是第 k 层的变换门.

3.3.3　动态记忆网络

自然语言处理中的大部分任务都可以通过语言输入转换为 QA 问题，因此 Kumar 等人提出了动态记忆网络[43]，能够处理输入序列和回答问题，形成情景记忆并产生相关回答.

动态记忆网络模型由输入、问题、情景记忆、回答四个模块组成，如图 21 所示.

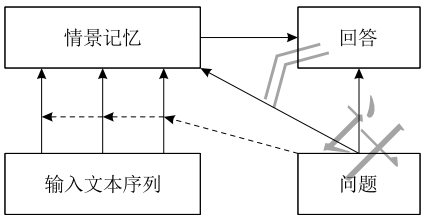

图 21　DMN 结构示意图

其中，输入模块将来自任务的原始输入文本编码成分布式向量表示. 问题模块将问题编码为分布式向量表示，该表示被传递到情景记忆模块中，形成情景记忆模块迭代的基础或初始状态. 情景记忆模块通过注意力机制选择注意力集中在哪些输入部分，然后产生一个记忆向量表示考虑到的问题以及以前的记忆信息. 每次迭代为模块提供有关输入的新的相关信息. 回答模块通过记忆模块的最终记忆向量生成答案.

(1) 输入模块

在自然语言处理问题中，输入是由 T_I 个单词组成的序列 $w_1,\cdots,w_{T_I}$，使用门控循环单元(Gated Recurrent Unit，GRU)对输入进行编码. 假设每个时刻 t 的输入为 x_t，隐状态为 $\boldsymbol{h}_t$，那么 GRU 的内部结构定义如下：

$$\begin{aligned}
z_t&=\sigma(W^{(z)}x_t+U^{(z)}h_{t-1}+b^{(z)})\\
\boldsymbol{r}_t&=\sigma(W^{(r)}x_t+U^{(r)}h_{t-1}+b^{(r)})\\
\tilde{h}_t&=\tanh(Wx_t+\boldsymbol{r}_t\circ Uh_{t-1}+b^{(h)})\\
\boldsymbol{h}_t&=z_t\circ h_{t-1}+(1-z_t)\circ\tilde{h}_t
\end{aligned} \tag{88}$$

其中，“$\circ$”表示逐元素乘法. $W^{(z)}$，$W^{(r)}$，$W\in\mathbb{R}^{n_H\times n_I}$，而 $U^{(z)}$，$U^{(r)}$，$U\in\mathbb{R}^{n_H\times n_H}$，维数 n 是一个超参数. 因此在每个时刻，使用 GRU 更新隐状态的过程可以简写为 $\boldsymbol{h}_t=GRU(x_t,h_{t-1})$.

当输入为单个句子时，输入模块计算 RNN 所有的隐状态，GRU 计算得到的输出表示的个数 $T_C=T_I$，与句子中单词个数相同，令 $\boldsymbol{c}_t$ 表示输入模块输出序列中的第 t 个元素. 当输入为多个句子时，输入模块将句子连接成一个长的单词序列，在每个句子后面插入一个句尾标记，每个句尾标记的隐状态就是输入模块的最终表示，输出表示的个数 T_C 等于句子个数.

(2) 问题模块

与输入模块类似，给定包含 T_Q 个单词的问题，问题编码在 t 时刻的隐状态 $q_t=GRU(\boldsymbol{L}[\boldsymbol{w}_t^Q],q_{t-1})$，其中 $\boldsymbol{L}$ 是嵌入矩阵，$\boldsymbol{w}_t^Q$ 是输入序列的第 t 个单词的向量表示. 问题模块的输出是最后的隐状态编码 $q=q_{T_Q}$.

(3) 情景记忆模块

该模块由内部记忆组成，使用注意力机制来更新内部记忆. 输入是当前的输入序列 c，问题表示 $\boldsymbol{q}$ 和先前的记忆 m^{i-1}. 注意力机制使用门函数 $g_t^i=G(\boldsymbol{c}_t,m^{i-1},\boldsymbol{q})$ 对候选事实 $\boldsymbol{c}_t$ 进行处理，为了计算门函数 g_t^i，首先定义一个计算输入 $\boldsymbol{c}$、记忆 $\boldsymbol{m}$ 和问题 $\boldsymbol{q}$ 之间相似度的特征向量：

$$\begin{aligned}
\boldsymbol{z}(\boldsymbol{c},\boldsymbol{m},\boldsymbol{q})=[&\boldsymbol{c},\boldsymbol{m},\boldsymbol{q},\boldsymbol{c}\circ\boldsymbol{q},\boldsymbol{c}\circ\boldsymbol{m},|\boldsymbol{c}-\boldsymbol{q}|,|\boldsymbol{c}-\boldsymbol{m}|,\\
&\boldsymbol{c}^{\mathrm{T}}W^{(b)}\boldsymbol{q},\boldsymbol{c}^{\mathrm{T}}W^{(b)}\boldsymbol{m}]
\end{aligned} \tag{89}$$

使用相似度特征向量 $\boldsymbol{z}(\boldsymbol{c},\boldsymbol{m},\boldsymbol{q})$，经过一个两层的前向神经网络来计算门函数 $G(\boldsymbol{c},\boldsymbol{m},\boldsymbol{q})$ 的值：

$$G(\boldsymbol{c},\boldsymbol{m},\boldsymbol{q})=\sigma(W^{(2)}\tanh(W^{(1)}\boldsymbol{z}(\boldsymbol{c},\boldsymbol{m},\boldsymbol{q})+b^{(1)})+b \tag{90}$$

记忆更新过程中，传送到回答模块的情景向量是 GRU 的最终状态，那么在输入序列 $c_1,\cdots,c_{T_C}$ 上使用修改后的 GRU，并用门函数 g_t^i 进行加权，得到 GRU 在时间步 t 更新的隐状态 $h_t^i=g_t^iGRU(\boldsymbol{c}_t,h_{t-1}^i)+(1-g_t^i)h_{t-1}^i$ 和最终传送到回答模块的情景向量 $\boldsymbol{e}^i=h_{T_C}^i$.

产生情景向量 $\boldsymbol{e}^i$ 之后，更新情景记忆 $m^i=GRU(\boldsymbol{e}^i,m^{i-1})$. 在经过 T_M 次传递后，把最终记忆 m^{T_M} 送给回答模块.

(4) 回答模块

回答模块也是一个 GRU 网络，初始值为 $a_0=m^{T_M}$，其输出为 $y_t=\mathrm{Softmax}(W^{(a)}\boldsymbol{a}_t)$，其中 $\boldsymbol{a}_t$ 为隐状态 $\boldsymbol{a}_t=GRU([y_{t-1},\boldsymbol{q}],a_{t-1})$，由前一时刻的输出、隐状态和问题经过 GRU 得到.

训练中使用 y_t 与答案序列概率分布的交叉熵

函数作为损失函数,进行反向传播训练,使交叉熵函数最小化得到上述各个模型参数.

该模型在 Facebook bAbI 数据集上进行了问答实验,在 20 个问答任务中获得了 93.6%的平均准确率,要略高于记忆网络的 93.3%.该模型还在 Stanford Sentiment Treebank(SST)数据集上进行情感分析的实验,在情感分析的二分类任务和细粒度分类任务中实现了最先进的性能.除此之外,还在标准华尔街日报数据集评估了该模型的词性标注性能,实验中实现了 97.5%的准确率,与最先进的模型性能相当.

3.3.4 键-值记忆网络

为了解决机器阅读文档并回答问题的困难,使用知识库来处理 QA 任务,Miller 等人提出了键值记忆网络[44],在存储器读取操作的寻址和输出阶段使用不同的编码,使读取文档变得更加容易.

KV-MemNN 的结构如图 22 所示,将上下文向量存储在存储器中的机制由键-值(Key-value)存储器实现,其中查找寻址阶段基于键(key)存储,而读取阶段基于值(value)存储.模型可以通过键和值的变换进行训练,使用随机梯度下降的标准反向传播过程实现参数的学习.

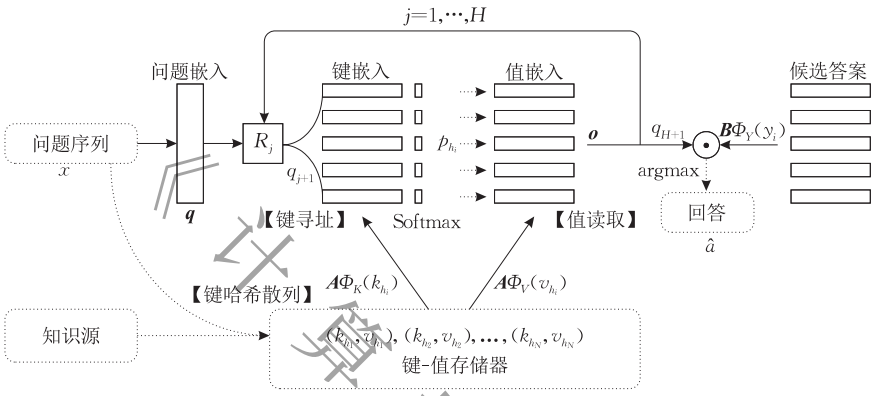

图 22 KV-MemNN 结构示意图

KV-MemNN 是基于端到端的记忆网络结构,把成对的向量$(\boldsymbol{k}_1,\boldsymbol{v}_1),\cdots,(\boldsymbol{k}_M,\boldsymbol{v}_M)$定义为记忆槽,问题表示为 x.内存的寻址和读取包括三步:

(1)键哈希散列(Key Hashing).根据输入的问题从知识源中检索与问题相关的事实,使用反向索引,找到大小为 N 的存储器子集$(k_{h_1},v_{h_1}),\cdots,(k_{h_N},v_{h_N})$,其中键与频率小于 1000 的问题之间至少共享一个词.

(2)键寻址(Key Addressing).即对存储器进行相关性评价.寻址期间,将问题与键进行比较,使用 Softmax 函数计算每个存储器键 k_{h_i} 与问题 x 的相似性,得到概率:

$$p_{h_i}=\text{Softmax}(\boldsymbol{A}\Phi_X(x)\cdot\boldsymbol{A}\Phi_K(k_{h_i})) \tag{91}$$

其中,Φ 是 D 维的特征映射函数,$\boldsymbol{A}$ 是一个维度为 $d\times D$ 的矩阵.

(3)值读取(Value Reading).值读取步骤中,通过对键寻址概率的加权求和来读取存储器的值,返回输出向量 $\boldsymbol{o}=\sum_i p_{h_i}\boldsymbol{A}\Phi_V(v_{h_i})$.

这样就完成了一层的操作.计算输入问题 x 的特征映射表示 $\boldsymbol{q}=\boldsymbol{A}\Phi_X(x)$,将值读取得到的输出向量 $\boldsymbol{o}$ 与 $\boldsymbol{q}$ 相加经过映射,得到输入到下一层的 $q_2=\boldsymbol{R}_1(\boldsymbol{q}+\boldsymbol{o})$,其中 $\boldsymbol{R}_1$ 是一个 $d\times d$ 维的矩阵,并且在第 j 个跳段(hop)使用不同的 R_j.重复这个过程,每一跳段都会更接近答案.通过更新的输入问题向量表示 $\boldsymbol{q}$ 得到键的寻址概率 $p_{h_i}=\text{Softmax}(\boldsymbol{q}_{j+1}^{\mathrm{T}}\boldsymbol{A}\Phi_K(k_{h_i}))$.

最后,在 H 个跳段之后,计算出最终预测的可能输出:

$$\hat{a}=\arg\max_{i=1,\cdots,C}\text{Softmax}(\boldsymbol{q}_{H+1}^{\mathrm{T}}\boldsymbol{B}\Phi_Y(y_i)) \tag{92}$$

其中 $y_i\in Y$ 是数据集中所有可能的候选答案.使用交叉熵作为损失函数,对模型端到端进行反向传播训练.

3.3.5 分层记忆网络

当记忆网络需要从极大的存储器中进行读取时,计算复杂性很高,因此 Chandar 等人提出了基于最大内积搜索(Maximum Inner Product Search,MIPS)的记忆选择机制,并建立了分层记忆网络(Hierarchical Memory Network,HMN)模型[45].核心思想是以分层的方式来构建存储器,与传统的普通存储方法相比,读取数据时需要处理的步骤少,更容易训练.

与 MN 相比,HMN 具有两个不同的存储和读取模块.

(1)存储模块.HMN 使用分层存储结构来代替传统存储器结构的单元阵列,并提出了三种存储器

结构构造方法：

① 基于散列(hash)的方法. 把记忆分成多个单元，只在与问题相关的单元附近进行 MIPS.

② 基于树的方法. 在叶子节点上创建带有记忆单元的搜索树，只对问题路径的叶子节点执行 MIPS.

③ 基于聚类的方法. 把单元聚类成多个相关的簇，只输出与问题最相关的簇并执行 MIPS.

(2) 读取模块. HMN 的读取只会在存储器中选出的子集上使用软注意力机制，选择过程是由 MIPS 算法控制的，可以使用有组织的存储器分层结构在子线性时间内检索最相关的事实. HMN 的读取模块通过更新输入来进行 MIPS，这样 MIPS 检索的结果包含正确的事实.

HMN 还提出了 K-MIPS 的概念：输入一组点 $\chi=\{x_1,\cdots,x_n\}$ 和一个问题向量 $\boldsymbol{q}$，K-MIPS 的目标函数是 $\arg\max^{(K)}_{i\in\chi}\boldsymbol{q}^{\mathrm{T}}x_i$，其中 $\arg\max^{(K)}$ 是返回 K 个使得 $\boldsymbol{q}^{\mathrm{T}}x_i$ 取最大值的 $\chi=\{x_1,\cdots,x_n\}$ 中点的下标索引. χ 对应于存储器，$\boldsymbol{q}$ 是输入模块计算得到的向量.

HMN 模型通过使用 K-MIPS 算法组织存储器，然后训练读取模块执行 MIPS. 读取模块使用 Softmax 函数，在每个读取步骤检索一组相关的候选单元：

$$\begin{aligned}C&=\arg\max^{(K)}h(\boldsymbol{q})\boldsymbol{M}^{\mathrm{T}}\\R_{\mathrm{out}}&=\mathrm{Softmax}^{(K)}(h(\boldsymbol{q})\boldsymbol{M}^{\mathrm{T}})\\&=\mathrm{Softmax}(h(\boldsymbol{q})\boldsymbol{M}[C]^{\mathrm{T}})\end{aligned}\tag{93}$$

其中 $h(\boldsymbol{q})\in\mathbb{R}^d$ 是问题的表示，C 是最大 K 个 MIP 候选单元的下标索引集，$\boldsymbol{M}\in\mathbb{R}^{N\times d}$ 是存储器，N 是存储器中的单元总数，$\boldsymbol{M}[C]$ 是 $\boldsymbol{M}$ 的子矩阵，其中 $\boldsymbol{M}[C]$ 的行通过 C 索引值得到.

