Computer Science and Application 计算机科学与应用, 2022, 12(12), 2813-2824
Published Online December 2022 in Hans. http://www.hanspub.org/journal/csa
https://doi.org/10.12677/csa.2022.1212286

Hans汉斯

基于深度神经网络的单词预测系统设计与实现

王　昕

同济大学电子与信息工程学院，上海

收稿日期：2022年11月15日；录用日期：2022年12月15日；发布日期：2022年12月22日

摘　要

文本生成与预测是自然语言处理中一个重要的研究领域，具有广阔的应用前景，例如通过输入法或者检索框打字时预测下一个单词或者文字。然后人们的喜好和习惯不尽相同，传统的预测方法难以有很好的预测效果。而随着深度神经网络的发展与应用，利用深度神经网络模型的文本预测系统识别准确率和速度也极大地提高。本文训练并评估了最为流行的深度神经网络预测模型，并设计了一个单词预测系统，使用前后端分离技术，前端是一个可视化网页界面，后端采用多个深度学习模型，方便评估模型效果。

关键词

单词预测，深度神经网络模型，系统

Design and Implementation of Word Prediction System Based on Deep Neural Network

Xin Wang

College of Electronic and Information Engineering, Tongji University, Shanghai

Received: Nov. 15th, 2022; accepted: Dec. 15th, 2022; published: Dec. 22nd, 2022

Abstract

Text generation and prediction is an important research field in natural language processing and has broad application prospects, such as predicting the next word or text when typing through an input method or a search box. However, people's preferences and habits are not the same, and traditional forecasting methods are difficult to have a good forecasting effect. With the development and application of deep neural networks, the recognition accuracy and speed of text predic-

文章引用: 王昕. 基于深度神经网络的单词预测系统设计与实现[J]. 计算机科学与应用, 2022, 12(12): 2813-2824.
DOI: 10.12677/csa.2022.1212286

tion systems using deep neural network models have also been greatly improved. This paper trains and evaluates the most popular deep neural network prediction model, and designs a word prediction system, using front-end and back-end separation technology. The front-end is a visual web interface, and the back-end uses multiple deep learning models to facilitate the evaluation of model effects.

Keywords

Text Prediction, Deep Neural Network Models, System

1. 引言

文本生成与预测是自然语言处理中一个重要的研究领域[1] [2]，具有广阔的应用前景。国内外已经有诸如 Automated Insights、Narrative Science 以及“小南”机器人和“小明”机器人等文本生成系统投入使用。这些系统根据格式化数据或自然语言文本生成新闻、财报或者其他解释性文本。例如，Automated Insights 的 WordSmith 技术已经被美联社等机构使用，帮助美联社报道大学橄榄球赛事、公司财报等新闻。这使得美联社不仅新闻更新速度更快，而且在人力资源不变的情况下扩大了其在公司财报方面报道的覆盖面。

2. 相关理论

2.1. 单词预测

单词预测是典型 NLP [3] [4]任务，其在生活中具有很多应用，如输入法的实时辅助输入系统，古书籍缺失文字修复等。这种预测的思路就是把整个句子看作是一个概率模型，下一个词是什么的概率是由前面的次序列所决定的。下面是整个句子产生的概率：

$$P(X)=\prod_{i=1}^{I}P(x_i \mid x_1,\cdots,x_{i-1}) \tag{1}$$

那么预测下一个单词就可以表示为：

$$\hat{x}=\arg\max_{x_i} P(x_i \mid x_1,\cdots,x_{i-1}) \tag{2}$$

2.2. LSTM

从循环神经网络 RNN [2] (Recurrent Neural Network)说起，这是一种用于处理序列数据的神经网络，常用语 NLP [3]等领域。

最普通的 RNN 主要形式如图 1 所示，这里 x 为当前状态输入，h 为接收到的上一节点输入；y 为当前状态输出，h'为传递到下一节点输出，可以看到输出 h'与 x 和 h 的值都相关。

而长短时记忆 LSTM [4] (Long short-term memory)是一种特殊的 RNN，主要为了解决长序列训练过程的梯度消失和梯度爆炸问题，即相比于普通 RNN，LSTM 能在更长的序列中有更好的表现。

