实验报告：自动写诗

韦俊林 (201928016029023)

2020 年 6 月 18 日

1 摘要

本次实验完成的是一个自然语言处理任务，使用“tang.npz”一个整理好的唐诗数据集，实现一个自动写诗模型。本次实验中基于 pytorch 深度学习框架，搭建一个 3 层的门控循环神经网络 (GRU)，还运用了 Embedding 实现 word2vec，最终模型能按照指定字段生成有一定质量的唐诗。

2 问题描述

基于 Python 语言和任意一种深度学习框架，完成数据读取、网络设计、网络构建、模型训练和模型测试等过程，最终实现一个可以自动写诗的程序。

随意给出首句，如给定“湖光秋月两相和”，输出模型续写的诗句。也可以根据自己的兴趣，进一步实现写藏头诗（不做要求）。要求输出的诗句尽可能地满足汉语语法和表达习惯。

3 解决方案

3.1 损失函数

本次实验使用的损失函数为 pytorch 库里的交叉熵损失函数，pytorch 库中的交叉熵损失函数是将 log_softmax 和 nll_loss 进行了结合，具体计算公式为：

$$\text{loss}(x, class) = weight[class]\left(-x[class] + \log\left(\sum_j \exp(x[j])\right)\right)$$

代码实现如下图所示：

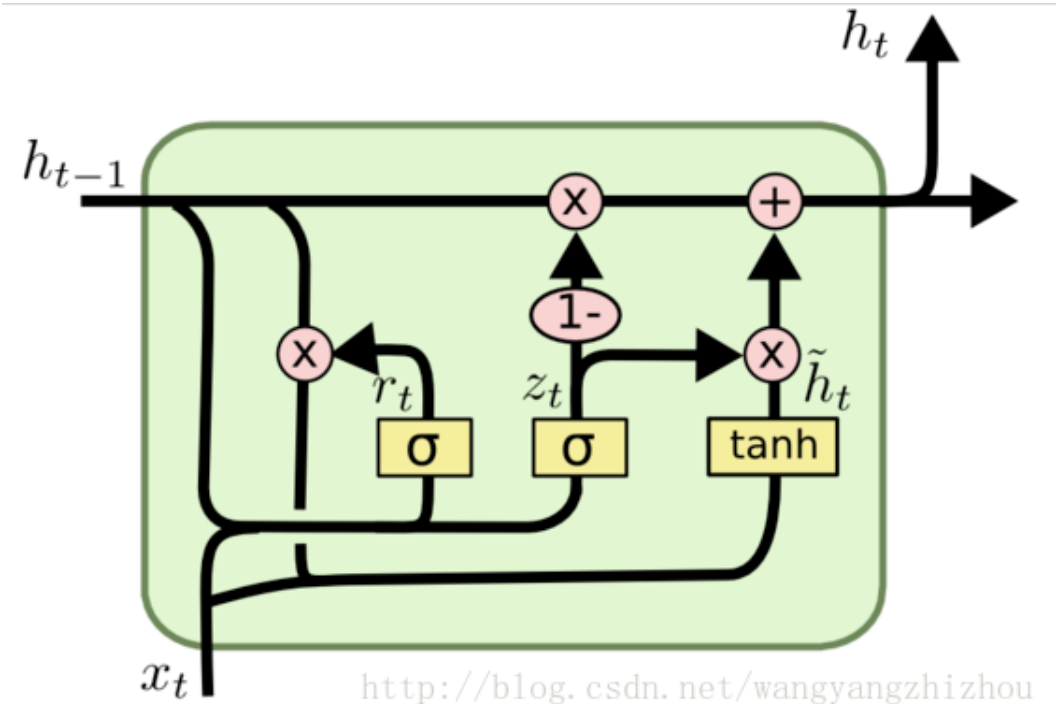
```
# 设置损失函数
criterion = nn.CrossEntropyLoss().to(device)
```

3.2 网络结构

本次实验搭建的网络是首先搭建一个 Embedding 层，然后接一个 3 层的门控循环神经网络，最后连接一个全连接层。

其中 Embedding 层实现将字词序号映射成相应的词向量，本次实验是将数据集中的字词映射成 128 维的词向量。

接着一个 3 层的 GRU，GRU 可以捕捉时间序列中时间步距离较大的依赖关系，通过可以学习的门来控制信息的流动，核心在于门控循环单元，如下图所示：



内部结构计算公式如下：

$$
\begin{aligned}
r_t &= \sigma(W_{ir}x_t + b_{ir} + W_{hr}h_{(t-1)} + b_{hr}) \\
z_t &= \sigma(W_{iz}x_t + b_{iz} + W_{hz}h_{(t-1)} + b_{hz}) \\
n_t &= \tanh(W_{in}x_t + b_{in} + r_t * (W_{hn}h_{(t-1)} + b_{hn})) \\
h_t &= (1 - z_t) * n_t + z_t * h_{(t-1)}
\end{aligned}
$$