基于聚类的近似 K-MIPS 算法是由 Auvolat 等人提出[46]，这种方法优于其他数据依赖和数据独立的近似 K-MIPS 方法. 对于大多数近似 MIPS 算法，可将 MIPS 转换为最大余弦相似搜索(Maximum Cosine Similarity Search，MCSS)问题：

$$\arg\max^{(K)}_{i\in\chi}\frac{\boldsymbol{q}^{\mathrm{T}}\boldsymbol{x}_i}{\|\boldsymbol{q}\|\|\boldsymbol{x}_i\|}=\arg\max^{(K)}_{i\in\chi}\frac{\boldsymbol{q}^{\mathrm{T}}\boldsymbol{x}_i}{\|\boldsymbol{x}_i\|}\tag{94}$$

当所有的数据向量 $\boldsymbol{x}_i$ 有相同的范数时，MCSS 与 MIPS 是等价的. 在存储单元和输入表示添加附加的维数，能够将 MIPS 转换为 MCSS. 在存储单元和输入向量上使用 P 和 Q 两个映射，使数据向量的范数相同，这样就把 MIPS 转换成 MCSS：

$$\begin{aligned}&P(x)=[x,1/2-\|x\|_2^2,1/2-\|x\|_2^4,\cdots,1/2-\|x\|_2^{2^m}]\\&Q(x)=[x,0,0,\cdots,0]\end{aligned}\tag{95}$$

因此，对于任何问题向量 $\boldsymbol{q}$ 的 MIPS 都具有这样的近似等价关系：

$$\arg\max^{(K)}_{i}\boldsymbol{q}^{\mathrm{T}}\boldsymbol{x}_i\simeq\arg\max^{(K)}_{i}\frac{Q(\boldsymbol{q})^{\mathrm{T}}P(\boldsymbol{x}_i)}{\|Q(\boldsymbol{q})\|_2\cdot\|P(\boldsymbol{x}_i)\|_2}\tag{96}$$

将 MIPS 转换为 MCSS 之后，就可以使用 K 均值或文献[47]分层聚类的方法来近似并加快余弦相似性搜索过程. 存储器中的数据聚类后，每次读取只需要 K 个点积，其中 K 是聚类质心的个数.

但是，这种方法只得到近似值，训练过程中会在梯度下降过程引入偏差，从而影响 HMN 的性能. 为了解决这个问题，提出了以下三种方法：

(1) 使用 K 个最大的单元索引添加到 mini-batch 中的所有读取查询中. 这样可以通过 GPU 进行有效的矩阵乘法运算，也有助于减少近似误差带来的偏差.

(2) 对于每次的读取访问，不仅使用近似值最大的几个聚类，还按照与质心点积成比例的概率分布，对其它的聚类随机抽样，这样也可以减少偏差.

(3) 也可以对存储器进行随机采样，添加到最大 K 个候选单元中.

3.3.6　基于二叉树的分层注意力 MN

前文所介绍的记忆网络大多是为了处理更大的存储器所提出的，但是存储器的访问效率容易被忽视. 大多数的存储器结构具有 $\Theta(n)$ 的访问复杂度，其中 n 是存储器的大小，因此在复制长度为 n 的序列时，需要执行 $\Theta(n^2)$ 次运算，这样的效率太低. 为了提高效率，Andrychowicz 等人提出了分层注意力记忆网络(Hierarchical Attentive Memory，HAM)[48]，基于叶子节点对应记忆单元的二叉树，使模型以 $\Theta(\log n)$ 复杂度进行存储器访问，相对于标准注意力机制具有的 $\Theta(n)$ 复杂度有了极大的改进.

HAM 模型由基于二叉树的 HAM 记忆模块和 LSTM 控制器组成，如图 23 所示. 模型的存储器是

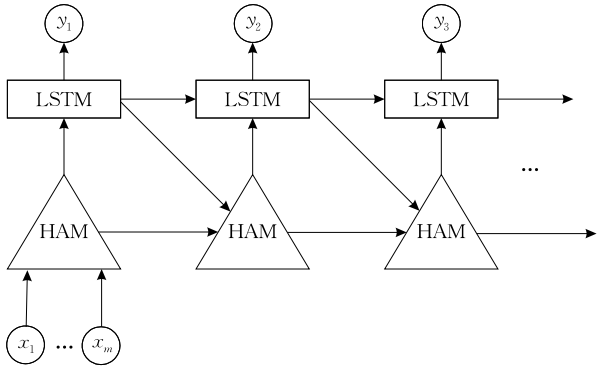

图 23　HAM 模型整体结构示意图

二叉树的结构,其中叶子节点包含存储在存储器中的数据,内部节点包含一些辅助数据,对更有效地查找到所需要的记忆内容有帮助.

假定输入序列为$x_1,x_2,\cdots$,输出序列为$y_1,y_2,\cdots$,并且$\boldsymbol{x}_i,\boldsymbol{y}_i\in\{0,1\}^b$是维数为$b$的二值向量,输入序列长度$\leq n$,其中$n$是2的正整数次幂.

HAM模型基于具有n个叶子节点的完全二叉树实现.$|V|=2n-1$表示二叉树的节点数,$L\subset V$表示叶子节点的子集,并且用$l(e)$和$r(e)$表示节点e的左节点和右节点,其中$e\in V\backslash L$为内部节点.

HAM模型将二叉树和注意力机制与LSTM结合,HAM包括以下4个过程:模型初始化、注意力机制、输出和更新.

(1)模型初始化.使用EMBED($\boldsymbol{x}_i$)将叶子节点的值初始化,没有输入$\boldsymbol{x}_i$的叶子节点的值初始化为0.然后使用JOIN自下而上初始化内部节点,得到父节点$h_e=\text{JOIN}(h_{l(e)},h_{r(e)})$.

(2)注意力机制阶段.从根部自上而下地执行SEARCH操作,向右的概率为p,向左的概率为$1-p$,直到到达叶子节点,得到注意力机制叶子节点a.

(3)输出.将注意力机制叶子节点a的值h_a输入到LSTM中,得到输出$y_t\in\{0,1\}^b$,经过sigmoid函数得到y_t的概率分布.

(4)更新.如图24所示,对注意力机制叶子节点a的值使用WRITE进行更新,然后自下而上地更新内部节点.

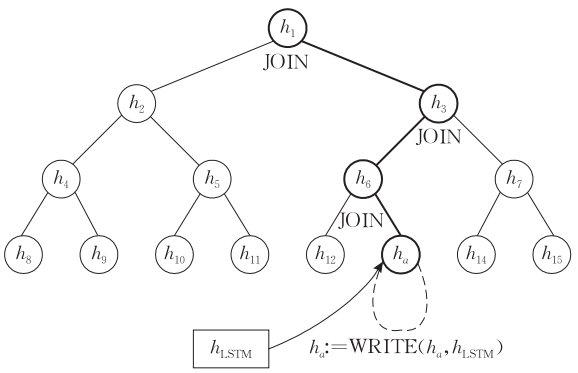

图24 HAM模块更新过程

其中,EMBED是$\mathbb{R}^b\to\mathbb{R}^d$的映射函数,JOIN是$\mathbb{R}^d\times\mathbb{R}^d\to\mathbb{R}^d$的映射函数,SEARCH是$\mathbb{R}^d\times\mathbb{R}^l\to[0,1]$的映射函数,WRITE是$\mathbb{R}^d\times\mathbb{R}^l\to\mathbb{R}^d$的映射函数.这四个映射变换可以使用任意的函数逼近器表示,例如多层感知器(Multilayer Perceptrons,MLPs).

训练时使用强化学习从纯输入-输出样本中训练模型[49].设x,y是输入输出序列,θ表示模型的参数,A表示执行过程中所有选择向左或向右的顺序.为了得到正确的输出,求输出y的最大似然概率:

$$L=\log p(y|x,\theta)=\log\Big(\sum_A p(A|x,\theta)p(y|A,x,\theta)\Big) \tag{97}$$

使用变分法,转化为最小化变分下界问题:

$$F=\sum_A p(A|x,\theta)p(y|A,x,\theta)\leq L \tag{98}$$

并使用强化学习来学习近似梯度:

$$\nabla F=\sum_A p(A|x,\theta)[\nabla\log p(y|A,x,\theta)+\log p(y|A,x,\theta)\nabla\log p(A|x,\theta)] \tag{99}$$

最后使用蒙特卡洛方法近似求得模型参数的估计值.

该模型将使用LSTM作为控制器的HAM与LSTM和使用注意力机制的LSTM进行比较,测试模型堆栈、FIFO和优先级队列的能力.实验结果显示使用注意力机制能够极大地减少LSTM在任务中的错误率(分类任务错误率由99%降到25%),而使用LSTM作为控制器的HAM错误率仅有0.04%.在2至4倍长度的输入时,具有注意力机制的LSTM也无能为力(仅能完成搜索操作),而使用LSTM作为控制器的HAM错误率最高仅为2.48%,而且时间复杂度只有$\Theta(\log n)$.

3.8.7 应用于大规模问答系统的MN

训练大规模问答系统时,一次只能训练很少的样本,因此在记忆网络模型的基础上,对于大规模问答数据集SimpleQuestions,文献[50]提出了嵌入式的QA系统.

嵌入式的QA系统模型包括以下4个模块:

(1)输入模块.完成存储器到内存的候选数据库(Freebase)实体,系统需要回答的问题,以及扩展内存的候选答案集(Reverb)实体这三类数据的预处理.具体执行过程分三步:

①将候选数据库实体重新定义为包含主体s、关系r和对象o的三元组.具有k个对象的实体$y=(s,r,\{o_1,\cdots,o_k\})$表示为向量$\boldsymbol{f}(y)\in\mathbb{R}^{N_S}$,这里$N_S$是所有实体和关系的个数,$\boldsymbol{f}(y)$的每一维对应一个关系或者实体,主体和关系的输入值为1,对象的输入值为$1/k$.

②将问题$\boldsymbol{q}$表示为$g(\boldsymbol{q})\in\mathbb{R}^{N_V}$,这里$N_V$是词典里单词的个数.问题$\boldsymbol{q}$对应的单词的值为1,其他设置为0.

③将候选答案集的实体$y=(s,r,o)$表示为向

量 $\boldsymbol{h}(y)\in\mathbb{R}^{N_S+N_V}$，是主体 s、对象 o 的符号袋子和关系 r 的单词袋子模型表示，维数大小是前两者之和.

(2) 泛化模块. 泛化模块用于预计算实体链接，即将候选答案集实体的主体和对象链接到基于候选数据库实体的存储器结构上. 这样就可以使用候选数据库训练出来的模型查询候选答案集实体.

(3) 输出模块. 给定输入问题后，输出模块执行内存查找，返回满足查询条件的支持实体，提供问题的答案.

为了避免对所有实体进行打分，首先使用近似实体链接生成一小组候选实体，然后再进行打分. 具体执行过程分两步：

① 首先从问题中生成所有可能的 n 元语法，只保留作为实体的 n 元语法，并丢弃所有作为子序列的 n 元语法. 最后在候选数据库实体中，检索与五个最长匹配的 n 元语法最多的链接，最后得到需要保留的两个实体.

② 使用两个嵌入矩阵 $\boldsymbol{W}_V\in\mathbb{R}^{d\times N_V}$ 和 $\boldsymbol{W}_S\in\mathbb{R}^{d\times N_S}$，计算问题 $\boldsymbol{q}$ 和候选数据库候选实体 y 的相似度，得到相似度打分：

$$S_{\mathrm{QA}}(\boldsymbol{q},y)=\cos(\boldsymbol{W}_V g(\boldsymbol{q}),\boldsymbol{W}_S \boldsymbol{f}(y)) \tag{100}$$

其中 $\boldsymbol{W}_V$ 和 $\boldsymbol{W}_S$ 分别表示词典中单词的嵌入矩阵和候选数据库实体关系的嵌入矩阵.

当对候选答案集中的实体 y 打分时，使用嵌入矩阵 $\boldsymbol{W}_{VS}\in\mathbb{R}^{d\times(N_V+N_S)}$ 计算余弦相似度：

$$S_{\mathrm{RVB}}(\boldsymbol{q},y)=\cos(\boldsymbol{W}_V g(\boldsymbol{q}),\boldsymbol{W}_{VS}\boldsymbol{h}(y)) \tag{101}$$

这里，嵌入矩阵 $\boldsymbol{W}_V$ 和 $\boldsymbol{W}_S$ 是需要从训练集中学习的参数. 最后根据计算出的相似度选择与问题最相似的候选实体.

(4) 响应模块. 响应模块处理输出模块的结果，并计算出预期的答案，返回所选支持实体的对象集合.

该模型在 WebQuestions，SimpleQuestions 和 Reverb 三个测试集上进行了评测，并与最先进的模型进行比较.

在基准 WebQuestions 数据集上，该模型获得了与最佳模型(41.3%)相当的 $F1$ 得分 41.2%. 在 SimpleQuestions 数据集上的实验结果，最优准确率达到 62%～63%，而支持事实的实验结果表明，支持事实有大约 86%落在 SimpleQuestions 问题的候选集中，这说明该模型对候选答案重新排名是很有效的. 在 Reverb 数据集上的实验结果显示，该模型的准确度是 67%，接近最先进的记忆网络模型.

3.3.8　用于视觉和文本问答系统的 DMN

DMN 在语言任务中有很好的预测准确率，Xiong 等人在此基础上对记忆模块和输入模块进行了改进[51]，不需要在训练期间标记支持事实，并且还能够回答视觉问题. DMN 用于视觉和文本问答系统包含文本 QA 的输入模块、视觉 QA 的输入模块和情景记忆模块组成.

(1) 文本 QA 的输入模块

DMN 使用单个 GRU 来处理输入文本中的所有单词，通过存储句子末端标记产生的隐状态来提取句子表示. 但是 GRU 只能获得句子级上文关系，而句子级上文关系可能不能很好地反映单词级别上的上下文关系，为了实现单词级 GRU 交互，Xiong 提出使用两个不同的模块替代单个 GRU.

句子阅读器. 使用位置编码，负责将单词编码成句子嵌入表示. 单词序列$[w_1^i,\cdots,w_{M_i}^i]$的编码经过位置加权得到每个句子的编码 f_i：

$$f_i=\sum_{M}^{j=1}\boldsymbol{l}_j\circ w_j^i \tag{102}$$

$$l_{jd}=(1-j/M)-(d/D)(1-2j/M) \tag{103}$$

其中，$\boldsymbol{l}_j$ 是由 l_{jd} 作为元素组成的位置编码列向量，d 是嵌入向量的分量的索引值，即分量的下标，D 是嵌入的维数，M 是句子中的单词个数.

输入融合层. 采用句子阅读器得到的输入事实编码 f_i，并使用双向 GRU，实现句子里单词之间正向和反向依赖关系的捕捉：

$$\begin{aligned}\overrightarrow{f}_i&=GRU_{fwd}(f_i,\overrightarrow{f}_{i-1})\\ \overleftarrow{f}_i&=GRU_{bwd}(f_i,\overleftarrow{f}_{i-1})\\ \overleftrightarrow{f}_i&=\overrightarrow{f}_i+\overleftarrow{f}_i\end{aligned} \tag{104}$$

(2) 视觉 QA 的输入模块

把图像分成小块，并把每一块看成文本中的一个句子. 视觉 QA 的输入模块包括以下三个部分：

① 局部特征提取. 使用基于 VGG-19 模型的 CNN 从图像中抓取特征.

② 视觉特征嵌入. 增加了一个双曲正切激活函数的非线性层，将局部区域特征向量投影到问题向量 $\boldsymbol{q}$ 使用的文本特征空间中.

③ 输入融合层. 前面提取的局部特征向量并没包含全局信息，因此增加了一个输入融合层. 首先遍历整个图像得到输入事实 F，然后在这些输入事实 F 上应用双向 GRU，产生全局感知输入事实 $\overleftrightarrow{\boldsymbol{F}}$.