LSTM 的结构及其和普通 RNN 得对比如图 2 所示。

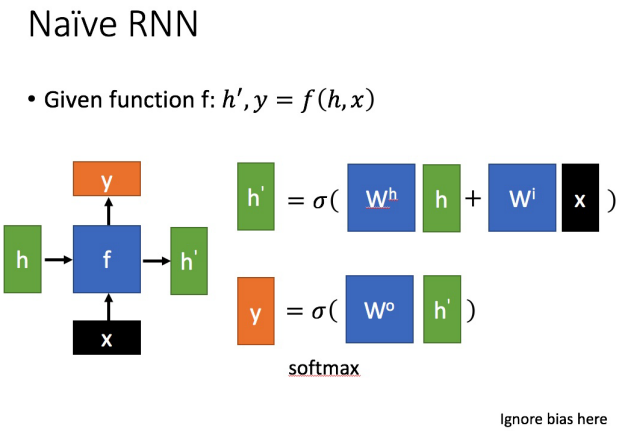


Figure 1. Schematic diagram of standard RNN structure
图 1. 标准 RNN 结构示意图

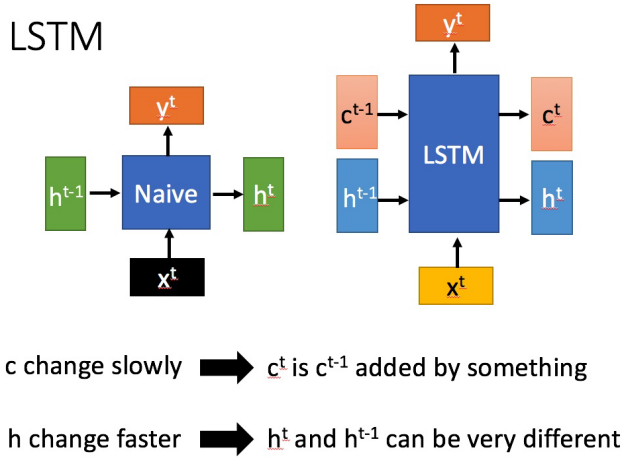
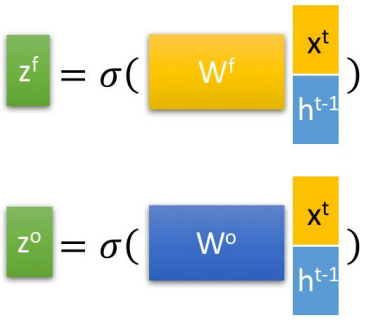


Figure 2. Schematic diagram of standard LSTM structure
图 2. 标准 LSTM 结构示意图

相比 RNN 仅有一个传递状态 h^t，LSTM 有两个传输状态 c^t (cell state)和 h^t (hidden state)。对于传递下去的 c^t 改变的很慢，通常输出的 c^t 是上一个 c^{t-1} 加上一些数值，而 h^t 在不同节点下往往有很大区别。

下面分析一下 LSTM 的内部结构，首先使用 LSTM 的当前输入和上次传递下来的值拼接得到四个状态，如下图：

其中 z^f, z^i, z^o 是由拼接向量乘以权重矩阵后，再通过一个 sigmoid 激活函数转换为 0~1 之间的数用来作为门控状态。



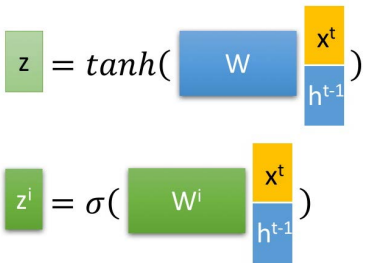


Figure 3. Schematic diagram of the internal structure of LSTM
图 3. LSTM 内部结构示意图

LSTM 图 3 最后的内部计算总共包含了三个阶段：

忘记阶段。这个阶段主要是对上一个节点传进来的输入进行选择性忘记。简单来说就是会“忘记不重要的，记住重要的”。具体来说是通过计算得到的 z^f (f 表示 forget)来作为忘记门控，来控制上一个状态的 c^{t-1} 哪些需要留哪些需要忘。

选择记忆阶段，这个阶段将这个阶段的输入有选择性地进行“记忆”。主要是会对输入 x^t 进行选择记忆。哪些重要则着重记录下来，哪些不重要，则少记一些。当前的输入内容由前面计算得到的 z 表示。而选择的门控信号则是由 z^i (i 代表 information)来进行控制。

输出阶段。这个阶段将决定哪些将会被当成当前状态的输出。主要是通过 z^o 来进行控制的。并且还对上一阶段得到的 c^o 进行了放缩(通过一个 tanh 激活函数进行变化)。

以上就是 LSTM 的内部结构，用门控来控制传输状态，记住要长时间记忆的内容，忘记不重要的信息，比 RNN 的记忆叠加方法要好很多。但也因为引入了更多的参数，使得模型训练困难，因此往往还会使用和 LSTM 效果相当但是训练更容易的 GRU 来构建大训练量的模型。

2.3. CRU

GRU [5]的输入输出结构与普通 RNN 是一样的。有一个当前的输入 x^t，和上一个节点传递下来的隐状态(hidden state) h^{t-1}，这个隐状态包含了之前节点的相关信息。下面来分析它的内部结构图 4，图 5。

首先，通过上一个传输下来的状态 h^{t-1} 和当前节点的输入 x^t 来获取两个门控状态。如下图所示，其中r控制重置的门控(reset gate)，z 为控制更新的门控(update gate)。

