基于pytorch + LSTM 的古诗生成

作业介绍:

本课程使用pytorch框架, 完成NLP任务:古诗生成,使用的模型为 LSTM, 并训练了词向量, 支持随机古诗和藏头诗生成, 并且生成的古诗具有多变性。

导包:

In [1]:
```
import os
import numpy as np
import pickle
import torch
import torch.nn as nn
from gensim.models.word2vec import Word2Vec
from torch.utils.data import Dataset, DataLoader
```

生成切分文件:

In [2]:
```
def split_text(file="poetry_7.txt", train_num=6000):
    all_data = open(file, "r", encoding="utf-8").read()
    with open("split_7.txt", "w", encoding="utf-8") as f:
        split_data_7 = " ".join(all_data)
        f.write(split_data_7)
    return split_data_7[:train_num * 64]
```

训练词向量:

In [3]:
```
def train_vec(split_file="split_7.txt", org_file="poetry_7.txt", train_num=6000)
    param_file = "word_vec.pkl"
    org_data = open(org_file, "r", encoding="utf-8").read().split("\n")[:train_n
    if os.path.exists(split_file):
        all_data_split = open(split_file, "r", encoding="utf-8").read().split("\
    else:
        all_data_split = split_text().split("\n")[:train_num]

    if os.path.exists(param_file):
        return org_data, pickle.load(open(param_file, "rb"))

    models = Word2Vec(all_data_split, vector_size=128, workers=7, min_count=1)
    pickle.dump([models.syn1neg, models.wv.key_to_index, models.wv.index_to_key]
    return org_data, (models.syn1neg, models.wv.key_to_index, models.wv.index_to
```

构建数据集:

In [4]:
```
class Poetry_Dataset(Dataset):
    def __init__(self, w1, word_2_index, all_data):
        self.w1 = w1
        self.word_2_index = word_2_index
        self.all_data = all_data

    def __getitem__(self, index):
        a_poetry = self.all_data[index]

        a_poetry_index = [self.word_2_index[i] for i in a_poetry]
        xs = a_poetry_index[:-1]
        ys = a_poetry_index[1:]
        xs_embedding = self.w1[xs]

        return xs_embedding, np.array(ys).astype(np.int64)

    def __len__(self):
        return len(self.all_data)
```

模型构建:

In [5]:
```
class Poetry_Model_lstm(nn.Module):
    def __init__(self, hidden_num, word_size, embedding_num):
        super().__init__()

        self.device = "cuda" if torch.cuda.is_available() else "cpu"
        ######定义模型######
        self.device = "cuda" if torch.cuda.is_available() else "cpu"
        self.hidden_num = hidden_num

        self.lstm = nn.LSTM(input_size=embedding_num, hidden_size=hidden_num, ba
                            bidirectional=False)
        self.dropout = nn.Dropout(0.3)
        self.flatten = nn.Flatten(0, 1)
        self.linear = nn.Linear(hidden_num, word_size)
        self.cross_entropy = nn.CrossEntropyLoss()


    def forward(self, xs_embedding, h_0=None, c_0=None):
        if h_0 == None or c_0 == None:
            h_0 = torch.tensor(np.zeros((2, xs_embedding.shape[0], self.hidden_n
            c_0 = torch.tensor(np.zeros((2, xs_embedding.shape[0], self.hidden_n
        h_0 = h_0.to(self.device)
        c_0 = c_0.to(self.device)
        xs_embedding = xs_embedding.to(self.device)
        ######定义模型######
        hidden, (h_0, c_0) = self.lstm(xs_embedding, (h_0, c_0))
        hidden_drop = self.dropout(hidden)
        hidden_flatten = self.flatten(hidden_drop)
        pre = self.linear(hidden_flatten)

        return pre, (h_0, c_0)
```

自动生成古诗:

In [9]:
```
def generate_poetry_auto():
    result = ""
    word_index = np.random.randint(0, word_size, 1)[0]

    result += index_2_word[word_index]
    h_0 = torch.tensor(np.zeros((2, 1, hidden_num), dtype=np.float32))
    c_0 = torch.tensor(np.zeros((2, 1, hidden_num), dtype=np.float32))

    for i in range(31):
        word_embedding = torch.tensor(w1[word_index][None][None])
        pre, (h_0, c_0) = model(word_embedding, h_0, c_0)
        word_index = int(torch.argmax(pre))
        result += index_2_word[word_index]

    return result
```