(3) 情景记忆模块

情景记忆模块通过将注意力集中在输入事实的一个子集上，从视觉 QA 的输入模块提供给它的输入事实 $\overleftrightarrow{\boldsymbol{F}}=[\overleftrightarrow{f}_1,\cdots,\overleftrightarrow{f}_N]$ 中检索信息. 通过注意门 g_i^t

和每个事实 $\vec{f}_i$ 的关联来实现注意力机制，并通过事实 $\vec{f}_i$ 与问题表示 $\boldsymbol{q}$ 和情景记忆之间的交互来进行计算，得到注意门 g_i^t：

$$\begin{aligned} z_i^t &= [\vec{f}_i \circ \boldsymbol{q}; \vec{f}_i \circ \boldsymbol{m}^{t-1}; |\vec{f}_i - \boldsymbol{q}|; |\vec{f}_i - \boldsymbol{m}^{t-1}|] \\ Z_i^t &= W^{(2)} \tanh(W^{(1)} z_i^t + b^{(1)}) + b^{(2)} \\ g_i^t &= \frac{\exp(Z_i^t)}{\sum_{k=1}^{M_i} \exp(Z_i^t)} \end{aligned} \tag{105}$$

其中，$\vec{f}_i$ 表示第 i 个事实，$\boldsymbol{m}^{t-1}$ 是前一时刻的情景记忆，“$\circ$”表示逐元素乘法，符号“;”表示把各个向量叠加起来，形成增广向量.

有了注意门 g_i^t 后，可以使用软注意力机制或者基于注意力机制的 GRU 来提取上下文向量 $\boldsymbol{c}_t$.

软注意力机制通过注意门 g_i^t 和排序好的事实向量 $\vec{\boldsymbol{F}}$ 的加权求和，得到上下文向量 $\boldsymbol{c}_t = \sum_{i=1}^{N} g_i^t \vec{f}_i$. 而基于注意力机制的 GRU 通过对传统 GRU 模型进行修改，得到更好的性能.

使用注意门 g_i^t 替代 GRU 中的更新门 u_i，如图 25 所示，使得 GRU 通过注意门来更新其内部状态：

$$h_i = g_i^t \circ \tilde{h}_i + (1 - g_i^t) \circ h_{i-1} \tag{106}$$

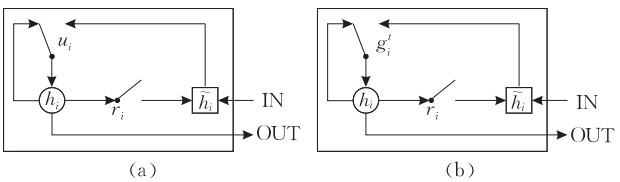

图 25 传统 GRU 模型(a)与基于注意力机制的 GRU 模型(b)的对比

得到的最终隐状态 h_N 就是上下文向量 $\boldsymbol{c}_t$. 通过上下文向量 $\boldsymbol{c}_t$ 和前一时刻的情景记忆向量 $\boldsymbol{m}^{t-1}$ 来更新 $\boldsymbol{m}^t$：

$$\boldsymbol{m}^t = GRU(c^t, \boldsymbol{m}^{t-1}) \tag{107}$$

其中，GRU 的初始隐状态设置为问题向量 $\boldsymbol{q}$.

或者也可以使用 ReLU 来更新记忆向量 $\boldsymbol{m}^t$：

$$\boldsymbol{m}^t = \mathrm{RuLU}(W^t[\boldsymbol{m}^{t-1}; c^t; \boldsymbol{q}] + b) \tag{108}$$

其中“;”表示把各个向量叠加起来，形成增广向量，$W^t \in \mathbb{R}^{n_H \times n_H}$，$b \in \mathbb{R}^{n_H}$，$n_H$ 是隐层包含的神经元的个数.

除此之外，Ramachandran 等人对 DMN 也做了扩展，提出了动态记忆张量网络(Dynamic Memory Tensor Network，DMTN)[52]. 由于注意力机制是模型性能提高的关键，因此使用神经张量网络[53]来制定相似性度量机制，用作打分的特征向量. 该模型与 DMN 比较，能处理的任务个数提高了 80%以上，与 Facebook 提出的端到端记忆网络相比在 bAbI 数据集上可处理的任务个数增加了 20%.

该模型在视觉问答(Visual Question Answering，VQA)数据集上进行实验. VQA 数据集由三个问题域组成：是/否、数字和其他. 实验结果表明，该模型在这三个问题域优于其它基线模型以及最先进的方法，尤其是在其他问题域上，具有更为广泛的优势.

3.3.9 应用于视频描述的 KV-MemNN

Jain 等人将 KV-MemNN 应用于多模态场景，处理序列到序列模型的新型键寻址机制中[54]，Jain 等人提出的模型将视频分解为视觉和语言片段，将每一帧画面作为键，其语义信息作为值，将这二者匹配起来作为键-值对处理，从而获得从视觉到语义嵌入空间的依赖关系[54].

模型基于编码器-解码器框架. 编码器网络学习从输入序列映射到固定长度的向量表示，将其传递给解码器以生成输出序列. 模型包括编码器模块、键值存储器和解码器模块三个部分.

(1) 编码器模块. 在长度为 T 的视频中，编码器模块把输入序列 $X = \{I_1, \cdots, I_T\}$ 的图像映射到相应的固定大小的上下文表示向量序列. 处理视频时，使用 CNN 编码器提取特征向量，并用 RNN 编码器处理提取的特征向量.

输入每一帧图像 $I_i \in \mathbb{R}^{N \times M}$，CNN 编码器学习从图像到维数为 D 的上下文表示的映射关系 f：$\mathbb{R}^{N \times M} \to \mathbb{R}^D$. RNN 编码器按顺序处理从 CNN 编码器提取的每一帧的特征，在每个时间步生成隐状态 $h_i^e = g(f(I_i), h_{i-1}^e)$，包括所有 T 个图像序列信息. 在保持时间依赖性的同时，将可变长度序列映射到固定长度的上下文向量.

(2) 键-值存储器. 由键-值对 $(k_1, v_1), \cdots, (k_T, v_T)$ 组成记忆槽，键和值用于把视觉空间前后关系转换到语言空间，并有效提取视觉特征和文本描述之间的关系. 定义键、值和键寻址、值读取如下：

键(Key). 使用 CNN 编码器，提取视频每一帧 I_i 的特征向量，这些特征向量通过 RNN 编码器合并为顺序结构，隐状态 h_i^e 被提取出来，作为表示视觉前后关系的键 $\boldsymbol{k}_i = h_i^e$.

值(Value). 对于每一帧图像 I_i，通过预训练模型 ψ 联合建模图像的视觉和语义嵌入信息，得到对应键的文本语义嵌入 $v_i = \psi(I_i)$，对于视频的每一帧，都有键-值记忆槽 $(\boldsymbol{k}_i, v_i)$.

键寻址. 键寻址过程如图 26 所示，使用软注意

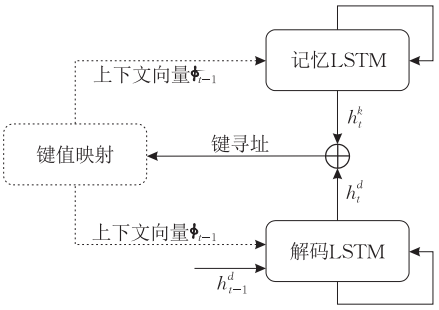

图 26　键寻址过程示意图

力机制得到每个存储器的对齐概率 a_i. 键寻址使用 LSTM,以前一时间步读取的值 $\boldsymbol{\phi}_{t-1}(K)$ 作为输入来读取过去的值,输出新的隐状态 $h_t^k=f^k(\boldsymbol{\phi}_{t-1}(K),h_{t-1}^k)$,其中 f^k 是循环网络神经元激活函数.

在 t 时刻,问题 $\boldsymbol{q}$ 是解码器隐状态 h_{t-1}^d 和键寻址隐状态 h_t^k 的加权求和:

$$\boldsymbol{q}=W_k h_t^k+W_d h_{t-1}^d \tag{109}$$

然后使用问题 $\boldsymbol{q}$ 和时刻 i 的键向量 $\boldsymbol{k}_i$ 通过双曲正切激活函数(tanh),得到第 i 个时刻特征向量 $\boldsymbol{k}_i$ 的相关性得分:

$$e_i^t=\boldsymbol{w}_t\tanh(\boldsymbol{q}+U_a\boldsymbol{k}_i) \tag{110}$$

其中 W_k,W_d,$\boldsymbol{w}_t$,U_a 是模型的参数.

最后,使用 Softmax 函数得到新的注意力分布概率:

$$\alpha_i^t=\exp\{e_i^t\}\Big/\sum_{j=1}^{N}\exp\{e_j^t\} \tag{111}$$

这样将视觉信息和语言信息分离成为键-值对,可以为解码器提供更好的上下文信息.

值读取. 记忆槽的值是键值特征向量的加权和.

$$\phi_t(K)=\sum_{i=1}^{T}\alpha_i^{(t)}\boldsymbol{k}_i,\ \phi_t(V)=\sum_{i=1}^{T}\alpha_i^{(t)}v_i \tag{112}$$

其中,α_i^t 是键寻址过程中求得的每个记忆槽的分布概率,作为权重将键和值的特征向量加权求和,得到该记忆槽的键 $\phi_t(K)$ 用于在下一时间步的键寻址,而记忆槽的值 $\phi_t(V)$ 作为输入传递到解码器产生下一个单词.

(3)解码器模块. 解码器也使用 LSTM,可以对大范围的依赖关系进行训练. LSTM 模型除了 RNN 中的隐状态 $\boldsymbol{h}_t$ 之外还具有存储记忆单元 $\boldsymbol{c}_t$,能够有效汇总在当前时刻以前观察到的信息. LSTM 的主要结构是三个控制门:输入门控制当前输入 x_t,遗忘门 f_t 自适应地遗忘旧的记忆,输出门 o_t 决定单元存储器输出到隐状态的程度. 然后在隐状态 $\boldsymbol{h}_t$ 上使用单层神经网络,通过 Softmax 函数得到候选单词的概率分布 p_t:

$$p_t=\text{Softmax}(U_p[\boldsymbol{h}_t,x_t,\phi_t(V)])+b_p \tag{113}$$

3.3.10　用于未知单词答案选择的 HMN

记忆网络在句子级存储器上进行推理,而没有把注意力集中在单词上,会使模型丢失一些细节信息. 因此文献[55]提出了与上文不同的分层记忆网络,将过去的事实分别编码成句子级和单词级存储,在句子级存储器上的推理模块上使用 k 最大池化,对与问题最相关的 k 个句子进行采样,并将这些句子送入单词级存储器的注意力机制,集中在与问题最相关的选定句子中的单词. 最后通过句子级推理模块和单词级注意力机制的输出共同得到问题答案预测.

模型结构包括句子级存储推理模块和单词级注意力机制模块,其整体结构如图 27 所示.

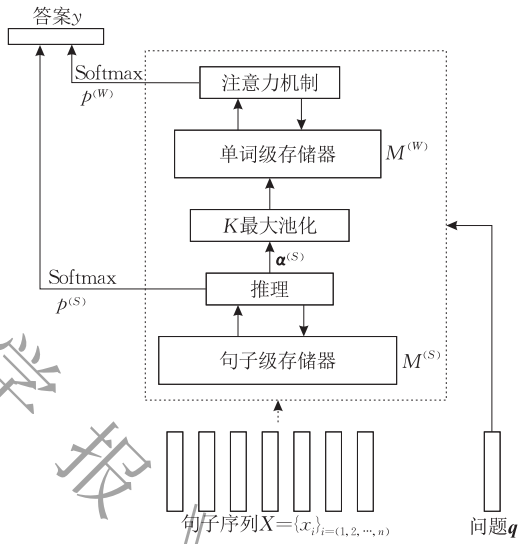

图 27　HMN 整体结构示意图

句子级存储推理模块. 输入一组 n 个句子 $X=\{x_i\}_{i=(1,2,\cdots,n)}$ 和问题 $\boldsymbol{q}$,将句子集 X 分别映射到句子级存储模块 $M^{(S)}$ 和具有低维分布表示的单词级存储模块 $M^{(W)}$. 利用句子级存储模块来软搜索(soft-search)相关句子. 基于软搜索结果,通过 k 最大池化方法对最相关的句子进行抽样,并且利用注意力机制来关注所选语句的单词级记忆. 目标回答 y 同时指导学习句子级存储的推理过程和单词级注意力机制.

句子级存储推理模块与端到端记忆网络完全相同,使用相邻权重结合的方案进行多重交互,最后得到句子级存储器推理模块预测的单词概率分布 $p^{(S)}$.

***K* 最大池化**. 对句子级存储推理模块最后一层计算出来的权重向量 $\boldsymbol{\alpha}^{(S)}$ 进行池化,采样出最大 k 个与问题最相关的句子,提供给单词级注意力机制

模块,将注意力集中在相关的单词上,减少计算的复杂度.

单词级的注意力机制. 使用双向 GRU 计算所有单词的隐状态,对于第 t 个单词 $\bar{\omega}_t$,前向 GRU 和反向 GRU 分别将其编码得到隐状态 $\overrightarrow{h}_t=G\overrightarrow{RU}(\boldsymbol{C}^R\bar{\omega}_t)$ 和 $\overleftarrow{h}_t=G\overleftarrow{RU}(\boldsymbol{C}^R\bar{\omega}_t)$,其中隐状态的维数与单词嵌入的维数相等,$\boldsymbol{C}^R$ 是句级存储器最后一层的单词嵌入矩阵. 然后将前向GRU隐状态 $\overrightarrow{h}_t$ 和反向GRU隐状态 $\overleftarrow{h}_t$ 求和得到单词对应的记忆信息 $M^{(W)}=\{\boldsymbol{m}_t\}_{t=(1,2,\cdots,|t|)}$,其中 $\boldsymbol{m}_t=\overrightarrow{h}_t+\overleftarrow{h}_t$. 这样,记忆向量 $\boldsymbol{m}_t$ 就包含了句子集合 X 中第 t 个单词 $\bar{w}_t$ 的上下文信息.

对于所选的包含有序单词序列 $\hat{\omega}=\{\hat{\omega}_t\}_{t=(1,2,\cdots,|t|)}$ 的某个句子 $\hat{X}$,计算句子级存储器中最后一层的问题向量 $\boldsymbol{u}_R^{(S)}$,和单词子集 $\hat{\omega}$ 对应的记忆子集 $\{\hat{m}_t\}_{t=(1,2,\cdots,|\hat{t}|)}$ 注意力对齐向量. 那么在单词级存储器中,归一化注意力权重 $\alpha^{(W)}=\{\alpha_t^{(W)}\}_{t=(1,2,\cdots,|\hat{t}|)}$ 为

$$\alpha_t^{(W)}=\mathrm{Softmax}(\boldsymbol{v}^{\mathrm{T}}\tanh(\boldsymbol{W}\boldsymbol{u}_R^{(S)}+\boldsymbol{U}\hat{m}_t)) \tag{114}$$

其中 $\boldsymbol{v}\in\mathbb{R}^{d\times 1}$,$\boldsymbol{W}\in\mathbb{R}^{d\times d}$,$\boldsymbol{U}\in\mathbb{R}^{d\times d}$ 是训练过程中需要学习的参数.

按照 Vinyals 等人的方法[56],在单词子集 $\hat{\omega}$ 上归一化注意力权重 $\alpha^{(W)}$,作为输出单词的概率分布:

$$p^{(W)}(\omega)=trans(p^{(W)}(\hat{\omega}))=trans(\alpha^{(W)}) \tag{115}$$

其中 $trans(\cdot)$ 表示单词子集概率分布 $p^{(W)}(\hat{w})\in\mathbb{R}^{|\hat{t}|}$ 到所有单词概率分布 $p^{(W)}(w)\in\mathbb{R}^{|V|}$ 的映射,就是将子集中的单词概率添加到字典中单词对应的位置,在这个过程中将没有选中单词的概率设置为 0.