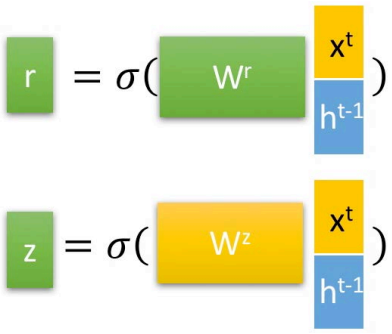


Figure 4. Schematic diagram of the internal structure of LSTM
图 4. LSTM 内部结构示意图

得到门控信号之后，首先使用重置门控来得到“重置”之后的数：

$$h^{t-1r} = h^t \odot r$$

再将 $h^{t-1\prime}$ 与输入 x^t 进行拼接，再通过一个 tanh 激活函数来将数据放缩到−1~1 的范围内。即得到如下图所示的 h'。

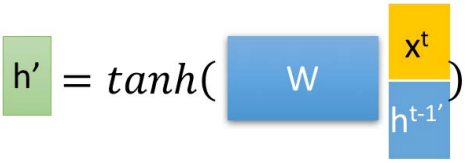


Figure 5. Schematic diagram of GRU internal activation function
图 5. GRU 内部激活函数示意图

这里的 h' 主要是包含了当前输入的 x^t 数据。有针对性地对 h' 添加到当前的隐藏状态，相当于“记忆了当前时刻的状态”。类似于 LSTM 的选择记忆阶段。

最后介绍 GRU 最关键的一个步骤，可以称之为“更新记忆”阶段。这时同时进行了遗忘和记忆两个步骤。我们使用了先前得到的更新门控 z (update gate)。

更新表达式：

$$h^t = (1-z)\odot h^{t-1} + z\odot h' \tag{3}$$

门控信号(这里的 z)的范围为 0~1。门控信号越接近 1，代表“记忆”下来的数据越多；而越接近 0 则代表“遗忘”的越多。GRU 很聪明的一点就在于，使用了同一个门控 z 就同时可以进行遗忘和选择记忆(LSTM 则要使用多个门控)。

总结来看，GRU 的输入输出结构与普通 RNN 相似，内部思想与 LSTM 相似，相比 LSTM 少了一个“门控”，参数能少，但功能近似。

2.4. TCN

典型的 TCN [6] [7]模型图 6 包括三个基本要素：因果卷积(Causal Convolution)、膨胀卷积(Dilated Convolution)和残差连接(Residual Connection)。

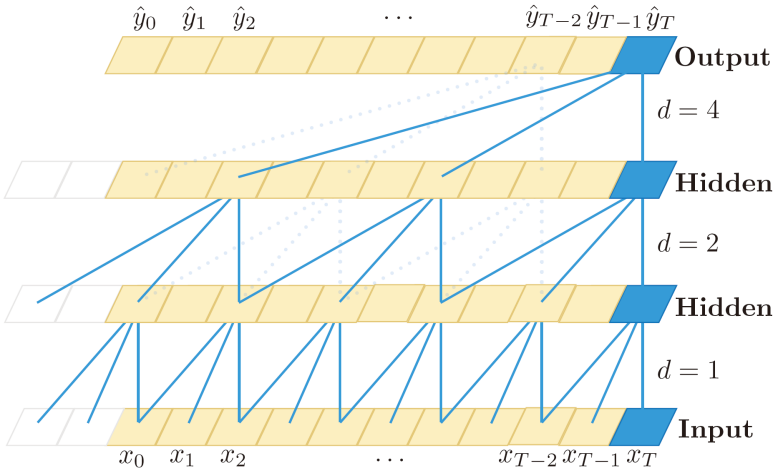


Figure 6. Schematic diagram of TCN structure
图 6. TCN 结构示意图

因果卷积是指在 T 时刻卷积操作只能与 T 时刻之前的数据做卷积，即模型只能获得 T 时刻之前的信

息，这么做是为了避免未来的信息泄露进来。简单的说，TCN 即一维全卷积(FCN) + 因果卷积。这种简单的卷积方式其获得的历史长度随着深度程线性增长，如果想要获得较久的历史信息就必须使用很大卷积核和很深的网络架构，为此引入了膨胀卷积。

膨胀卷积允许卷积时的输入存在间隔采样，如下图所示，采样率受图中的 d 控制。最下面一层的 $d = 1$，表示输入时每个点都采样，中间层 $d = 2$，表示输入时每 2 个点采样一个作为输入。膨胀卷积使得有效窗口的大小随着层数呈指数型增长。这样卷积网络用比较少的层，可以获得很大的感受野。

自 2015 年残差网络(ResNet) [8]提出以来，其就被广泛应用于神经网络的各个模型之中以解决模型退化问题。为了增加模型的深度和获取更好的训练效果，TCN 的每两层之间引入残差连接形成一个残差块。残差连接使得网络可以以跨层的方式传递信息。一个残差块包含两层的卷积和非线性映射，在每层中还加入了 WeightNorm 和 Dropout 来正则化网络图 7。