藏头诗生成:

In [10]:
```
def generate_poetry_acrostic():
    input_text = input("请输入四个汉字：")[:4]
    result = ""
    punctuation_list = [", ", "。", ", ", "。"]
    for i in range(4):
        result += input_text[i]
        h_0 = torch.tensor(np.zeros((2, 1, hidden_num), dtype=np.float32))
        c_0 = torch.tensor(np.zeros((2, 1, hidden_num), dtype=np.float32))
        word = input_text[i]
        for j in range(6):
            word_index = word_2_index[word]
            word_embedding = torch.tensor(w1[word_index][None][None])
            pre , (h_0,c_0) = model(word_embedding,h_0,c_0)
            word = word_2_index[int(torch.argmax(pre))]
            result += word

    return result
```

主函数: 定义参数, 模型, 优化器, 模型训练

In [11]:
```
if __name__ == "__main__":

    all_data, (w1, word_2_index, index_2_word) = train_vec(train_num=300)

    batch_size = 32
    epochs = 100
    lr = 0.01
    hidden_num = 128
    word_size, embedding_num = w1.shape
```

```
    dataset = Poetry_Dataset(w1, word_2_index, all_data)
    dataloader = DataLoader(dataset, batch_size)

    model = Poetry_Model_lstm(hidden_num, word_size, embedding_num)
    model = model.to(model.device)

    optimizer = torch.optim.AdamW(model.parameters(), lr=lr)

    for e in range(epochs):
        for batch_index, (batch_x_embedding, batch_y_index) in enumerate(dataloa
            model.train()
            batch_x_embedding = batch_x_embedding.to(model.device)
            batch_y_index = batch_y_index.to(model.device)

            #模型预测
            pre, _ = model(batch_x_embedding)
            #计算损失
            loss = model.cross_entropy(pre, batch_y_index.reshape(-1))
            # 梯度反传 , 梯度累加, 但梯度并不更新, 梯度是由优化器更新的
            loss.backward()
            # 使用优化器更新梯度
            optimizer.step()
            # 梯度清零
            optimizer.zero_grad()

            if batch_index % 100 == 0:
                # model.eval()
                print(f"loss:{loss:.3f}")
                print(generate_poetry_auto())
```

loss:7.707
舫，，，，，，，，。，，，，，，，，，，，，，，。，，，，。，，。，
loss:6.962
则，。。，。。，，。。。。，，。。，，。，。。。。，。。，，，，，。
loss:6.800
立，。，，。。。，。，，，，。，。。。。。。。。。。。。。，，。。，。
loss:6.725
窅。。，有，，有。有，。。，色，色，有，有。，，，有，色，，有，
loss:6.644
马山花花花，山山山，花海，山山山山山花花，山花，花山山山山花花，
loss:6.583
壶山山山山，山，山山山，山山，山海，山山，山，山山海，山山海，山
loss:6.414
程山风风山无，，风，，，，，，，，，，，，，，，，，，，，，，，，，
loss:6.292
甲有三海不，，山，，，，，，，，，天。。山。一。。，一。，，，，
loss:6.163
半色三三一天，，来一一。一一。。。一一山。。。一一。。。一一。。
loss:6.110
双帆三路六水，，一山山一天。一一。山一无天。，一山一无无。。山一
loss:6.051
缘门三光斗，海天，一一，一天。一天。一一。一一。一一。一一。一一
loss:5.956
开得三十斗斗，一山不一一。一山一来。一一。一一水，一风一人。一来
loss:5.864
嗤台下车自斗险，一山一山人色。一山一一一。一山一人水。一来一似人
loss:5.792
陈榜风车斗天，天山不一一水。一来风来不人，一风一子一水。海山一人
loss:5.737
慵门由气千海菲，一花一山一天。一来一花一天。一山一年一水，一来一
loss:5.676
人人风风三远，海山一山一。金风一有无，一山一山不来。一来一有一无
loss:5.601
优今极树海有，萧山一山一青。一山风山一无。海山一里天时。一来一山
loss:5.574
弼门十草政苇阳，一山衔子一浮，。山烟垒一人。一来一声不节。一来一
loss:5.532
慧门十气斗注开，匹山无榔一水花。一山一山一人，一山一人频心。一来
loss:5.509
幔得下气接峰菲，砌霄一咏一山中。海时一望不来，海山烟山无春。一山
loss:5.474
槟阙喧光海告，一山似光一天。一天一来一来来，一来一风一无来。一山
loss:5.408
静台风光自生，匹下一光一天。一山风时一水，一里应花不公。一是一声
loss:5.378
晨蛇红山六重垴，瘴下一阁旧水中。海来一影一无花，一山遥风一云花。
loss:5.333
迥色烟路三有，东山一阁一天。一来一山一无。一来一人一无。海家一人
loss:5.311
斜台花气政澄，一山无阁一水。一山风风不水。一来一人一无，一山一处
loss:5.268
巧浪下车都重，一山清阁一公。一教一时一公。一教如头一野，海里一影
loss:5.251
逢台三人斗林天，一门一气一天端。白是一年知水，一山一复无天。未来
loss:5.227
穿寂下结接三天，破山隐咏如天林。一来一影一人客，一风一山一沧公。
loss:5.216
蜂门高构不天宣，一到天里一水心。一来一山不时老，，知一年初不花。
loss:5.204
术门走光香氏垴，一见一咏兴短仙。海阳不头一无绿，一来风年一大仙。