通过句子级存储器模块 $M^{(S)}$ 输出的单词概率分布 $p^{(S)}$ 和单词级存储器模块 $M^{(W)}$ 输出的 $p^{(W)}$,可以预测输出单词的联合概率分布:

$$p(w)=p^{(S)}(w)+p^{(W)}(w) \tag{116}$$

最后,使用目标回答 y 来同时指导学习句子级存储器的推理过程和单词级的注意力机制,并选择交叉熵作为损失函数,使用随机梯度下降算法进行训练[57].

该模型在四个合成域对话数据集上进行答案选择任务,包括两个机票预定域和两个酒店预订域,与记忆网络进行比较,并且分别比较了仅使用句子级推理模块和仅使用单词级推理模块的分层记忆网络. 实验结果显示,仅使用单词级推理模块的 HMN 的测试错误数量要少于仅使用句子级推理模块的 HMN,而且联合使用两个模块能获得更好的性能,测试错误数量远小于记忆网络.

3.4 其他类型的记忆网络

除了上述基于记忆网络基础模型的扩展模型,还有很多学者提出了自己对于记忆网络的构想,这些模型结构包括主动长期记忆网络、前馈神经网络的简化注意力模型、使用虫洞连接的记忆增强神经网络、用于知识库问答的事实记忆网络、存储器增强的神经网络和记忆增强的神经图灵机.

3.4.1 主动长期记忆网络

当人工神经网络顺序学习不同任务时,持续学习的内容会受到干扰和遗忘,因此 Furlanello 等人提出了主动长期记忆网络(Active Long Term Memory Networks, A-LTM)[58],它是一种连续多任务深度学习模型,能够在获取知识的同时保持之前学习的输入与输出之间的关联关系. 模型利用深度神经网络的非凸性质,使用蒸馏损失(Distillation Loss)[59]来保留之前学习的非活跃任务的知识.

基于 McClelland 等人提出的海马理论[60],哺乳动物通过海马(Hippocampus)和新皮质(Neocortex)来保持长期记忆,平衡稳定性和灵敏度,因此既能够体验到新的环境,又不会使自身承受突变环境的风险,从而避免灾难性推理(Catastrophic Inference, CI)行为.

受此启发,A-LTM 模型的输入和输出使用双系统来表现隐因子的变化. 第一部分是成熟稳定的新皮质网络(N),在发育阶段具有监督源的同质环境中进行训练. 第二部分是灵活的海马网络(H),在一般的非结构化记忆学习环境,从 N 初始化,并从 H 的输出活动中规范化. 这种双重机制可以在保持 N 的稳定性的同时关注新的输入.

A-LTM 把学习过程分为两个学习场景:发育期和成熟期. 在发育期,N 在受控环境中训练,其中存在相同对象的多个示例. 使用多任务目标函数训练 N 来预测对象的语义和图像的类标签. 收敛后,N 的学习率为 0.

在成熟期,用 N 学习的结果初始化 H 网络,H 网络学习场景发生改变,其中对象通常只有单一视图,并且类别数量增加两个数量级. H 用多任务目标函数进行训练,用来预测新的高维语义学习任务,并使用经验回放机制预测缺少输出行为标签的发育任务 N 的输出,使用经验回放机制情况下,N 具有区分同一对象的不同视图的能力,即先前发育期的经验和成熟期知识.

因此,A-LTM 学习的核心思想是所有的记忆任务都需要兼顾过去的经验才能够找到一个发育期和成熟期组成的多任务的最优解. 因此,如果在新的学习环境下缺少标签,导致输入输出的不确定,那么

H 学习中缺失的信息必须用 N 的预测值来进行补充. 在这种情况下，对付输入分布的变化产生的不稳定性，需要辅之以回放(auxiliary replay)机制.

定义智能体与连续环境相互作用关系为联合概率分布 $P(y,x)$，包括视觉刺激 $x\in X$ 和智能体动作的隐因子 $y\in Y$. 智能体通过感知机制 $\psi(x):X\mapsto S$ 来接收刺激信息，并根据深度神经网络感知的层次表示 $\phi(s):S\mapsto \Phi^d$ 进行决策. 通过 Softmax 函数，将最后一层 $\phi^d(s)$ 转换为动作的概率分布. 对于特定的环境信息 x，使用正确的动作行为 y 来对训练过程进行监督. 通过最小化交叉熵损失函数 $L(\phi^d(s),y)$ 来更新深度神经网络隐层形成的层次表示 ϕ.

定义发育期学习场景 1，此时智能体与场景相互作用关系为联合概率分布 $P_1(y_1,x_1)$；成熟期学习场景 2，此时智能体与场景相互作用关系为联合概率分布 $P_2(y_2,x_2)$. 以下讨论中仅考虑从学习场景 1 到学习场景 2 的简单学习过程. 假定动作神经网络的输入输出映射函数为 $f(w_0,w_1,w_2;x):X=(x_1,x_2)\mapsto Y=(y_1,y_2)$，其中 w_0 是学习场景 1 到 2 共享的参数，w_1,w_2 是从共有表示 x 到单独任务智能体动作的隐因子 y_1,y_2 映射的特定参数.

A-LTM 学习多任务序列过程包含以下两步来解决优化问题：

(1) 用目标函数 $\min\limits_{w_0,w_1}L(f(w_0,w_1;x_1),y_1)$，学习场景 1 的联合概率分布 $P_1(y_1,x_1)$ 和动作 $f(w_0,w_1;x_1)$，其中 w_0,w_1 的初始值 w_0^0,w_1^0 由高斯分布 $N(0,\sigma)$ 随机抽样得到.

(2) 用目标函数 $\min\limits_{w_0,w_2}L(f(w_0,w_2;x_2),y_2)$ 学习场景 2 的联合概率分布 $P_2(y_2,x_2)$ 和动作 $f(w_0,w_1;x_1)$，学习场景 1 的 $P_1(y_1,x_1)$ 的结果作为 $P_2(y_2,x_2)$ 的初始值. 其中共享参数 w_0 的初始值 $w_0^0=w_0^*$，w_0^* 是学习场景 1 中的 w_0 的最终学习结果，而 w_2 的初始值为 $w_2^0\sim N(0,\sigma)$.

那么，学习场景 1 和 2 组成多任务学习，省略多任务学习在学习场景 1 和 2 目标函数之间的权衡比例因子，同时求解这两个任务的目标函数为

$$\min_{w_0,w_1,w_2}L(f(w_0,w_1;x_1),y_1)+L(f(w_0,w_2;x_2),y_2) \tag{117}$$

其中初始值 w_0^0,w_1^0,w_2^0 均服从高斯分布 $N(0,\sigma)$.

学习任务缺少学习场景 1 的输出行为标签 y_1，只有学习场景 2 的输出行为标签 y_2 时，可以使用经验回放知识蒸馏学习结果替代缺少的学习场景 1 的输出行为标签 y_1. A-LTM 使用经验回放，用过去学习场景 1 训练的稳定模块新皮质网络 N 替代缺失的学习场景 1 的输出行为标签 y_1，这样 H 面对新的学习场景 2 时，任务学习的目标函数为

$$\begin{aligned}\min_{w_0,w_1,w_2}\ & L(f(w_0,w_1;x_1),f(w_0^*,w_1^*;x_1))+\\ & L(f(w_0,w_2;x_2),y_2)\\ & w_0^0=w_0^*,w_1^0=w_1^*,w_2^0\sim N(0,\sigma)\end{aligned} \tag{118}$$

其中 $f(w_0^*,w_1^*;x_1)$ 通过以前训练好的新皮质网络 N，根据经验回放来模拟正确的行为 y_1.

3.4.2　前馈神经网络的简化注意力模型

Raffel 等人提出了一种适用于前馈神经网络的注意力机制简化模型[61]，其结构如图 28 所示，可以解决序列的长期记忆问题.

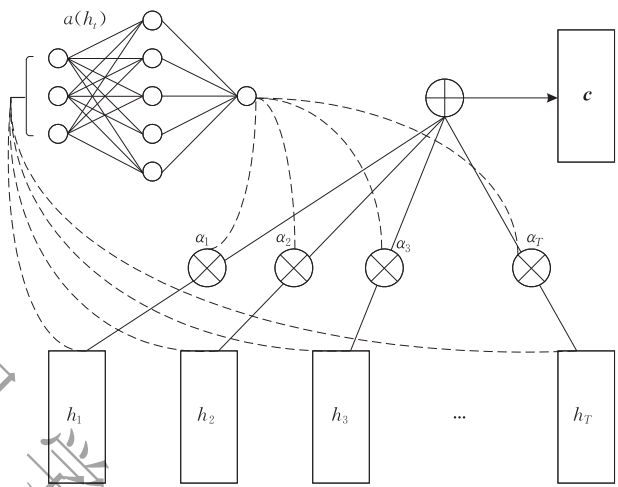

图 28　前馈注意力机制模型结构示意图

注意力机制是一种学习序列长期依赖关系的方法，它能够抓住模型在不同时间状态之间更直接的依赖性. 基于 Bahdanau 等人的研究[62]，模型在每个时刻产生输入序列的隐状态 $\boldsymbol{h}_t$，通过引入对齐注意力向量 $\boldsymbol{\alpha}_{tj}$ 作为权重，对计算的输入隐状态序列 h_j 加权求和，得到模型的上下文向量 $\boldsymbol{c}_t$：

$$\boldsymbol{c}_t=\sum_{j=1}^{T}\boldsymbol{\alpha}_{tj}h_j \tag{119}$$

其中 T 是输入序列长度，$\boldsymbol{\alpha}_{tj}$ 是每个隐状态 h_j 在 t 时间步的权重：

$$e_{tj}=a(s_{t-1},h_j),\boldsymbol{\alpha}_{tj}=\frac{\exp(e_{tj})}{\sum\limits_{k=1}^{T}\exp(e_{tk})} \tag{120}$$

其中 a 是一个学习函数，在给定前一时刻的输出隐状态 s_{t-1} 和输入的隐状态序列 h_j 时，计算当前序列重要程度.

然后计算新的输出隐状态序列元素 s_t 的条件后验概率，s_t 取决于前一时刻的输出隐状态 s_{t-1}，上下文向量 $\boldsymbol{c}_t$ 和模型 $t-1$ 时刻的输出：$s_t=f(s_{t-1},\boldsymbol{c}_t,y_{t-1})$.

将上述注意力机制进行简化，如图所示，前馈注意力机制模型将输入隐状态序列 $\boldsymbol{h}_t$ 输入函数 $a(\boldsymbol{h}_t)$ 中，产生概率对齐向量 $\boldsymbol{\alpha}$. 然后以 $\boldsymbol{\alpha}$ 作为权重对 $\boldsymbol{h}_t$ 加权求和，得到上下文向量 $\boldsymbol{c}_t$.

将上述前馈注意力机制简化，用于从整个序列当中产生单个上下文向量 $\boldsymbol{c}_t$：

$$\boldsymbol{e}_t=a(\boldsymbol{h}_t),\ \alpha_t=\frac{\exp(\boldsymbol{e}_t)}{\sum_{k=1}^{T}\exp(\boldsymbol{e}_t)},\ \boldsymbol{c}=\sum_{t=1}^{T}\alpha_t\boldsymbol{h}_t \tag{121}$$

其中 α_t 是对齐函数，并只取决于 $\boldsymbol{h}_t$. 使用注意力权重 α_t 计算状态序列 $\boldsymbol{h}_t$ 的自适应加权和，得到输入序列的固定长度的嵌入上下文向量 $\boldsymbol{c}$. 使用注意力机制可以随着时间集成与当前输出最相关的输入序列隐信息，因此模型可以处理可变长度序列建模问题.

当序列长度 T 固定时，没有注意力机制的前馈神经网络模型可以用于序列数据建模. 而当 T 变化时，需要增加时间积分，最简单的方法就是计算状态序列 $\boldsymbol{h}_t$ 的未加权的平均值：

$$\boldsymbol{c}=\frac{1}{T}\sum_{t=1}^{T}\boldsymbol{h}_t.$$

假定模型的输入为 x_t，计算隐状态为

$$\boldsymbol{h}_t=\mathrm{LReLU}(W_{xh}x_t+b_{xh}) \tag{122}$$

其中 $W_{xh}\in\mathbb{R}^{D\times 2}$，$b_{xh}\in\mathbb{R}^{D}$，$\mathrm{LReLU}(x)$ 为带泄漏非线性整流函数，其中

$$\mathrm{LReLU}(x)=\max(x,0.01x).$$

令对齐函数为 $a(\boldsymbol{h}_t)=\tanh(W_{hc}\boldsymbol{h}_t+b_{hc})$，根据式(105)计算上下文向量 $\boldsymbol{c}$，并计算中间向量 $\boldsymbol{s}$：

$$\boldsymbol{s}=\mathrm{LReLU}(W_{cs}\boldsymbol{c}+b_{cs}),W_{cs}\in\mathbb{R}^{D\times D},b_{cs}\in\mathbb{R}^{D} \tag{123}$$

最后输出 y：

$$y=\mathrm{LReLU}(W_{sy}\boldsymbol{s}+b_{sy}),W_{sy}\in\mathbb{R}^{1\times D},b_{sy}\in\mathbb{R} \tag{124}$$

训练时用输出 y 与预测值的平方误差作为损失函数，并使用 adam 自适应梯度下降算法优化参数[63].

3.4.3 使用虫洞连接的记忆增强神经网络

Gülçehre 等人提出序列暂态自动关系发现(Temporal Automatic Relation Discovery In Sequences，TARDIS)模型，提出了一种新型记忆增强神经网络[64]，通过使用虫洞连接减少梯度消失的影响，更容易学习序列长期依赖性关系. 模型的控制器将过去的隐状态选择性存储到外部存储器中，在需要时重新访问. TARDIS 的内存作为虫洞连接过去的存储器，有助于梯度的传播和时间依赖性的学习.

TARDIS 的结构如图 29 所示，是具有外部存储矩阵 $\boldsymbol{M}_t$ 的记忆增强神经网络，使用 RNN 控制器生成读取权重 w_t^r，与存储器 $\boldsymbol{M}_t$ 点乘得到读取向量 $\boldsymbol{r}_t=(\boldsymbol{M}_t)^{\mathrm{T}}w_t^r$. TARDIS 的控制器将当前隐状态的线性映射写入存储器：$\boldsymbol{M}_t[i]=W_m\boldsymbol{h}_t$.

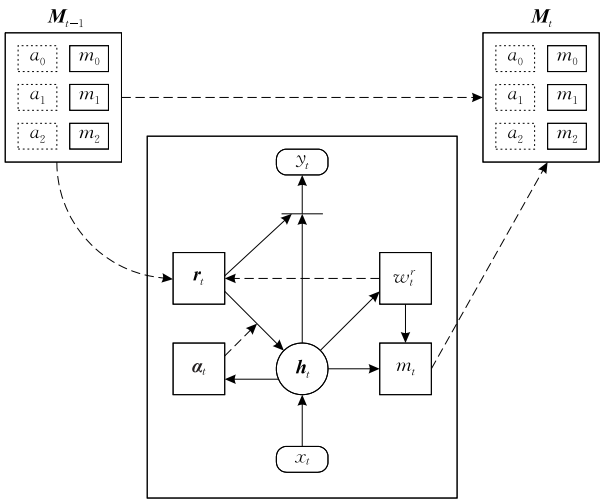

图 29 TARDIS 结构示意图

在每个时间步，控制器的隐状态 $\boldsymbol{h}_t$ 取决于从存储器读取的内容 $\boldsymbol{r}_t$，当前的输入 x_t 和前一时间步的隐状态 h_{t-1}：$\boldsymbol{h}_t=\phi(x_t,h_{t-1},\boldsymbol{r}_t)$.