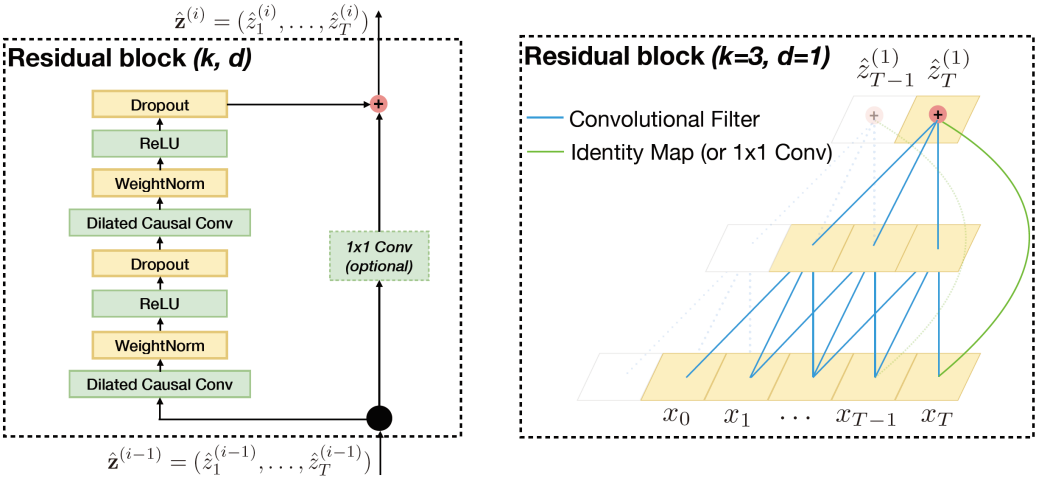


Figure 7. Schematic diagram of residual module structure
图 7. 残差模块结构示意图

TCN 的优点主要有以下几个：

1) 并行性。TCN 可以将句子并行的处理，不需要像 RNN 那样顺序的处理。

2) 灵活的感受野。TCN 的感受野可以根据不同的任务不同的特性灵活定制。

3) 稳定的梯度。TCN 不太存在梯度消失和爆炸问题。

4) 内存更低。TCN 在一层里面卷积核是共享的，内存使用更低。

同样，TCN 也有以下几个缺点：

1) TCN 在迁移学习方面没有那么强的适应能力。这是因为在不同的领域，模型预测所需要的历史信息量可能是不同的。因此，在将一个模型从一个对记忆信息需求量少的问题迁移到一个需要更长记忆的问题上时，TCN 可能会表现得很差，因为其感受野不够大。

2) 论文中描述的 TCN 还是一种单向的结构，在语音识别和语音合成等任务上，纯单向的结构还是相当有用的。但是在文本中大多使用双向的结构，当然将 TCN 也很容易扩展成双向的结构，不使用因果卷积，使用传统的卷积结构即可。

3) TCN 毕竟是卷积神经网络的变种，虽然使用扩展卷积可以扩大感受野，但是仍然受到限制，相比于 Transformer 那种可以任意长度的相关信息都可以抓取到的特性还是差了点。TCN 在文本中的应用还有待检验。

3. 基于深度神经网络的单词预测模型

为了设计具有对比效果的单词预测系统，本文分别采用了 LSTM，CRU 和 TCN 三种不同的方法做字符预测，其中两端的 Encoder 层和 Decoder 都是一样的。其中 Encoder 层是单层的 Embedding；Decoder 是单层线性层。

TCN 字符预测模型的中间层为 4 层，卷积核大小 $k=3$，膨胀因子 $d=1,2,4,8$，每层的输入、输出通道数目均为 600；Encoder 层使用单层嵌入输入大小为 10,000，输出为 600；Decoder 使用单层全连接层，输入大小为 600，输出大小为 10000。TCN 字符预测模型结构图如图 8：