loss:5.188
逗门高结斗峰城，瓮贲台子一相林。但来一山谁时渡，争山一年一公青。
loss:5.137
提果高踞政自枝，振山清咏一短心。有是一年三无世，万月烟风一禅仙。
loss:5.118
推得觉光都氏宣，匹然台气见天心。一是一山无无在，一家烟处一浮时。
loss:5.100
宴门西光接峰堰，三章新山一诗端。天山一人无无有，未酒应处有园花。
loss:5.041
板门山光乘可中，瘴章编花旧节台。满山一年秋无，一来秋声无紫仙，一
loss:5.049
跋门筛结政绩坳，砌山编气一天羊。一风一年无野楣，檀梦遥处不沧仙。
loss:5.035
冉门不屭香其宣，振棠风子一盘心。海是不年不紫渡，一山一声不春花。
loss:5.084
眠门十光接绩坳，一光清阁一一长。一阁一头来多处，争山遥翩一禅仙。
loss:5.069
琳榜下传量氏履，砌到黄阁一并心。天知风觉无无在，愿里一人一云花。
loss:4.973
估门下光政冈席，甘岭烟咏统短公。木是一年无野会，一耳遥日共不花。
loss:4.984
墓门十光三露先，一光花咏一天莲。一是一教无野有，争山遥山未朦青。
loss:4.925
拘榜喧传接难先，砌棠编声一天公。遥今一藤知无在，一山遥风一人青。
loss:4.905
菩兢喧涧列莽先，砌马台花一染璃。此是一日清无在，一来深日一大花。
loss:4.940
骑女烟路乡铁婢，雕挛呵护色芙中。海是一风清无线，一指一时一留寒。
loss:4.860
犯门筛车日绩城，三山一护旧兰新。赤是一年天无在，海梦一风一沧青。
loss:4.844
经身秋雨度生城，一爱三花一短天。他阁揽教回时客，更山遥人一阳青。
loss:4.809
样门宝辣接绩坳，砌下编阁色染衣。白是一年天清林。海是一处无敷里，
loss:4.771
卜挤盘光六铁先，瘴章编气吐诗春。海上一年清无夜，黄来秋公一不来。
loss:4.760
邑帆矗物总峰坳，一霄此阁一诗屏。赤露一层知知后，残落天人一疑羹。
loss:4.733
篇暄童传姓氏先，文章台水吐青来。遥知一时知圣妄，一及一翩一大青。
loss:4.728
第璃猊推又注枝，匹章台阁水战春。怪是盛砻无时夜，万里一人共阳青。
loss:4.629
褒门高车政林坳，愿灵此阁一染屏。澄是盛舒频染供，愿报烟风一仔屏。
loss:4.663
慰得下结政绩韬，愿棠一咏色短屏。澄蚩一风驯田化，愿易遥风一短林，
loss:4.629
并茫风路乡注室，一霄载骨色娉挠。白是盛砻循番处，一绕烟瓠尚公仙。
loss:4.599
屠门下车看绩坳，无鬟一咏一染衣。海是海有来染醲，愿枝风翩尚仔中。
loss:4.563
渺宕淑气都难阳，软门盛岭色战场。白知盛德来醉会，一遣我村勒玉青。
loss:4.485
造门下光政萧宣，一凭衣溪吐染栖。铁阁一盏频染绿，白天一验一人家。
loss:4.462
坳兆喧传姓氏先，文章台根吐青深。家关揽路天无在，白月如日牛肯衣。
loss:4.394
弛兢喧檄渡氏，砌下又阁疑染蓉。菜甲一年初沙谙，愿得安验一仔中。笋
loss:4.370
尾余繁光认无茞，一棠无咏号虎莲。好里背路知无阔，愿鞭疑家一仔青。