存储器中的每个单元都是先前隐状态的线性映射，因此从存储器中读取内容时，控制器隐状态的调整可以看作跨时间创建快捷连接的一种方式.

TARDIS 实现机制主要包括寻址机制和控制器的读写操作.

寻址机制. TARDIS 的存储矩阵 $\boldsymbol{M}_t$ 包括不相交的地址部分 $A_t\in\mathbb{R}^{k\times a}$ 和内容部分 $C_t\in\mathbb{R}^{k\times c}$，即存储矩阵为 $\boldsymbol{M}_t=[A_t;C_t]\in\mathbb{R}^{k\times(c+a)}$. 存储矩阵的地址向量固定为随机稀疏向量. 控制器可读存储器的地址和内容部分，但只会写入存储器的内容部分.

连续读取权重 $\bar{w}_t^r$ 由多层感知器生成：

$$\begin{aligned}\pi_t[i]&=\boldsymbol{a}^{\mathrm{T}}\tanh(W_h^{\gamma}\boldsymbol{h}_t+W_x^{\gamma}x_t+W_m^{\gamma}\boldsymbol{M}_t[i]+W_u^{\gamma}\boldsymbol{u}_t)\\ \bar{w}_t^r&=\mathrm{Softmax}(\pi_t)\end{aligned} \tag{125}$$

其中 $\{\boldsymbol{a},W_h^{\gamma},W_x^{\gamma},W_m^{\gamma},W_u^{\gamma}\}$ 是需要学习的参数，读取权重 w_t^r 是从 $\bar{w}_t^r$ 抽样或者通过 $\arg\max\bar{w}_t^r$ 得到的独热向量表示. 对当前时刻之前的读取权重求和并归一化，得到 $\boldsymbol{u}_t$：

$$\boldsymbol{u}_t=\mathrm{norm}\Big(\sum_{i=1}^{t-1}w_i^r\Big) \tag{126}$$

向量 $\boldsymbol{u}_t$ 表示访问存储器中每个单元的频率，可以帮助注意力机制基于访问每个单元的频率进行注意力选择.

TARDIS 控制器. TARDIS 使用 LSTM 作为控

制器，并使用存储器中读取的内容来修改门，并对重置门使用 Gumbel-sigmoid 激活函数，控制器 LSTM 的控制单元 $\boldsymbol{c}_t$ 和隐层计算如下：

$$\begin{aligned}\tilde{c}_t&=\tanh(\beta_t W_h^g h_{t-1}+W_x^g x_t+\alpha_t W_r^g \boldsymbol{r}_t)\\ \boldsymbol{c}_t&=f_t c_{t-1}+i_t \tilde{c}_t\\ \boldsymbol{h}_t&=o_t\tanh(\boldsymbol{c}_t)\end{aligned}\tag{127}$$

其中 f_t，i_t，o_t 分别是 LSTM 的遗忘门、输入门和输出门，α_t，β_t 是重置门的标量参数，用来控制从存储器和前一隐状态的信息流入 LSTM 的控制单元 $\boldsymbol{c}_t$ 的程度.

控制器可以通过选择要读取和写入的单元，创建虫洞连接，来表示输入之间的依赖关系.

3.4.4　用于知识库问答的事实记忆网络

在学习问题回答的任务时，记忆网络可以有效处理复杂的推理并具有可扩展性. Jain 提出的事实记忆网络[65]，可以从知识库中提取和推理相关事实来学习回答问题.

在建立模型之前，首先需要对候选数据库(Freebase)进行事实提取. 将所有可能的问题单词与候选数据库实体的替代名称进行匹配，并丢弃所有作为另一个 n 元(n-gram)语法的子序列的 n 元语法，将其余实体之一作为对象的所有事实都添加到候选事实列表中.

事实记忆网络模型由线性连接的 L 个计算层和一个输出函数组成. 初始计算层将候选事实列表和问题嵌入作为输入，输出函数将最终计算层作为输出，以候选数据库实体的形式生成一个答案列表.

其中，模型的计算层包括两个输入：

(1)事实列表 $F=\{f_1,\cdots,f_{|F|}\}$，其中每个事实 f 的形式是 $(s,\boldsymbol{R},o)$，其中 $\boldsymbol{R}$ 是主体 s 和对象 o 之间的关系向量.

(2)计算问题嵌入表示 $\boldsymbol{q}\in\mathbb{R}^d$.

对于每一个事实 $f\in F$，计算它的非归一化的分数 $g(f)$ 和归一化的分数 $h(f)$. 使用打分函数 $g(f)$ 计算输入的问题嵌入表示和假设问题 $\boldsymbol{q}'=(s,\boldsymbol{R})$ 之间的相似度 $g(f)=\boldsymbol{q}^{\mathrm{T}}(s+\boldsymbol{R})$. 如果事实有多个表述方式，则计算它们相似度的平均值，最后使用 Softmax 函数得到归一化分数 $h(f)$，归一化分数 $h(f)$ 能够反映出各个事实的重要程度.

然后根据计算的分数修改事实列表 $F=\{f_1,\cdots,f_{|F|}\}$ 和问题嵌入表示 $\boldsymbol{q}$，分为以下三步：

(1)事实修剪. 选择一个阈值，在事实清单中移除打分比阈值要小的事实，事实修剪能够大大缩短回答问题的响应时间，可以探索问题实体周围的更大的搜索空间.

(2)问题嵌入. 将所收集的知识 $f=(s,\boldsymbol{R},o)$ 加入问题嵌入表示 $\boldsymbol{q}$ 中：

$$\boldsymbol{q}'=\boldsymbol{q}+\sum_{f\in F}h(f)(s+\boldsymbol{R})\tag{128}$$

(3)事实添加. 对于属于事实 f 的每个对象实体 o，找到知识库中相关的所有事实，如果事实 f 的打分分数大于阈值，则将其添加到事实列表 F 中. 修改后的事实列表与新问题嵌入表示 $\boldsymbol{q}'$ 一起构成这一层的输出，提供给下一个计算层或输出函数作为输入.

输出函数根据前一层的输入计算得分 $h(f)$，答案集由最高得分的事实的对象实体组成.

在训练过程中，用于知识库问答的事实记忆网络需要最小化的损失函数为

$$L_{\mathrm{QA}}=\sum_{(\boldsymbol{q},A)}\sum_{n=1}^{L}\frac{n}{L}\left\|\,|F_n|\sum_{a\in A}a-|A|\sum_{f\in F_n}h(f)\cdot o\right\|^2\tag{129}$$

$(\boldsymbol{q},A)$ 是回答问题训练集 D 中的问-答对，其中 $\boldsymbol{q}$ 是问题，A 是包含问题正确答案的回答列表. n 指的是网络中的第 n 个计算层(共有 L 个计算层). 该损失函数表示给定事实列表的对象实体 o 接近给定答案列表 a 的程度，并使用 $h(f)$ 加权. 最小化该损失函数会使模型网络生成更短的问题回答路径，更快地从问题中获得答案.

3.4.5　存储器增强的神经网络

使用内部存储器增强的神经网络可以学习复杂任务，但是由于内存的规模很小，限制了该网络适用的范围. 因此，Rae 等人提出一种端到端的可微内存访问方法，称为稀疏访问内存(Sparse Access Memory，SAM)[66].

对于包含 N 个单词的存储器，SAM 能够在 $\Theta(\log N)$ 的时间复杂性执行前向后向访问，只使用 $\Theta(N)$ 空间实现初始化，每个时间步占用 $\Theta(1)$ 的空间.

SAM 包括读取、写入和控制器三个主要部分.

(1)读取. 稀疏读取操作定义为在存储器中选定单词的加权平均值：

$$\tilde{r}_t=\sum_{i=1}^{K}\tilde{w}_t^R(s_i)\boldsymbol{M}_t(s_i)\tag{130}$$

其中 $\tilde{w}_t^R$ 为非零实体，通过 $s_1,\cdots,s_K$ 进行索引，是稀疏的权重向量.

对于读取基于内容的寻址权重 w_t^R，保留 K 个最大非零实体并将其余设为 0，K 个最大值分别对应问题 $\boldsymbol{q}$ 的 K 个最接近的点，使用近似最近邻(Approximate Nearest Neighbors，ANN)方法来计算 $\tilde{w}_t^R$.

(2)写入. 写入操作权重同样被约束为包含一

定数量的非零实体.控制器写入先前读取的位置,更新上下文相关的存储器或最近访问的位置.写入权重定义为

$$w_t^W=\alpha_t(\gamma_t w_{t-1}^R+(1-\gamma_t)I_t^U) \tag{131}$$

其中 γ_t 是控制器输出的插值门参数,α_t 是写入门参数,w_{t-1}^R 是前一时间步的读取权.最近最少访问的单词 I_t^U 在写入之前设置为 0,当读取操作稀疏(权重有 K 个非零实体)时,写入操作也是稀疏的.I_t^U 定义为存储器中单词的指示变量,当单词使用值 U 最少时为 1,其余情况为 0.

单词使用值有两种定义,第一种是写入权重的时间衰减总和:

$$U_T^{(1)}(i)=\sum_{t=0}^{T}\lambda^{T-t}(w_t^W(i)+w_t^R(i)) \tag{132}$$

其中 λ 是衰减因子.

另一种定义为

$$U_T^{(2)}(i)=T-\max\{t:w_t^W(i)+w_t^R(i)>\delta\} \tag{133}$$

其中 δ 是一个调整参数.

(3)控制器.控制器使用单层 LSTM 实现,在每个时刻,LSTM 输入 x_t 和前一时刻读取的单词 r_{t-1},然后通过线性层产生用于存储器访问的读取和写入参数的向量 $\boldsymbol{p}_t=(q_t,\boldsymbol{a}_t,\alpha_t,\gamma_t)$.然后,把当前时刻从存储器中读取的单词 $\boldsymbol{r}_t$ 与 LSTM 的输出向量连接形成增广向量,通过线性层得到最终的输出 y_t.

3.4.6 记忆增强的神经图灵机

NTM 中包含模拟记忆过程的存储器结构,用来存储和检索信息以简化学习算法.但是当存储器很大时,模型难以收敛,容易过拟合.因此 Zhang 等人提出了不同的记忆增强的神经图灵机存储器结构来解决这个问题[67].

首先定义一下存储器可见性的概念:如果能够通过控制器写入操作直接修改存储器,则称为"受控",如果不能则称为"隐藏".Zhang 基于原始的 NTM 提出了三种变体:NTM1、NTM2 和 NTM3,它们的结构如图 30 所示.

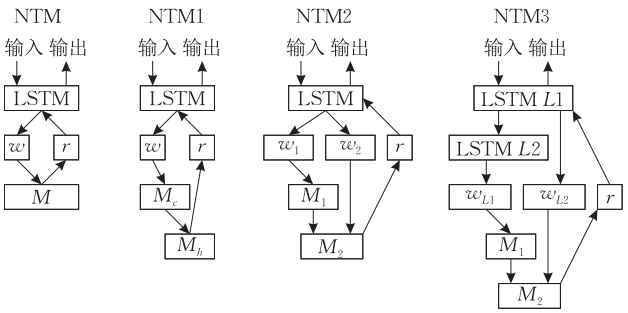

图 30 NTM 及三种变体结构示意图

(1)NTM1 与原始 NTM 相比,增加了额外的隐存储器 M_h,M_h 不受控制器模块控制,而是与受控存储器 M_c 相连,隐存储器 M_h 将内容积累保存在受控存储器 M_c 中,从而使存储器读写过程平滑,防止出现大的偏离.时刻 t NTM1 的读写过程为

$$\begin{aligned}&M_c(t)=h(M_c(t-1),w(t-1),c(t))\\&M_h(t)=aM_h(t-1)+bM_c(t)\\&r(t)=w_r(t)M_h(t)\end{aligned} \tag{134}$$

M_c 在 $t-1$ 时刻通过写头 $w(t-1)$ 更新,生成 t 时刻的控制存储器输出,并用来更新隐存储器 M_h.然后新的隐存储器用来在时间 t 生成读头 $r(t)$.其中 $c(t)$ 是控制器的输出,h 是更新受控存储器和写入权重的函数,用来实现擦除和添加的功能.a 和 b 是标量混合权重,并且读头从 $M_h(t)$ 读取.

(2)NTM2 同样使用了两个存储器 M_1 和 M_2,并且分层连接.存储器 M_2 通过另一个写头 w_2 连接到控制器的输出上,这样两个存储器都是连接到两个不同的写头的受控存储器.而且 M_1 仅由写头 w_1 修改,而 M_2 由 M_1 和写头 w_2 共同修改.那么 NTM2 模型的读写过程为

$$\begin{aligned}&M_1(t),w_1(t)=h(M_1(t-1),w_1(t-1),c(t))\\&\widetilde{M}_2(t),w_2(t)=h(M_2(t-1),w_2(t-1),c(t))\\&M_2(t)=a\widetilde{M}_2(t)+bM_1(t)\\&r(t)=w_r(t)M_2(t)\end{aligned} \tag{135}$$

其中,M_1 和 M_2 是由相同控制器生成的,但具有不同的写入权重,而读取只从 M_2 存储器输出.

(3)NTM3 与前两种的不同,使用了两层控制器,并且每层控制器通过写头的非线性变换,写入到不同的存储器.更深层的存储器由上层存储器和写头共同更新.那么模型的读写过程为

$$\begin{aligned}&M_1(t),w_{L1}(t)=h(M_1(t-1),w_{L1}(t-1),c_{L1}(t))\\&\widetilde{M}_2(t),w_{L2}(t)=h(M_2(t-1),w_{L2}(t-1),c_{L2}(t))\\&M_2(t)=a\widetilde{M}_2(t)+bM_1(t)\\&r(t)=w_r(t)M_2(t)\end{aligned} \tag{136}$$

模型的存储器不仅从最终层 LSTM 的输出接受写入,而且从中间层输出接受写入操作.这样的目的是,具有中间层 LSTM 的输出可以使最后一层的存储器平滑变化.

3.5 小结与分析

本节主要介绍了各种记忆网络的扩展模型,从它们实现记忆的机理入手,对不同的记忆网络模型进行了详细的分类和比较.

这些模型实现记忆的方式有以下四种:通过 RNN 的神经元节点传递信息、通过 LSTM 的遗忘

门和记忆门处理信息、外部存储器、注意力机制，根据其记忆方式的不同的分类结果如表 8 所示. 其中，应用最多也最为广泛的是使用外部存储器来保留需要记忆的信息，这是因为外部存储器的结构简单明了，可塑性强，具有很好的变种和泛化能力，还可以与其他类型的神经网络结合. 而注意力机制通过变送器模型的发展，未来也将会与更多的神经网络模型结合，甚至跳出机器学习的领域，有更为广泛的应用方向.

表 8　记忆网络扩展模型

记忆方式	模型
RNN 前后神经元节点传递信息	具有长期记忆的 RNN
LSTM 遗忘门和记忆门	ALSTM、TLSTM
外部存储器	MemN2N、GMemN2N、DMN、KV-MemNN、HMN、SAM、TARDIS、基于 NTM 的记忆网络模型
注意力机制	AMSRN、HAM、前馈神经网络的注意力简化模型

4　记忆网络应用

记忆网络模型从发展初始就有了极为广泛的应用，其中涵盖的领域包括了自然语言处理、计算机视觉、语音处理，还有很多在其他领域的应用，比如医学领域、工业流程领域等等.

4.1　自然语言处理

问答任务. Weston 等人最初提出记忆网络[2]主要是用于问题回答任务，其中长期记忆作为知识库，模型输出文本作为问题回答. Taylor 等人在此基础上做了更深入的研究，提出了能够实现大规模问题回答的系统[28]，在 WebQuestions 基准数据集上达到最高的性能指标.