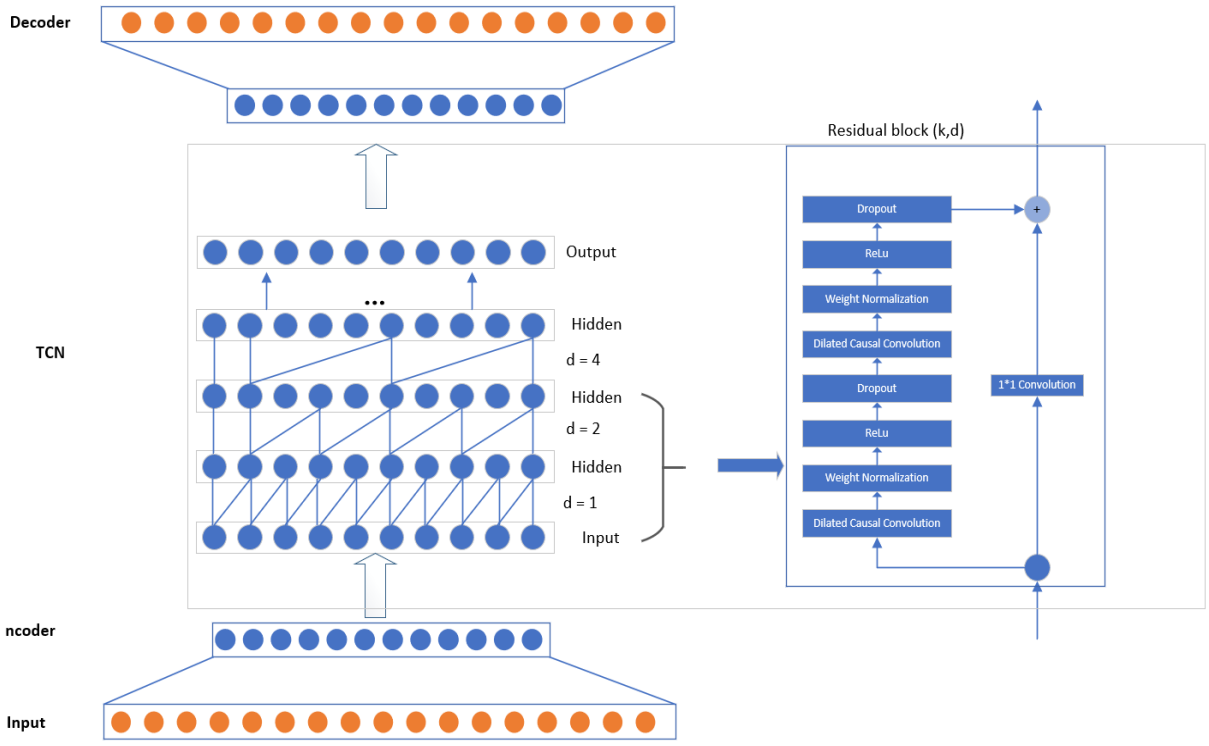


Figure 8. TCN character prediction model structure diagram
图 8. TCN 字符预测模型结构图

4. 实验结果与分析

4.1. 数据集介绍

Penn Treebank (PTB)数据集是一个文本处理领域里使用最为广泛的数据集，其最早由 Marcus 等根据来自 1989 年华尔街日报上的 2499 篇文章构建。当用作单词级的处理任务时共有 888,000 个样本用来训练，70,000 个样本作为验证集，79,000 个样本作为测试集。用作字符级别的任务时共有 5,059,000 个样本作为测试，396,000 个样本作为验证集，446,000 个样本作为测试集。单词总数为 10,000 个。本实验为单词级的预测任务，数据集在 Miyamoto 的开源代码中有提供。

4.2. 实验环境

CPU：Intel(R) Xeon(R) Gold 6230

GPU：Tesla V100 SXM2

Python：3.7.0

Pytorch：1.7.1

4.3. 评价指标

这里性能采用两个指标来评估，其一是交叉熵损失(Cross Entropy Loss)；假设误差是二值分布，可以视为预测概率分布和真实概率分布的相似程度，其在多分类任务中的表达式为：

$$L=\frac{1}{N}\sum_i L_i=\frac{1}{N}\sum_i -\sum_{c=1}^{M} y_{ic}\log\left(p_{ic}\right) \tag{4}$$

其二是困惑度 ppl (perplexity)，ppl 是用在自然语言处理领域(NLP)中，衡量语言模型好坏的指标。它主要是根据每个词来估计一句话出现的概率，并用句子长度作正则化，公式为：