最后连接一个全连接层输出。

网络实现代码如下图所示：

```python
class PoetryModel(nn.Module):
    def __init__(self, vocab_size, embedding_dim, hidden_dim, num_layers):
        super(PoetryModel, self).__init__()
        self.hidden_dim = hidden_dim
        self.num_layers = num_layers
        self.embeddings = nn.Embedding(vocab_size, embedding_dim)

        self.gru = nn.GRU(embedding_dim, hidden_dim, num_layers=num_layers, bidirectional=False)
        self.linear = nn.Linear(hidden_dim, vocab_size)


    def forward(self, input, hidden=None):
        seq_len, batch_size = input.size()

        if hidden is None:
            h_0 = input.data.new(self.num_layers, batch_size, self.hidden_dim).fill_(0).float()
            h_0 = Variable(h_0)
        else:
            h_0 = hidden

        embeds = self.embeddings(input)
        output, hidden = self.gru(embeds, h_0)

        output = self.linear(output.view(seq_len * batch_size, -1))
        return output, hidden
```

4 实验分析

4.1 数据集介绍

本次实验采用的数据集是来自 GitHub 上中文诗词爱好者收集的 5 万多首唐诗原文，作者已将该数据集处理成一个 numpy 的压缩包 tang.npz，包含三个对象。

data:(57580,125) 的 numpy 数组，总共 57580 首诗歌，每首诗歌长度为 125 字符；

word2ix: 每个词和它对应的序号；

ix2word: 每个序号和它对应的词。

具体细节为，将每首诗歌先转成 list，然后在是个的前后加上起始符 <START> 和终止符 <END>。

导入数据集代码如下图所示：

```python
def load_dataset(file):
    dataset=np.load(file, allow_pickle=True)
    data=dataset['data']
    ix2word=dataset['ix2word'].item()
    word2ix=dataset['word2ix'].item()

    return data,ix2word,word2ix
```