Samothrakis 等人在记忆网络的基础上做出了改进，使用递归软最大函数来增强匹配功能，提出了匹配记忆循环网络[68]，可对存储器上的问题进行编码、解码和回答.

而 Sukhbaatar 等人解决了记忆网络需要支持事实进行监督学习的问题，提出了端到端记忆网络[4]，使记忆网络的应用更加广泛.

Caballero 将 Skip-thought vectors 模型[69]添加到现有的端到端记忆网络框架里[70]，可以学习多参数多跳段语义关系，以完成 QA 任务.

主题标签推荐. 在端到端记忆网络的基础上，Huang 等人引入了分层注意力机制[71]，将 twitter 文本信息和相应的用户兴趣信息组合在一起进行主题标签推荐任务.

真值发现. Li 等人将记忆网络应用于真值发现[72]，使用前馈存储器网络和反馈存储器网络来学习关于同一对象的语句的可信的表示，并采用记忆机制来学习源信息的可靠性，并通过真值预测来使用源信息.

机器阅读. Cheng 等人将 LSTM 用于机器阅读[40]，从左到右地处理文本序列，并通过记忆和注意力机制进行浅层推理. 并通过语言模型、情感分析和自然语言推理这三个方面的实验表明了提出的机器阅读器的性能与现有技术相当或者更优.

机器理解(Machine Comprehension)及问答是自然语言处理中的代表性问题，Pan 等人提出了一种多层嵌入记忆网络，用于机器阅读任务[73]. 该模型使用多层嵌入来编码文档，实现全向匹配(Full-orientation matching)的记忆网络，以获得上下文和问题之间的交互关系.

嵌入表示. Palangi 等人将 LSTM 用于深度句子嵌入表示学习[74]，模拟上下文信息，还将句子的关键信息嵌入到一个语义向量中，并使其随着时间推移从新输入中获取有用的信息. 此模型对噪声具有很好的鲁棒性，而且每个单元通常分配有特定的关键词. 将其应用于 Web 文档检索中，效果明显优于常用的句子嵌入表示学习方法.

分布式语义模型通常需要足够的单词示例来学习高质量的语义向量表示，而人类则可以从一个或几个代表性示例中猜出一个词的意思. 因此，Sun 等人提出了一种基于存储器的单词嵌入式表示学习方法[75]，增量从大型语料库中学习，并快速适应新添加的微小数据，能够从相当有限的上下文中获取高质量的单词嵌入式表示.

机器翻译. Wang 等人引入外部记忆增强的 RNN 解码器[76]，可以显著提高汉译英翻译任务的性能.

人机对话. Ganhotra 等人在端到端记忆网络的基础上，提出了基于知识的端到端记忆网络[77]. 该模型把现有的知识库实体作为先验知识，在对话任务中，从知识库中提取信息，进行信息匹配，从而实现直接的人机对话互动过程.

顺序推荐. Huang 等人将 RNN 与键-值记忆网络集成在一起，整合知识库信息来增强键-值记忆网

络的语义表示能力,完成顺序推荐任务[78].该模型利用大规模知识库信息来改进顺序推荐过程,将顺序偏好表示与属性偏好表示组合在一起,作为用户偏好的最终表示.

情感分类. Tang 等人提出了用于情感分类的深度记忆网络[79],可以在对情感分类时,获取上下文单词的重要程度,与 LSTM 相比这种方法更简单快捷,并且效果更好.

Chen 等人提出了一个具有存储器结构的循环注意力神经网络[80],可以在评论中识别意见目标的情绪.该模型使用多层注意力机制来获取大范围分离的情感特征,从而对不相关的信息具有更好的鲁棒性.

语义表示. Tran 等人提出了乘法树结构的 LSTM[81],是现有树形结构 LSTM 的扩展,用于合并树中节点之间的关系信息.该模型在子节点上定义了不同的组合函数,可以更好地表达句子,完成语义表示任务.

4.2 计算机视觉

图像标注和图像内容描述. Jia 等人将 LSTM 用于图像文字描述任务[82].模型将从图像中提取的语义信息作为额外输入添加到 LSTM 中,用于学习与图像内容更紧密耦合的文字描述.这样的模型可以更好地描述图像内容,在各种基准数据集上实现了最好的性能.

在键-值记忆网络基础上,Jain 等人将其扩展到视频领域[54],把视频字幕分解成视觉和语言片段,作为键-值对处理,并通过键-值记忆网络完成视频字幕标注任务.

Chunseong Park 提出了上下文序列记忆网络[83](Context Sequence Memory Network,CSMN),不同用户对同一图像有着不同的个性化描述.模型以用户的独特词汇表作为先验知识,在 Instagram 数据集上可以完成个性化的图像标注和图像文字描述任务.

Vinyals 等人将记忆网络用于单样本学习[41],采用基于深度神经特征度量学习的思想,通过使用外部记忆增强神经网络,可以从少量样本中进行学习,不仅可以用于图像的学习,对语言任务也有很高的预测精度.而 Santoro 等人则提出了专注于存储器内容的方法[33],同样能够很好地进行单样本学习.

Wang 等人提出了一种多模态记忆模型用来描述视频内容[84].该模型构建了视觉和文本信息共享的存储器,用来模拟长期视觉文本的相互依赖性,并进一步模拟视觉注意力过程.该模型与神经图灵机类似,外部存储器通过与具有多个读取和写入操作的视频和句子交互,存储并检索视觉和文本内容.

Donahue 等人提出了长期循环卷积神经网络[85](Long-Term Recurrent Convolutional Networks,LTRCN),将 LSTM 与卷积神经网络相结合,适用于端到端的训练大规模视觉学习,比如视频识别、图像文字描述等任务.该模型可以将视频帧输入映射到自然语言文本输出,并且可以模拟复杂的时间暂态关系,实现视频内容文字描述的功能.

视频问题回答. Miller 等人在动态记忆网络的基础上,增加了一个新的图像输入模块,使模型能够回答视觉问题[44]. Ma 等人使用记忆增强的神经网络预测视觉问题的答案[86].

Kim 等人同样将记忆网络用于视频内容问题回答任务,提出了深度嵌入记忆网络[87],通过学习大量的卡通视频使 AI 智能体能够完成视频故事问答的任务.

图像识别. Moniz 等人提出卷积残差记忆网络[88],将卷积残差网络与 LSTM 相结合,用于实现图像识别的深度卷积网络.与没有存储机制的类似深度残差网络相比,具有更少深度,计算量也更少.

场景标记(Scene labeling)可以看作序列预测任务,为此 Abdulnabi 等人提出了一种基于注意力的情境记忆网络[89],由 CNN 和基于注意力机制的存储器模块组成,能够有效利用上下文关系来提高场景局部分类的准确性.

Kaiser 等人提出了一个可用于终身学习(lifelong)的长期记忆模块[90],可以添加到不同的深度学习模型,提供单样本(one-shot)学习的功能.无论是在图像分类上,实现的简单卷积模型,还是深度序列到序列循环卷积模型,使用这个长期记忆模块增强的网络都能够记住单样本并进行终身学习.

图像生成和超分辨率. Parmar 等人将基于注意力机制的变送器模型推广到图像生成的序列建模[91],证明了变送器可以对文本以外的模态进行操作,并且在 ImageNet 图像生成建模、图像类调节和图像超分辨率任务中都取得了最好的效果.

视觉显著性预测. Fernando 等人提出了记忆增强的条件生成对抗网络(Memory Augmented Conditional Generative Adversarial Networks,MC-GAN),用于特定视觉显著性预测任务[92].该模型利用具有记忆结构的条件生成对抗性网络的语义建模能力,获取主体行为模式和任务相关的因素,能够同

时学习不同任务之间的上下文语义和关系.

4.3 语音处理

语义标注. Kim 等人将记忆网络应用于语音对话领域，提出了一种对讲话者敏感的对偶存储器网络[93]（Speaker-Sensitive Dual Memory Networks，SSDMN），用于多轮语义槽标注任务（multi-turn slot tagging）. 该模型不仅将用户过去说话的每个单词编码并存储在存储器中，还生成一个单独的存储器，对系统问题的目标语义槽进行编码并保存，避免解析出嘈杂的自然语言对话.

音乐的生成. Huang 等人将变送器进行了改进，提出了相对位置的自注意力机制[94]，使模型能够捕获局部特征和位置信息，因此实现了音乐的生成建模，这同时大大地拓展了变送器的应用领域.

文本到语音（Text-To-Speech，TTS）. Li 等人提出基于 Tacotron2 和变送器的 TTS 模型[95]，可以生成接近人类录制的音频样本，并且能够进行并行训练和学习远程依赖性，从而加快训练速度，使音频韵律更加流畅.

4.4 其他方面的应用

Parisotto 等人将记忆网络与深度强化学习结合[96]，存储器采用二维空间结构，并能够实现稀疏写入. 写入存储器的记忆内容与智能体在环境中的当前位置相对应. 该模型能够让智能体使用强化学习技术，完成 2D 迷宫任务.

Baskar 等人将记忆网络与残差网络结合，提出了双向残差记忆网络结构[97]，使用具有残差和时间延迟连接的深度前馈层来模拟短时依赖性，能够以较低的计算复杂度获取过去和未来的信息.

Bornschein 等人将记忆网络与生成式模型结合，提出了具有外部存储器的变分自编码器[98]. 该模型将具有随机寻址的存储器模块的输出看作条件混合分布，其中读取操作对应于对离散存储器地址进行采样并检索相应内容的操作. 因此可以将变分推理[99]应用于存储器寻址，有助于模型通过推理执行精确的内存查找过程.

Pham 等人提出了 DeepCare[100]，将深度动态记忆网络应用于医学预测，可以通过读取病历、记忆疾病轨迹和护理过程，以及存储病史等来推断当前的疾病状况并预测未来的治疗结果. 这样的模型能够模拟疾病发展过程，支持治疗方案的制定，并提供电子病历的预后效果预测，还能够估计糖尿病和精神疾病患者的计划外再入院率.

Prakash 等人将记忆网络应用于临床诊断，提出了压缩记忆神经网络[101]（Condensed Memory Neural Networks，CMNN）. 模型将维基百科的原始文本作为知识源，使用电子健康记录中的医学笔记来进行推理，能够预测在复杂的临床情况下，最可能的诊断结果.

Schwaller 等人将药物化学中的有机合成反应看作反应物、试剂和产品之间的机器翻译问题，提出了多头注意力机制的分子变送器[102]（Molecular Transformer，MT）. 该算法不需要手工制作规则，并能够准确预测细微的化学转化，还可以准确地估计其自身的不确定性.

记忆网络还可以应用到网络安全方面，Woodbridge 等人提出了基于 LSTM 的预测域生成算法（Domain Generate Algorithm，DGA）[103]，可以对 DGA 进行实时预测，并准确地完成多分类，从而由 DGA 生成的域查找到特定的恶意软件.

Lee 等人以新皮质和海马的互补学习系统的神经认知理论为基础，提出了一种用于深度神经网络的双存储器结构[104]，使用可穿戴设备从日常生活中的持续用户行为中学习，从而理解人类在现实生活中的认知行为.

Fernando 等人提出了一种树状记忆网络[105]，用于在序列到序列映射学习问题中，联合建模序列中多个序列与短期关系之间的长期关系. 并在飞机轨迹建模和行人轨迹建模两个问题中证明了该模型的有效性和灵活性.

多视图序列学习中，不同视图之间存在一致性和互补性的两种交互作用. 因此，Zadeh 等人提出了一种新的多视图序列学习神经网络结构，称为记忆融合网络[106]，明确地考虑了这两种交互作用，并随着时间的推移不断地对它们进行增量建模. 该模型在基准数据集上的结果优于所有现有的多视图学习方法.

业务流程的灵活执行涉及多个关键决策，包括下一步要执行的任务以及分配给该任务的资源. 因此 Khan 等人将记忆增强的神经网络模型应用到流程分析预测[107]，可以减少模型参数的同时实现很好的预测效果，并为企业特定流程分析应用，提供很好的模型.

由于记忆网络的外部存储器是非结构化的，不能编码结构化对象. 因此 Pham 等人提出了关系动态记忆网络[108]，其存储器结构为多关系图，使用这样的存储器来表示结构化数据并进行操作，可以预测各种重要问题，比如软件源代码漏洞、分子生物活

性和化学成分相互作用等.

知识追踪(Knowledge Tracing)可以跟踪学生在参与学习活动时对一个或多个概念的不断发展的知识状态. Zhang 等人因此提出了动态键-值记忆网络[109](Dynamic Key-Value Memory Networks, DKV-MemNN)可以利用基础概念之间的关系,直接输出学生对每个概念的掌握程度. 该模型可以在在线学习平台中部署,以提高学生的学习效率.

神经网络在处理数据时是逐渐将收集的信息整合到模型权重中,需要非常低的学习率,如果训练数据的概率分布发生变化,那么神经网络权重就会花很长的时间才会适应这种变化. 因此,Sprechmann 等人提出了用于参数自适应的记忆网络[110],将示例存储在存储器中,然后使用基于上下文的查找过程来局部修改神经网络的权重. 这种方法在多种渐进式终身学习环境中适用于各种监督学习任务,比如图像分类和语言建模等,并改善了连续学习的性能.

4.5 小结与分析

本节主要介绍了记忆网络目前的主要应用领域以及一些小众的应用.

如表 9 所示,在应用最为广泛的自然语言处理领域,记忆网络可以完成问答、主题标签推荐、真值发现、机器阅读、嵌入表示学习、机器翻译、人机对话、顺序推荐、情感分类、语义表示等任务.

表 9 记忆网络在各个领域的应用

领域	内容	模型
自然语言处理	问答任务	MN、Skip-Thought MN、DMN、KV-MemNN、匹配记忆循环网络
	主题标签推荐	具有分层注意力机制的 MemN2N
	真值发现	用于真值发现的 MN
	机器阅读	用于机器阅读的 LSTM、多层嵌入记忆网络
	嵌入表示学习	用于深度句子嵌入表示学习的 LSTM、基于存储器的单词嵌入式表示学习
	机器翻译	变送器、WTN、ANN、Evolved Transformer、外部记忆增强的 RNN
	人机对话	基于知识库的 MemN2N
	顺序推荐	增强键-值记忆网络
	情感分类	深度记忆网络、具有存储器的 RAN
	语义表示	乘法树结构的 LSTM
计算机视觉	图像标注和图像内容描述	用于图像描述的 LSTM、KV-MemNN、CSMN、LTRCN
	视频问题回答	DMN、记忆增强的神经网络、深度嵌入记忆网络
	图像识别	卷积残差记忆网络
	视觉显著性预测	MC-GAN
语音处理	语义标注	SSDMN
	音乐生成	Music Transformer
	文本到语音	基于变送器的 TTS 模型
其他领域	2D 迷宫任务	记忆网络与深度强化学习结合
	医学预测	DeepCare
	临床诊断	CMNN
	化学反应预测	Molecular Transformer
	预测域生成算法	基于 LSTM 的 DGA
	人类行为认知	双存储器结构神经网络
	业务流程分析预测	记忆增强的神经网络
	知识追踪	DKV-MemNN
	参数自适应	用于参数自适应的记忆网络

在计算机视觉领域,记忆网络可以完成图像标注与图像内容描述、视频问答、图像识别、视觉显著性检测等任务.

在应用较少的语音处理领域,记忆网络可以完成语义标注和音乐生成的任务.