$$PP(S)=P\left(w_1 w_2\cdots w_N\right)^{-\frac{1}{N}} \tag{5}$$

4.4. 训练过程

对于三种模型，训练时均设置 $lr=0.2$，迭代次数为 200。下面是三种方法的损失变化过程图 9、图 10、图 11。

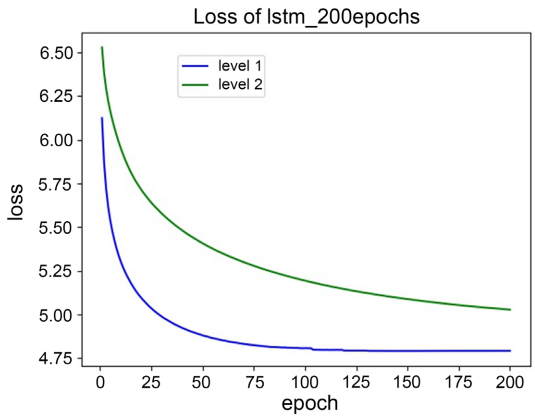


Figure 9. Loss drop graph of LSTM training process
图 9. LSTM 训练过程 loss 下降图

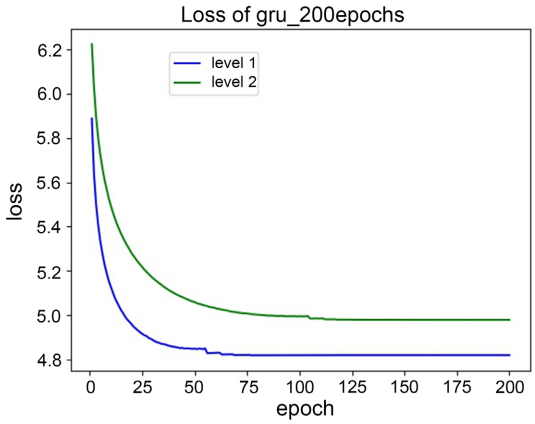


Figure 10. Loss drop graph of GRU training process
图 10. GRU 训练过程 loss 下降图

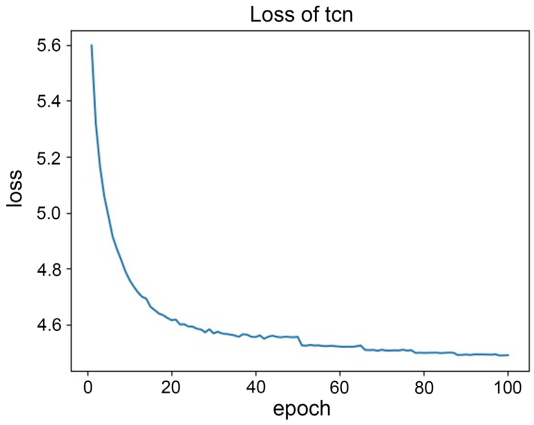


Figure 11. Loss drop graph of TCN training process
图 11. TCN 训练过程 loss 下降图

4.5. 实验结果分析

对比三种不同方法的损失曲线可以发现，TCN 可以更早收敛，而且对应的损失也比较小，但是误差存在抖动情况。而 GRU 和 LSTM 训练效果相差不多、收敛较慢，误差变化相对比较平滑。

基于 LSTM、GRU 和 TCN 的三个模型效果如下图 12、图 13 所示，ppl 分别为 120.63、123.94、89.24，交叉熵损失为 4.79，4.82, 4.49。其中 TCN 模型性能最好，LSTM 和 GRU 性能相似。

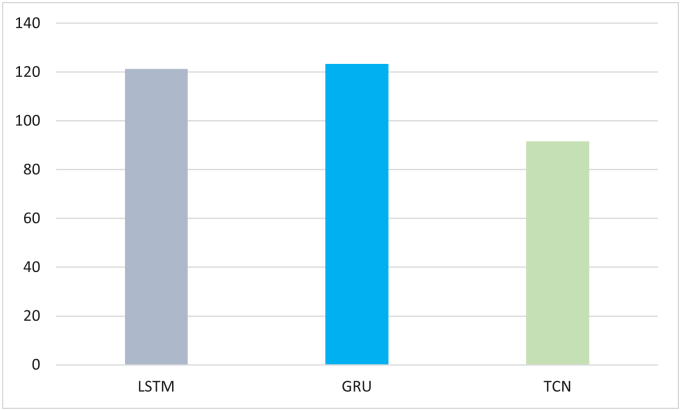


Figure 12. ppl index comparison chart of three methods
图 12. 三种方法 ppl 指标对比图

ID	Method	Cross-Entropy Loss
1	LSTM (Long Short-Term Memory)	4.79
2	GRU (Gated Recurrent Unit)	4.82
3	TCN (Temporal Convolution Network)	4.49

Figure 13. Comparison chart of cross-entropy indicators of three methods
图 13. 三种方法交叉熵指标对比图

5. 单词预测系统实现

5.1. 总体设计

NextWordPrediction 系统是一个 Web 应用程序，主要分为前端和后端两个部分。前端通过浏览器客户端接收服务器发送的页面文件并显示，接受用户请求发送给服务端处理；后端分为 Web 逻辑处理和模型控制两部分，Web 逻辑处理负责接受客户端的请求，调用相应模型控制接口返回响应结果，渲染页面文件返回给客户端。

5.2. 模块设计

系统主要包括两个模块：单步预测和数据集评估

单步预测模块的流程如图 14 所示。首先后端加载模型，前端提交预测任务，后端通过调用预测接口返回前端预测结果，前端接受并显示最终结果。

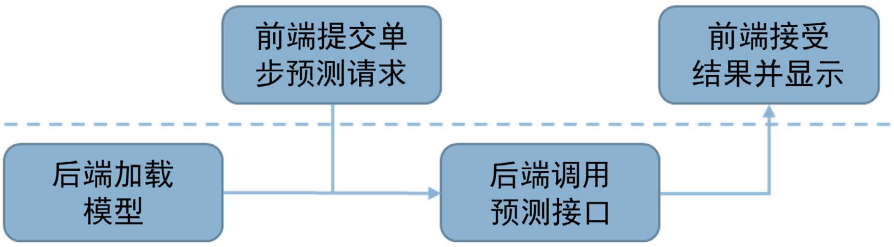


Figure 14. Schematic diagram of single-step prediction module
图 14. 单步预测模块示意图

数据集评估模块如图 15 所示。通过在测试集上评估三个模型的性能进行比较，主要指标为 PPL 和交叉熵损失，前端发起评估请求，后端加载模型并返回评估结果，最终在前端展示评估结果。

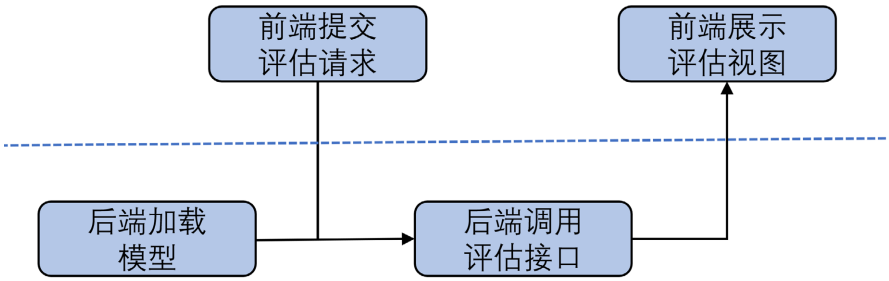


Figure 15. Schematic diagram of the dataset evaluation module
图 15. 数据集评估模块示意图

5.3. 系统实现与展示

本系统的前端基于 HTML、CSS 和 JavaScript 语言，后端开发语言为 Python。

前端框架为 Bootstrap。Bootstrap 是美国 Twitter 公司的设计师 Mark Otto 和 Jacob Thornton 合作基于 HTML、CSS、JavaScript 开发的简洁、直观、强悍的前端开发框架，使得 Web 开发更加快捷。Bootstrap 提供了优雅的 HTML 和 CSS 规范，它即是由动态 CSS 语言 Less 写成。