loss:4.384
棱鱼腻矗挥画新，每穰芋叶焉行公。大寞独风藏心翠，重来文劚尚未笼。
loss:4.360
势花三柳自道涯，甡然天近一二端。十儿不随铭黄线，一乡应香寸槛钱。
loss:4.335
格门下结接林坳，砌下编篱作短屏。菜甲一年频音醪，愿甲卧竹拥觳脐。
loss:4.203
廷得下车并溶溶，小阑载国下深层。读甲一影偕舍渡，笋恭恒河死涛花。
loss:4.167
佳迷宿逦耕印熙，一步巅榔赋成公。天是一斜谁能老，万竿寒声一阳沉。
loss:4.149
忉蛇腻上谪点森，脂下宜手展青莲。雄是盛主留浏绿，只妍阵时一战窝。
loss:4.109
踵门下结耸绩坳，砌下编篱作毋挠。怒令如舒驯奋绿，拍奏阵洲无阳赠，
loss:4.024
男瀛风处度生阳，匹落盛篱似战场。白是一回金色夜，万梦遥槎汉疑香。
loss:4.027
甘光觉路六千阳，匹碎红里旧战场。满阁纡萸循筋綯，笋酒遥望寸一听。
loss:3.917
畅鸡吞剑认卒溶，砌下又篱作忉挠。离途一舒频染醲，愿得弼时拥禅脐。
loss:3.975
访清隐樯政绩宣，砌下编篱作毋挠。白笑蒲砻循雕泉，故乡无劚一飏鹭。
loss:3.932
萤鱼下车列绝名，井碎珊瑚作珠屏。三家握砻凭浏浏，海翮无虱故壤长。
loss:3.855
篷怪猊床侍释音，筠亚芒拍韭本丰。三苞风年僧舍信，海心遥处一钓筹。
loss:3.812
野本由柳自芳菲，西步无光不夕台。怪知年鸡天此在，畅飙陆理无翠家。
loss:3.728
篱八别翻三脚开，功嶂一莱一更馨。香日不幸黄凝集，万臣，身是老有寒
loss:3.786
星本由岸斗四知，也庸划房见相滨。巡海枯睹辛饶湿，笋丝脱花拥禅扉。
loss:3.717
试茫爽气豁无沙，招砾一溪照战枪。白发蒲葱罗公在，一桑烟忆久园稽。
loss:3.632
酷香筛辣春溶溶，砌醅傍天入短屏。菜甲初舒频染醪，更耳心槎贡人葩。
loss:3.579
度迷古事度峰催，更携此溪茎楫长。荡阳纵有天清夜，漫鸡壁垒几头红。
loss:3.535
仅叶高结接溶伦，甘载聪篱鑸挠支。换今内优来骑浏，鹿省风安似福星。
loss:3.456
缦门筛辣春溶溶，瓮醅芒教咏短屏。菜甲初舒频染绿，笋鳞未劚尚留青。
loss:3.405
桴榜喧传姓氏先，文章台阁吐青莲。天教盛德色敷治，故遣王味重未深。
loss:3.372
齐朝古雄三绸缪，大步樊笼半作淳。采是对处秋兴处，万沾万香或一寒。
loss:3.376
摘鱼腻极不斗龈，甘棠载咏入诗篇。澄蚩短语兼只妇，淹鞭熏起冠舜廷。
loss:3.368
科暄方法名冬舒，雕稻墨台芽诃茅。汲板绝时桅论水，教吴镇日一不飞。
loss:3.286
质得下樯列林坳，砌下编篱作短屏。菜甲初舒频染绿，笋恭沥河色定青。
loss:3.204
腻箐驯祥波峰寅，三霄异听白忉工。离蛇张兽鹏之化，愿鞭未劚尚留青。
loss:3.130
嵫日走割挥林坳，砌下编骨色琴。海是突砻循良泉，踏年金处尚洞箫。铁
loss:3.151
成耳极目闻载我，况蟾清西照子荧。夜来绛荔辛勤处，谨方脱壳射摇陶。
loss:3.136
叟日扶结接绝伦，砌下编篱作短屏。菜甲初舒呕翼揲，绝乡为虱却蛟家。