除了上述机器学习的三大应用领域,记忆网络在很多小众的领域也有出其不意的效果,比如 2D 迷宫任务、医学预测、临床诊断、预测域生成算法、人类行为认知、业务流程分析、知识追踪、参数自适应等领域.

可以看到,记忆网络现有的应用主要还是集中在自然语言处理,未来经过众多研究人员的不断探索和挖掘,记忆网络也将具有更为广泛的应用.

5 总结及未来趋势与展望

5.1 总 结

记忆网络是近年来机器学习领域中的新生方向,这一方向与传统的机器学习方法的不同在于记忆网络能够使用独立的存储器使有用的信息保存下

来，从而完成需要记忆的任务和推理，这一方向具有可期的研究前景，蕴含着巨大的潜在效益.

文章首先指出了记忆网络的发展过程，以及记忆网络的具体定义，并对具有历史意义的与记忆网络相关的学习模型：RNN、LSTM、NTM、MN 和变送器进行了介绍.

在第二部分，我们详细介绍了记忆网络的发展现状，目前记忆网络实现记忆功能的方法都各有不同，比如 RNN 是通过每个状态与前一状态之间的关系来保留部分过去的信息；LSTM 是通过遗忘门和记忆门来对重要的信息进行处理；NTM 使用图灵机模型来增强神经网络的记忆能力；MN 则使用了独立的外部存储器进行信息的记忆；而变送器抛弃了传统的 RNN 结构，仅使用注意力机制来权衡输入信息对当前输出的重要程度.因此，我们按照每个模型实现记忆功能机制的不同类别，对更多的记忆网络模型进行分类介绍和对比.

除此之外，还有在此基础上新提出的记忆网络分支模型，比如动态记忆网络以及在视觉和文本问答系统当中的应用.还有键-值记忆网络，将上下文向量以键-值对的形式保存在存储器中，可以完成知识库问答和视频内容描述的任务.以及分层记忆网络，分别从存储器结构、输入文本内容结构以及二叉树模型结构三个不同的角度来实现不同情境下的记忆功能.

最后，我们对记忆网络目前的应用场景：自然语言处理、计算机视、语音处理等领域进行了分析和梳理，可以看到记忆网络现有的主要研究领域还是在自然语言处理方向，但依然可以应用于众多冷门的领域，如医学、网络安全、业务流程等.

随着记忆网络理论与方法研究的深入，记忆网络将会更为成熟，并应用于更多的机器学习场景，终将为机器学习领域做出更大的贡献.

5.2　未来趋势与展望

作为机器学习领域中的一个新兴方向，记忆网络近几年来取得了飞速的发展，因为研究人员都意识到记忆和学习相比，同样是一个复杂的过程.虽然现有的记忆网络模型都取得了令人激动的成就，但是无论是记忆网络本身的模型结构，还是记忆网络的算法以及评价标准都是值得研究的问题.而且记忆网络在各个领域的应用还只是刚刚开始，依然有许多未知的领域值得我们去探索.对于记忆网络现有的问题和未来的发展方向，我们给出以下一些自己的想法和见解：

(1) 目前来说，记忆网络如果想要实现更精确的推理，需要对记忆的内容进行更大范围的扩展.如果可以使用现有数据库中的内容，添加到记忆网络的存储器模块中，这样可以大大减少记忆网络的运算复杂性，使得记忆网络推广到更多应用领域.

(2) 记忆网络研究的关键问题是输入的查询问题与存储器中信息的匹配问题，如何定义匹配标准，如何使得匹配过程计算复杂性更低，如何能够更准确地进行信息的匹配，这影响着记忆网络的输出结果的精度.

(3) MN 在问答系统的迁移学习上有良好的表现，可见迁移学习方法在大数据系统上有应用前景.依托于 MN 的记忆和寻址能力，它将具有更强的知识迁移能力.相对于 LSTM 模型，MN 的记忆容量理论上是不受限制的，因此 MN 可以在公共知识库上进行更好的迁移学习[111].

(4) 人们通常认为遗忘是记忆的负面现象或者是记忆失效，但是遗忘的真实含义却没有完全被揭示出来.希伯来大学计算机与神经科学教授 Tishby 提出了信息瓶颈理论[112]，指出神经网络就像把信息挤出瓶颈口一样，只留下与一般概念最相关的特征，去掉大量无用的噪声数据.并且论证了遗忘能够通过减少过时信息对决策的影响提升了学习的灵活性，还阻止了对于特定过去时间记忆信息的过拟合，从而提升了模型的通用性.根据这个观点，记忆的目标不是通过时间来传递信息，而是应该把记忆看作最优化决策过程[113].因此神经网络中记忆的短暂即逝和保持同样重要.所以对于记忆网络来说，如何从存储器中选择特定信息以便遗忘，在记忆网络中如何引入遗忘机制，也是未来的发展方向.

(5) 记忆网络在未来应该探索更复杂的记忆结构，来应对更复杂的数据，比如涉及到多个动词和名词、具有多样性输出结构的句子，以及需要学习高阶因果关系或者学习时间和空间关系推理的情况.对于 MN 来说，应该使用更复杂的记忆管理机制和句子结构化表示形式进行记忆，如图结构，树结构，空间暂态结构等，存储器模块也可以考虑加入类似于 LSTM 遗忘门，用来处理无用信息的遗忘结构.

(6) 与记忆网络相对应，变送器是基于生物学机制实现记忆功能的神经网络.根据当前的研究进展[23]，将循环结构加入变送器弥补了它自身不能捕获序列依赖关系的缺点.因此记忆网络与变送器具有互补性，将二者进行更加深入的研究和结合也是记忆网络未来的发展方向.

(7)从本文可以看到很多与记忆网络结合的神经网络模型,特别是记忆能力较弱的 RNN 和 LSTM.这些模型的提出不仅完善了记忆网络,而且在此基础上涌现出各种各样的应用.因此,将记忆网络与现有的其他神经网络进行融合创新,不仅能提高原有神经网络模型的记忆能力,还能丰富记忆网络的应用种类和领域.

(8)现有的记忆网络模型性能评价都是基于数据集或者应用领域的评价标准,并没有统一用来衡量模型记忆能力的评估方法,因此只能在特定数据集上对多个记忆网络模型进行比较.所以,未来记忆网络模型的统一评估方法亟待建立和完善,这样有利于方便地比较不同模型的记忆能力.

(9)记忆网络的主要应用是在自然语言处理领域中,但是由于语言的多义性,如何在对自然语言处理的过程中正确地推断出真实表达的含义,这也是需要解决的问题之一.

(10)在计算机视觉领域,记忆网络的主要应用还是在一些与文本有关的内容,比如视频问答、字幕生成、视频内容描述等.因此,可以将记忆网络与 CNN 结合用于更多的计算机视觉领域,甚至可以与当前最新的 GAN 相结合[90],获得更好的处理图像和视频的能力.

(11)在语音处理领域,记忆网络的应用还不够丰富,因此,未来可能更多地将记忆网络与现有的语音处理模型相结合,使语音处理模型具有更好的记忆能力,从而获得更好的性能.

(12)随着这两年机器学习领域的快速发展,人工智能也不断走进我们的生活,比如智能家居、语言助手、无人驾驶等,人工智能的发展离不开机器学习模型的记忆能力.因此,如何加速人工智能模型与记忆网络的融合,让记忆网络的优势在人工智能领域充分发挥出来,这也是记忆网络未来的发展方向.

(13)记忆网络已经开始应用于各个行业,比如医学[100-102]、网络安全[103]、业务流程分析[107]等等,可以说记忆网络将会在各行各业崭露头角.但是,记忆网络在这些行业的应用才刚刚起步,因此如何改进记忆网络来适应这些不同的行业应用,也是记忆网络未来的发展方向.

(14)记忆网络还可以应用于工业生产,比如机器设备的故障诊断预测.将设备过去的工作状态和故障状态作为记忆网络的记忆,以设备当前的运行参数作为序列输入,通过记忆网络对当前输入和过去运行状态进行分析推理,可以在设备发生故障之前就进行预测,并紧急停车进行检修.这一应用若能成功实现,将会为工业生产领域带来巨大的经济效益.

(15)记忆网络研究的问题本质上是如何选择参与到预测输出的输入特征问题,既要考虑到过往收集到的有用信息,同时,又不能引入过大的存储复杂性和访问复杂性,应该开发自适应调度策略,根据学习场景的变化,按信息收集的时间粒度和信息的有用性粒度,动态地分层次存储信息,使得很快地就能定位到对当前输出最有用的特征上.存储单元的大小应能动态变化,存储的信息也要定时的调整更新,不但存储以前学习到的特征,也要存储以前学习到的结果,像人一样,根据以往学习的结果,不用太多的学习,就能很快地识别和推理出想要的预测结果.

(16)人类的记忆具有联想、类比、匹配、排序等功能,而且,不限定学习到的结果的使用范围.不一定学习到的文字理解,就不能用于图像内容理解;人类到了一个陌生的环境,可以根据以往到过的地方,大致事先给出环境可能包含的物体和物体之间关系的大致轮廓,如何使得记忆网络能够记住抽象的概念,以及抽象的概念之间的可能关系组合,而不单单是低层次的特征,而且这些概念以什么形式存储,都是值得认真思考的问题.

从以上的分析可以看出,记忆网络的发展越来越快,应用领域也越来越广泛,并且在各个领域的模型种类丰富多样.这需要我们及时的关注最新的记忆网络发展情况,总结和分析已有的研究内容,为以后记忆网络的研究提供参考.

参 考 文 献

[1] Hochreiter S, Schmidhuber J. Long short-term memory. Neural Computation, 1997, 9(8): 1735-1780

[2] Weston J, Chopra S, Bordes A. Memory networks//Proceedings of the 3rd International Conference on Learning Representations. San Diego, USA, 2015: 23-32

[3] Graves A, Wayne G, Danihelka I. Neural turing machines. arXiv preprint arXiv:1410.5401, 2014

[4] Sukhbaatar S, Szlam A, Weston J, Fergus R. End-to-end memory networks//Proceedings of the Annual Conference on Neural Information Processing Systems. Montreal, Canada, 2015: 2440-2448

[5] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need//Proceedings of the Annual Conference on Neural Information Processing Systems. Long Beach, USA, 2017: 5998-6008

[6] Greff K, Srivastava R K, Koutník J, et al. LSTM: A search space odyssey. IEEE Transactions on Neural Networks and Learning Systems, 2017, 28(10): 2222-2232

[7] Fader A, Zettlemoyer L, Etzioni O. Paraphrase-driven learning for open question answering//Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics. Sofia, Bulgaria, 2013: 1608-1618

[8] Bordes A, Weston J, Usunier N. Open question answering with weakly supervised embedding models//Proceedings of the Machine Learning and Knowledge Discovery in Databases-European Conference. Nancy, France, 2014: 165-180

[9] Mikolov T, Karafiát M, Burget L, et al. Recurrent neural network based language model//Proceedings of the 11th Annual Conference of the International Speech Communication Association. Makuhari, Japan, 2010: 1045-1048

[10] Kingma D P, Ba J. Adam: A method for stochastic optimization //Proceedings of the 3rd International Conference on Learning Representations. San Diego, USA, 2015: 56-65

[11] Srivastava N, Hinton G, Krizhevsky A, et al. Dropout: A simple way to prevent neural networks from overfitting. The Journal of Machine Learning Research, 2014, 15(1): 1929-1958

[12] Szegedy C, Vanhoucke V, Ioffe S, et al. Rethinking the inception architecture for computer vision//Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition. Las Vegas, USA, 2016: 2818-2826

[13] Kalchbrenner N, Espeholt L, Simonyan K, et al. Neural machine translation in linear time. arXiv preprint arXiv: 1610.10099, 2016

[14] Zhou J, Cao Y, Wang X, et al. Deep recurrent models with fast-forward connections for neural machine translation. Transactions of the Association for Computational Linguistics, 2016, 4: 371-383

[15] Wu Y, Schuster M, Chen Z, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. arXiv preprint arXiv:1609.08144, 2016

[16] Gehring J, Auli M, Grangier D, et al. Convolutional sequence to sequence learning//Proceedings of the 34th International Conference on Machine Learning. Sydney, Australia, 2017: 1243-1252

[17] Shazeer N, Mirhoseini A, Maziarz K, et al. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer//Proceedings of the 5th International Conference on Learning Representations. Toulon, France, 2017: 101-109

[18] Li J, Chen Y, Cai L, et al. Dense transformer networks. arXiv preprint arXiv:1705.08881, 2017

[19] Ahmed K, Keskar N S, Socher R. Weighted transformer network for machine translation. arXiv preprint arXiv:1711.02132, 2017

[20] Zhang B, Xiong D, Su J. Accelerating neural transformer via an average attention network//Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics. Melbourne, Australia, 2018: 1789-1798

[21] Chia Y K, Witteveen S, Andrews M. Transformer to CNN: Label-scarce distillation for efficient text classification. arXiv preprint arXiv:1909.03508, 2019

[22] Girdhar R, Carreira J, Doersch C, et al. Video action transformer network//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Long Beach, USA, 2019: 244-253

[23] Dehghani M, Gouws S, Vinyals O, et al. Universal transformers //Proceedings of the 7th International Conference on Learning Representations. New Orleans, USA, 2019: 35-43

[24] So D R, Liang C, Le Q V. The evolved transformer// Proceedings of the 36th International Conference on Machine Learning. Long Beach, USA, 2019: 5877-5886

[25] Lee J, Lee Y, Kim J, et al. Set transformer: A framework for attention-based permutation-invariant neural networks// Proceedings of the 36th International Conference on Machine Learning. Long Beach, USA, 2019: 3744-3753

[26] Dai Z, Yang Z, Yang Y, et al. Transformer-XL: Attentive language models beyond a fixed-length context//Proceedings of the 57th Conference of the Association for Computational Linguistics. Florence, Italy, 2019: 2978-2988

[27] Liu Da-Rong, Chuang Shun-Po, Lee Hung-Yi. Attention-based memory selection recurrent network for language modeling. arXiv preprint arXiv:1611.08656, 2016

[28] Taylor A, Marcus M, Santorini B. The Penn Treebank: An overview//Treebanks. Springer, Dordrecht, 2003: 5-22

[29] Godfrey J J, Holliman E C, McDaniel J. SWITCHBOARD: Telephone speech corpus for research and development// Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing. San Francisco, USA, 1992, 1: 517-520

[30] Graff D, Chen K. Chinese gigaword. LDC Catalog, 2005, 1: 58563-58230

[31] Tran K, Bisazza A, Monz C. Recurrent memory networks for language modeling//Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego, USA, 2016: 321-331

[32] Mikolov T, Joulin A, Chopra S, et al. Learning longer memory in recurrent neural networks//Proceedings of the 3rd International Conference on Learning Representations. San Diego, USA, 2015

[33] Santoro A, Bartunov S, Botvinick M, et al. One-shot learning with memory-augmented neural networks. arXiv preprint arXiv:1605.06065, 2016

[34] Wang Jianyong, Zhang Lei, Guo Quan, Yi Zhang. Recurrent neural networks with auxiliary memory units. IEEE Transactions on Neural Networks and Learning Systems, 2018, 29(5): 1652-1661

[35] Goodfellow I, Bengio Y, Courville A. Deep Learning: Adaptive Computation and Machine Learning. Cambridge, MA: MIT Press, 2016

[36] Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks//Proceedings of the 30th International Conference on Machine Learning. Atlanta, USA, 2013: 1310-1318

[37] Danihelka I, Wayne G, Uria B, et al. Associative long short-term memory//Proceedings of the 33rd International Conference on Machine Learning. New York City, USA, 2016: 1986-1994

[38] Zhang Xingxing, Lu Liang, Lapata M. Top-down tree long short-term memory networks//Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego, USA, 2016: 310-320