后端框架为 Flask。Flask 是一个轻量级的可定制框架，使用 Python 语言编写。它可以很好地结合 MVC 模式进行开发。Flask 还有很强的定制性，可以根据需求来添加相应的功能，在保持核心功能简单地同时实现功能的丰富与扩展，其强大的插件库可以让用户实现个性化的网站定制，开发出功能强大的网站。

整体网络以及功能界面如图 16、图 17、图 18 所示。

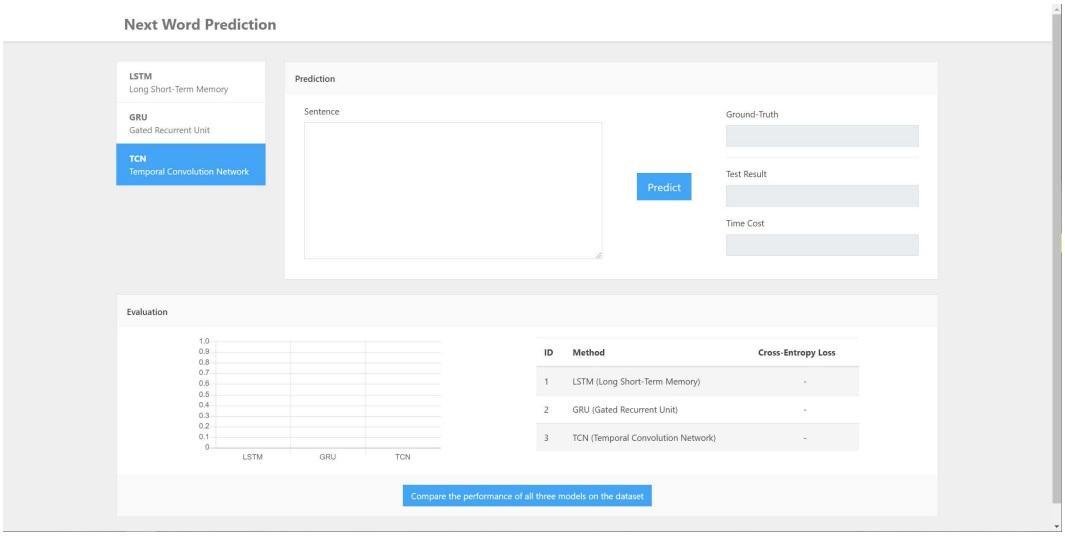


Figure 16. Overall web interface
图 16. 整体 web 界面

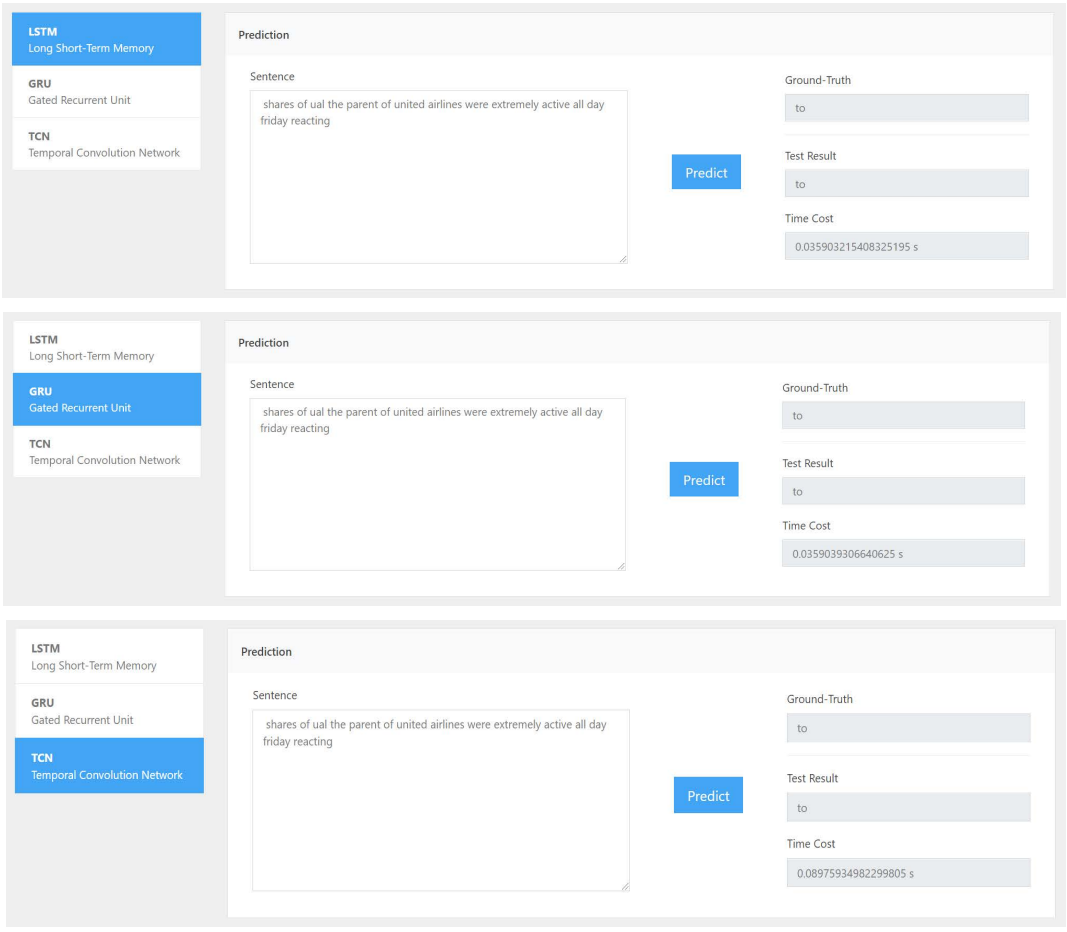


Figure 17. Schematic diagram of prediction function
图 17. 预测功能示意图

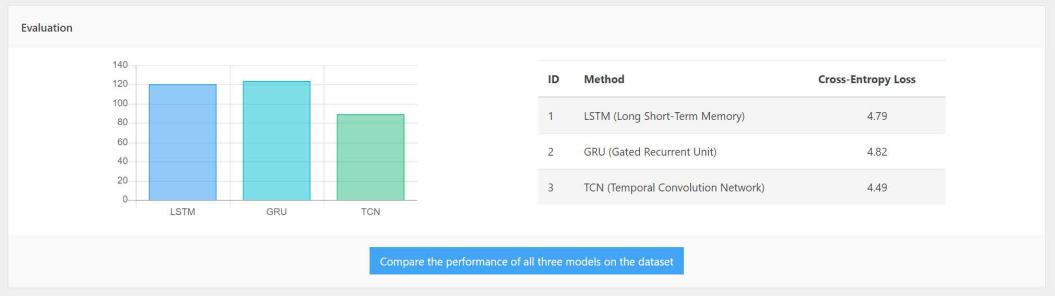


Figure 18. Dataset evaluation function demonstration
图 18. 数据集评估功能展示

6. 结论

本文基于当下最为流行的三种深度神经网络设计了单词预测模型，其中基于 TCN 的模型总体上达到了最好的效果，同时利用 web 前后端技术将模型部署到云端平台，可视化展示预测的结果并对结果评估。实现了系统的设计需求。当然也有改进的地方，例如在更多的数据集上作训练和验证，实时展示模型的性能消耗等。

参考文献

[1] Paperno, D., Kruszewski, G., Lazaridou, A., Pham, Q.N., Bernardi, R., Pezzelle, S., Baroni, M., Boleda, G. and Fernández, R. (2016) The LAMBADA Dataset: Word Prediction Requiring a Broad Discourse Context. In: *Proceedings of the* 54*th Annual Meeting of the Association for Computational Linguistics* (*Volume* 1: *Long Papers*), Association for Computational Linguistics, Berlin, 1525-1534. https://doi.org/10.18653/v1/P16-1144