4.2 实验结果

本次完成本次实验的项目总共包含 4 个文件，分别是

main.py：主函数包括载入数据集，初始化参数、模型，训练模型，输出结果等功能；

model.py：搭建网络模型以及相应训练函数；

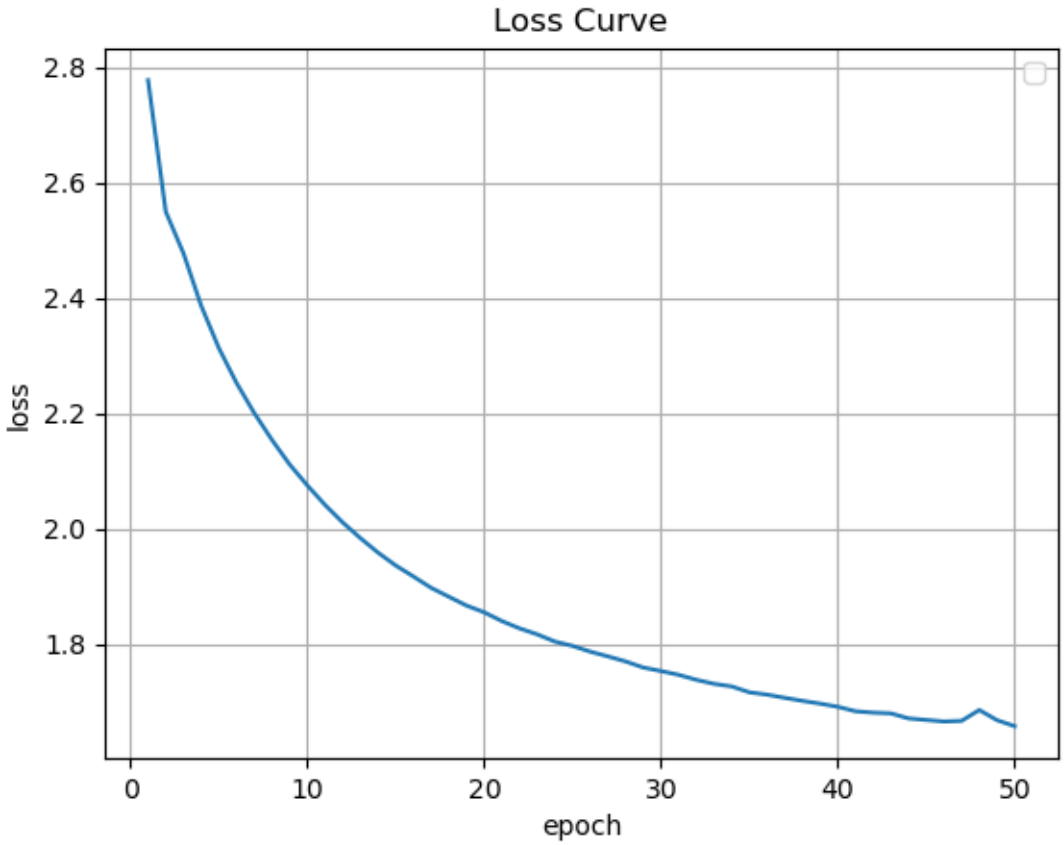
utils.py：包括一些辅助函数，比如载入数据集函数、生成诗歌、保存结果等功能；
tang.npz：用于训练的数据集。
本次实验模型的超参数以及一些设定的参数配置如下：

```
class config(object):
    dataset_file='tang.npz'
    lr=1e-3
    batch_size=128
    num_epochs=50
    start_words_1='机器学习'
    start_words_2='秋水共长天一色'
    max_gen_len = 125
    device = torch.device('cuda:0' if torch.cuda.is_available()
    else 'cpu')
    train_or_load=False
    num_layers=3
    save_path=mkdir('20200617_1827')
```

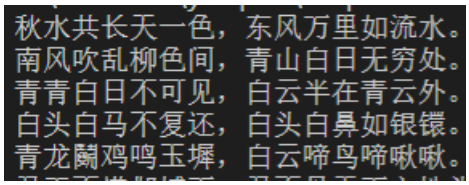
训练过程中，terminal 记录的部分过程：

```
Train Epoch: 1 [0/57580 (0.0%)] Loss: 8.943035   elapse: 12.095 sec
Train Epoch: 1 [6272/57580 (10.9%)]     Loss: 2.927370   elapse: 26.153 sec
Train Epoch: 1 [12672/57580 (22.0%)]    Loss: 2.727026   elapse: 40.476 sec
Train Epoch: 1 [19072/57580 (33.1%)]    Loss: 2.588480   elapse: 54.926 sec
Train Epoch: 1 [25472/57580 (44.2%)]    Loss: 2.629054   elapse: 69.622 sec
Train Epoch: 1 [31872/57580 (55.3%)]    Loss: 2.802446   elapse: 84.623 sec
Train Epoch: 1 [38272/57580 (66.4%)]    Loss: 2.683708   elapse: 99.784 sec
Train Epoch: 1 [44672/57580 (77.6%)]    Loss: 2.494002   elapse: 115.169 sec
Train Epoch: 1 [51072/57580 (88.7%)]    Loss: 2.745704   elapse: 130.576 sec
Train Epoch: 1 [48492/57580 (99.8%)]    Loss: 2.801627   elapse: 145.953 sec
the folder has been created
```

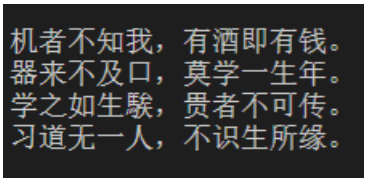
随着迭代次数 epoch 增加，模型的交叉熵损失曲线：



模型输出实例，以生成“秋水共长天一色”为开头的诗歌：



模型输出实例，以生成藏头诗“机器学习”：



5 总结与分析

训练的模型基本达到了任务要求，生成的诗歌单从每一句来看都带有古诗词的味道。但从整体上来，整首诗或者上下文来看相互之间没有联系，古诗没有一个明确主旨。并且可能是迭代次数不够或者是网络不够复杂，有些句子仅仅是形式上是古诗，但实际上却是个病句。

本次实验是一个 NLP，相比于上两次 CV 的实验，从理解数据集、处理数据、搭建网络到最后的训练网络几乎是完全不同。因此，经过本次实验基本掌握了完成 NLP 任务的基本流程以及一些技术细节。