[39] Graves A, Schmidhuber J. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. Neural Networks, 2005, 18(5-6): 602-610

[40] Cheng Jianpeng, Dong Li, Lapata M. Long short-term memory-networks for machine reading//Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Austin, USA, 2016: 551-561

[41] Vinyals O, Blundell C, Lillicrap T, et al. Matching networks for one shot learning//Proceedings of the Annual Conference on Neural Information Processing Systems. Barcelona, Spain, 2016: 3630-3638

[42] Liu Fei, Perez J. Gated end-to-end memory networks//Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics. Valencia, Spain, 2017: 1-10

[43] Kumar A, Irsoy O, Ondruska P, et al. Ask me anything: Dynamic memory networks for natural language processing//Proceedings of the 33rd International Conference on Machine Learning. New York City, USA, 2016: 1378-1387

[44] Miller A H, Fisch A, Dodge J, et al. Key-value memory networks for directly reading documents//Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Austin, USA, 2016: 1400-1409

[45] Chandar S, Ahn S, Larochelle H, et al. Hierarchical memory networks. arXiv preprint arXiv:1605.07427, 2016

[46] Auvolat A, Vincent P. Clustering is efficient for approximate maximum inner product search. arXiv preprint arXiv: 1507.05910, 2015

[47] Zhong S. Efficient online spherical k-means clustering//Proceedings of the International Joint Conference on Neural Networks. Montreal, Canada, 2005, 5: 3180-3185

[48] Andrychowicz M, Kurach K. Learning efficient algorithms with hierarchical attentive memory. arXiv preprint arXiv: 1602.03218, 2016

[49] Williams R J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine Learning, 1992, 8(3-4): 229-256

[50] Bordes A, Usunier N, Chopra S, Weston J. Large-scale simple question answering with memory networks. arXiv preprint arXiv:1506.02075, 2015

[51] Xiong C, Merity S, Socher R. Dynamic memory networks for visual and textual question answering//Proceedings of the 33rd International Conference on Machine Learning. New York City, USA, 2016: 2397-2406

[52] Ramachandran G S, Sohmshetty A. Ask me even more: Dynamic memory tensor networks (extended model). arXiv preprint arXiv:1703.03939, 2017

[53] Socher R, Chen D, Manning C D, Ng A Y. Reasoning with neural tensor networks for knowledge base completion//Proceedings of the 27th Annual Conference on Neural Information Processing Systems. Lake Tahoe, USA, 2013: 926-934

[54] Jain A K, Agarwalla A, Agrawal K K, Mitra P. Recurrent memory addressing for describing videos//Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops. Honolulu, USA, 2017: 2200-2207

[55] Xu Jiaming, Shi Jing, Yao Yiqun, et al. Hierarchical memory networks for answer selection on unknown words//Proceedings of the 26th International Conference on Computational Linguistics. Osaka, Japan, 2016: 2290-2299

[56] Vinyals O, Fortunato M, Jaitly N. Pointer networks//Proceedings of the Annual Conference on Neural Information Processing Systems. Montreal, Canada, 2015: 2692-2700

[57] Bottou L. Stochastic gradient learning in neural networks. Proceedings of Neuro-Nımes, 1991, 91(8): 12

[58] Furlanello T, Zhao J, Saxe A M, et al. Active long term memory networks. arXiv preprint arXiv:1606.02355, 2016

[59] Hinton G E, Vinyals O, Dean J. Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531, 2015

[60] McClelland J L, McNaughton B L, O'Reilly R C. Why there are complementary learning systems in the hippocampus and neocortex: Insights from the successes and failures of connectionist models of learning and memory. Psychological Review, 1995, 102(3): 419

[61] Raffel C, Ellis D P W. Feed-forward networks with attention can solve some long-term memory problems. arXiv preprint arXiv:1512.08756, 2015

[62] 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, USA, 2015: 200-208

[63] Cho K, Van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation//Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing. Doha, Qatar, 2014: 1724-1734

[64] Gülçehre Ç, Chandar S, Bengio Y. Memory augmented neural networks with wormhole connections. arXiv preprint arXiv:1701.08718, 2017

[65] Jain S. Question answering over knowledge base using factual memory networks//Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. San Diego, USA, 2016: 109-115

[66] Rae J W, Hunt J J, Danihelka I, et al. Scaling memory-augmented neural networks with sparse reads and writes//Proceedings of the Annual Conference on Neural Information Processing Systems. Barcelona, Spain, 2016: 3621-3629

[67] Zhang Wei, Yu Yang, Zhou Bowen. Structured memory for neural turing machines. arXiv preprint arXiv: 1510.03931, 2015

[68] Samothrakis S, Vodopivec T, Fasli M, Fairbank M. Match memory recurrent networks//Proceedings of the 2016 International Joint Conference on Neural Networks. Vancouver, Canada, 2016: 1339-1346

[69] Kiros R, Zhu Y, Salakhutdinov R, et al. Skip-thought vectors//Proceedings of the Annual Conference on Neural Information Processing System. Montreal, Canada, 2015: 3294-3302

[70] Caballero E. Skip-thought memory networks. arXiv preprint arXiv:1511.06420, 2015

[71] Huang Haoran, Zhang Qi, Gong Yeyun, Huang Xuanjing. Hashtag recommendation using end-to-end memory networks with hierarchical attention//Proceedings of the 26th International Conference on Computational Linguistics. Osaka, Japan, 2016: 943-952

[72] Li Luyang, Qin Bing, Ren Wenjing, Liu Ting. Truth discovery with memory network. Tsinghua Science and Technology, 2017, 22(6): 609-618

[73] Pan Boyuan, Li Hao, Zhao Zhou, et al. MEMEN: Multi-layer embedding with memory networks for machine comprehension. arXiv preprint arXiv:1707.09098, 2017

[74] Palangi H, Deng L, Shen Y, et al. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. IEEE/ACM Transactions on Audio, Speech, and Language Processing, 2016, 24(4): 694-707

[75] Sun Jingyuan, Wang Shaonan, Zong Chengqing. Memory, show the way: Memory based few shot word representation learning//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. Brussels, Belgium, 2018: 1435-1444

[76] Wang Mingxuan, Lu Zhengdong, Li Hang, Liu Qun. Memory-enhanced decoder for neural machine translation//Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Austin, USA, 2016: 278-286

[77] Ganhotra J, Polymenakos L. Knowledge-based end-to-end memory networks. arXiv preprint arXiv:1804.08204, 2018

[78] Huang J, Zhao W X, Dou H, et al. Improving sequential recommendation with knowledge-enhanced memory networks //Proceedings of the 41st International ACM SIGIR Conference on Research & Development in Information Retrieval. Ann Arbor, USA, 2018: 505-514

[79] Tang Duyu, Qin Bing, Liu Ting. Aspect level sentiment classification with deep memory network//Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing. Austin, USA, 2016: 214-224

[80] Chen Peng, Sun Zhongqian, Bing Lidong, Yang Wei. Recurrent attention network on memory for aspect sentiment analysis//Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing. Copenhagen, Denmark, 2017: 452-461

[81] Tran N K, Cheng W. Multiplicative tree-structured long short-term memory networks for semantic representations//Proceedings of the 7th Joint Conference on Lexical and Computational Semantics. New Orleans, USA, 2018: 276-286

[82] Jia X, Gavves E, Fernando B, Tuytelaars T. Guiding the long-short term memory model for image caption generation //Proceedings of the 2015 IEEE International Conference on Computer Vision. Santiago, Chile, 2015: 2407-2415

[83] Chunseong Park C, Kim B, Kim G. Attend to you: Personalized image captioning with context sequence memory networks//Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition. Honolulu, USA, 2017: 6432-6440

[84] Wang Junbo, Wang Wei, Huang Yan, et al. M3: Multimodal memory modelling for video captioning//Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 7512-7520

[85] Donahue J, Anne Hendricks L, Guadarrama S, et al. Long-term recurrent convolutional networks for visual recognition and description//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Boston, USA, 2015: 2625-2634

[86] Ma Chao, Shen Chunhua, Dick A R, et al. Visual question answering with memory-augmented networks//Proceedings of the 2018 IEEE Conference on Computer Vision and Pattern Recognition. Salt Lake City, USA, 2018: 6975-6984

[87] Kim Kyung-Min, Heo Min-Oh, Choi Seong-Ho, Zhang Byoung-Tak. DeepStory: Video story QA by deep embedded memory networks//Proceedings of the 26th International Joint Conference on Artificial Intelligence. Melbourne, Australia, 2017: 2016-2022

[88] Moniz J, Pal C J. Convolutional residual memory networks. arXiv preprint arXiv:1606.05262, 2016

[89] Abdulnabi A H, Shuai B, Winkler S, Wang G. Episodic CAMN: Contextual attention-based memory networks with iterative feedback for scene labeling//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. Honolulu, USA, 2017: 6278-6287

[90] Kaiser L, Nachum O, Roy A, Bengio S. Learning to remember rare events//Proceedings of the 5th International Conference on Learning Representations. Toulon, France, 2017: 201-209

[91] Parmar N, Vaswani A, Uszkoreit J, et al. Image transformer //Proceedings of the 35th International Conference on Machine Learning. Stockholmsmässan, Sweden, 2018: 4052-4061

[92] Fernando T, Denman S, Sridharan S, Fookes C. Task specific visual saliency prediction with memory augmented conditional generative adversarial networks//Proceedings of

the 2018 IEEE Winter Conference on Applications of Computer Vision. Lake Tahoe, USA, 2018: 1539-1548

[93] Kim Young-Bum, Lee Sungjin, Sarikaya R. Speaker-sensitive dual memory networks for multi-turn slot tagging//Proceedings of the 2017 IEEE Automatic Speech Recognition and Understanding Workshop, Okinawa, Japan, 2017: 541-546

[94] Huang C Z A, Vaswani A, Uszkoreit J, et al. Music transformer: Generating music with long-term structure//Proceedings of the 7th International Conference on Learning Representations. New Orleans, USA, 2019: 205-213

[95] Li N, Liu S, Liu Y, et al. Close to human quality TTS with transformer. arXiv preprint arXiv:1809.08895, 2018

[96] Parisotto E, Salakhutdinov R. Neural map: Structured memory for deep reinforcement learning//Proceedings of the 6th International Conference on Learning Representations. Vancouver, Canada, 2018: 115-123

[97] Baskar M K, Karafiát M, Burget L, et al. Residual memory networks: Feed-forward approach to learn long-term temporal dependencies//Proceedings of the 2017 IEEE International Conference on Acoustics, Speech and Signal Processing. New Orleans, USA, 2017: 4810-4814

[98] Bornschein J, Mnih A, Zoran D, Rezende D J. Variational memory addressing in generative models//Proceedings of the Annual Conference on Neural Information Processing Systems. Long Beach, USA, 2017: 3923-3932

[99] Kingma D P, Welling M. Auto-encoding variational Bayes//Proceedings of the 2nd International Conference on Learning Representations. Banff, Canada, 2014

[100] Pham T, Tran T, Phung D Q, Venkatesh S. DeepCare: A deep dynamic memory model for predictive medicine//Proceedings of the Advances in Knowledge Discovery and Data Mining-20th Pacific-Asia Conference. Auckland, New Zealand, 2016: 30-41

[101] Prakash A, Zhao S, Hasan S A, et al. Condensed memory networks for clinical diagnostic inferencing//Proceedings of the 31st AAAI Conference on Artificial Intelligence. San Francisco, USA, 2017: 3274-3280

[102] Schwaller P, Laino T, Gaudin T, et al. Molecular transformer for chemical reaction prediction and uncertainty estimation. arXiv preprint arXiv:1811.02633, 2018

[103] Woodbridge J, Anderson H S, Ahuja A, Grant D. Predicting domain generation algorithms with long short-term memory networks. arXiv preprint arXiv:1611.00791, 2016

[104] Lee Sang-Woo, Lee Chung-Yeon, Kwak Dong-Hyun, et al. Dual-memory neural networks for modeling cognitive activities of humans via wearable sensors. Neural Networks, 2017, 92: 17-28

[105] Fernando T, Denman S, McFadyen A, et al. Tree memory networks for modelling long-term temporal dependencies. Neurocomputing, 2018, 304: 64-81

[106] Zadeh A, Liang P P, Mazumder N, et al. Memory fusion network for multi-view sequential learning//Proceedings of the 32nd AAAI Conference on Artificial Intelligence. New Orleans, USA, 2018: 5634-5641

[107] Khan M A, Le H, Do K, et al. Memory-augmented neural networks for predictive process analytics. arXiv preprint arXiv:1802.00938, 2018

[108] Pham T, Tran T, Venkatesh S. Relational dynamic memory networks. arXiv preprint arXiv:1808.04247, 2018

[109] Zhang J, Shi X, King I, et al. Dynamic key-value memory networks for knowledge tracing//Proceedings of the 26th International Conference on World Wide Web. Perth, Australia, 2017: 765-774

[110] Sprechmann P, Jayakumar S M, Rae J W, et al. Memory-based parameter adaptation//Proceedings of the 6th International Conference on Learning Representations. Vancouver, Canada, 2018: 113-121

[111] Liang Tian-Xin, Yang Xiao-Ping, Wang Liang, et al. Review on research and development of memory neural networks. Journal of Software, 2017, 28(11): 2905-2924 (in Chinese)
(梁天新, 杨小平, 王良等. 记忆神经网络的研究与发展. 软件学报, 2017, 28(11): 2905-2924)

[112] Tishby N, Zaslavsky N. Deep learning and the information bottleneck principle//Proceedings of the 2015 IEEE Information Theory Workshop. Jerusalem, Israel, 2015: 1-5

[113] Yang Qing-Feng, Wu Meng-Qiu. Memory philosophy: The key to decoding artificial intelligence and its development. Exploration and Free Views, 2018, 1(11): 86(in Chinese)
(杨庆峰, 伍梦秋. 记忆哲学: 解码人工智能及其发展的钥匙. 探索与争鸣, 2018, 1(11): 86)

LIU Jian-Wei, Ph. D., associate professor. His main research interests include machine learning, intelligent information processing, analysis, prediction, controlling of complicated system, and analysis of the algorithm and the designing.

WANG Yuan-Fang, M. S. candidate. His main research interest is machine learning.

LUO Xiong-Lin, Ph. D., professor. His main research interests include intelligent control, and analysis, prediction, controlling of complicated system.

Background

Deep memory network is a general term for neural network models with memory function, which is mainly to solve the prediction problem of sequence-dependent dependence, and can be predicted by memorizing the effective information learned before. Memory network usually have independent memory modules or other structures capable of memory function. The former stores important information in an independently readable and writable memory and reads it when needed; while the latter method usually modify the internal structure of the cell to retain the information that needs to be remembered.

Deep memory network have achieved unprecedented performance in a wide variety of different application areas. For example, image classification, face recognition, human-level concept learning, playing Atari games and AlphaGo.

Deep memory network combines the benefits of memory network and deep learning. On one hand, memory network has a wider scope of applicability since it can enhance the memory of the model. On the other hand, deep learning can extract a good representation at different levels of abstraction, which disentangles better the factors of variations underlying the data.

In this paper we aim to survey and place a number of issues related to deep memory network in a broad context, and compare the advantages and disadvantages of different memory methods. For the most basic memory network models RNN, LSTM, NTM, MN and transformer, we summarize and compare their prons and cons.

In this paper, we give a systematical survey of the deep memory network models. We point out the advantages, disadvantages and application scenes of different memory networks. Finally, we give a discussion on open issues related to memory networks.

This work is supported by the Science Foundation of China University of Petroleum Beijing (2462018QZDX02).