[2] Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H. and Bengio, Y. (2014) Learning Phrase Representations Using RNN Encoder-Decoder for Statistical Machine Translation. In: *Proceedings of the* 2014 *Conference on Empirical Methods in Natural Language Processing* (*EMNLP*), Association for Computational Linguistics, Doha, 1724-1734. https://doi.org/10.3115/v1/D14-1179

[3] Goldberg, Y. (2016) A Primer on Neural Network Models for Natural Language Processing. *Journal of Artificial Intelligence Research*, **57**. https://doi.org/10.1613/jair.4992

[4] Irie, K., Tüske, Z., Alkhouli, T., Schlüter, R. and Ney, H. (2016) LSTM, GRU, Highway and a Bit of Attention: An Empirical Overview for Language Modeling in Speech Recognition. 3519-3523. https://doi.org/10.21437/Interspeech.2016-491

[5] Chung, J., Gulcehre, C., Cho, K. and Bengio, Y. (2014) Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling.

[6] Bai, S., Kolter, J.Z. and Koltun, V. (2018) An Empirical Evaluation of Generic Convolutional and Recurrent Networks for Sequence Modeling.

[7] van den Oord, A., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A. and Kavukcuoglu, K. (2016) WaveNet: A Generative Model for Raw Audio.

[8] He, K.M., Zhang, X.Y., Ren, S.Q. and Sun, J. (2015) Deep Residual Learning for Image Recognition. 2016 *IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), Las Vegas, 27-30 June 2016, 770-778. https://doi.org/10.1109/CVPR.2016.90