loss:3.093
屿仙含月庆若有，偶无匡叶问不妆。暴骇铿訇两鹏径，茧妍脱测附官联。
loss:3.068
颓门梅象不萧梦，鲤舶骸唤层头足。一清握手知公飞，腹缈扶莽一染思。
loss:3.042
宴歌坏邦控上头，井马罗山统舍耶。双螯客仆常敷化，笋留犹起重系肩。
loss:3.083
元滩螺前生笑累，入灵清胎飓母长。拜篷一献声虞翠，缦陌涛锐牙爪伤。
loss:3.030
荷身周气量余粟，愿乞骸前入诗足。菜甲初舒频染醲，云磬金肃尚女收。
loss:2.957
匏闪清国菊告中，鲲章台和暮井衣。剧阳一除秋首客，调竿民欲争霭凝。
loss:2.853
启璃浓露不千点，玳瑁筵酒玉几层。远种纤埃侵皓素，檀心普马御人狃。
loss:2.856
格义下旋雌远砧，愿乞骸骨还为狡。疏清揽砻知公志，愿更弼时重仔肩。
loss:2.774
椒璃浓露艳幽光，郑宅高芽两粉枪。白嫩蛎房调最滑，尘更佳浪此云泥。
loss:2.827
闽门御极量余眸，砌乞骸传作芙挠。酋亭初帆吠狂手。笋，舌蜜厌楼鲂。

描述一下你的模型：

模型概述

该模型是一个基于深度学习的古诗生成系统，它利用word2vec技术将汉字表示为词向量，并通过LSTM（长短期记忆网络）和一个全连接层来构建预测模型。通过训练，该模型能够根据给定的前一个字预测下一个字，从而生成古诗。

模型细节

1. **汉字表示（word2vec）**
    - **输入**：汉字集合（例如，从大量古诗中提取的所有不重复汉字）。
    - **处理**：使用word2vec算法将每个汉字转换为一个固定维度的词向量。这些词向量能够捕捉到汉字之间的语义关系，例如相似的汉字在向量空间中会彼此靠近。
    - **输出**：一个汉字到词向量的映射表，以及每个汉字对应的词向量。
2. **封装数据**
    - **目的**：为了方便模型处理，需要将汉字及其对应的词向量封装成数据格式。
    - **处理**：为每个汉字的词向量分配一个唯一的索引，并创建一个字典来存储这些索引和词向量的对应关系。同时，将古诗文本转换为一系列索引，以便模型能够读取和预测。
    - **输出**：索引到词向量的映射字典，以及古诗文本转换为索引序列的数据集。
3. **构建模型**
    - **架构**：模型由一个LSTM层和一个全连接层组成。LSTM层负责捕捉序列数据中的长期依赖关系，而全连接层则用于将LSTM的输出转换为预测下一个字的概率分布。
    - **输入**：前一个字的词向量（通过索引从字典中检索得到）。
    - **输出**：下一个字的可能性的概率分布（通常是一个softmax层输出的概率向量）。

4. **训练**

- **过程**：使用大量古诗文本作为训练数据，通过前向传播和反向传播算法来训练模型。在训练过程中，模型会尝试根据前一个字的词向量来预测下一个字的索引。
- **损失函数**：通常使用交叉熵损失函数来衡量模型预测的准确性，并通过梯度下降等优化算法来更新模型的权重。
- **终止条件**：训练过程会在达到预定的迭代次数、损失函数收敛到某个阈值或模型在验证集上的性能不再提升时终止。

5. **生成古诗**

- **过程**：在训练完成后，模型可以根据给定的起始字或句子生成古诗。这通常是通过在模型中输入起始字的词向量，并反复使用模型的预测输出来生成下一个字来实现的。
- **控制**：可以通过设置生成古诗的长度、使用特定的起始字或句子以及调整生成过程中的随机性来控制生成的古诗的风格和内容。

总结

该模型利用word2vec将汉字表示为词向量，并通过LSTM和全连接层构建了一个能够预测下一个字的深度学习模型。通过训练，该模型能够生成符合古诗风格的文本。这种模型在文学创作、自然语言处理等领域具有广泛的应用前景。