# 运用pytorch进行rnn建模

## **Sina（用前60个数预测第61个数）**

从0到2π中等间隔的取4000个数，放入sin()函数中，得到我们需要的样本值。接下来将前3000个数据作为训练集，后1000个数据作为测试集。训练集与测试集中都把前60个数作为输入数据，第61个数作为标签，形成一个训练样本，最后将所有样本打乱放入网络。损失函数采用均方误差。

网络结构

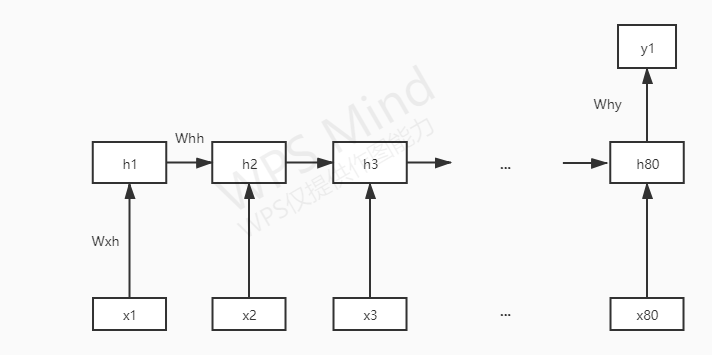


图1 rnn网络结构图

其中





的形状为80\*1，的形状为80\*80，bh的形状为80\*1，why的形状为80\*1，by的形状为1，所以总共的参数量为80\*80+80\*3+1=6641

得到结果：

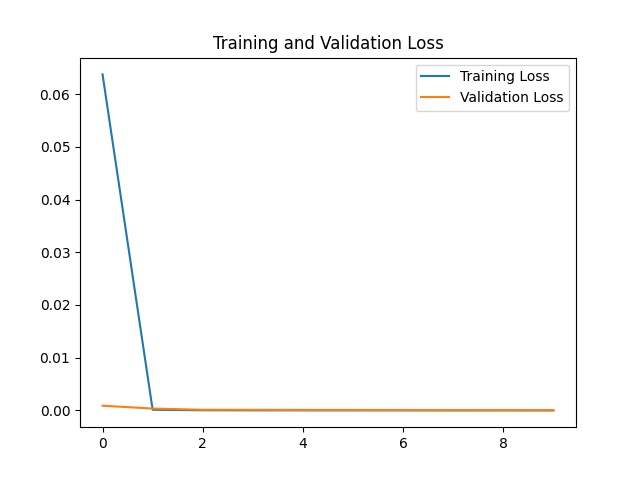


图2 训练集与测试集损失函数的变化情况

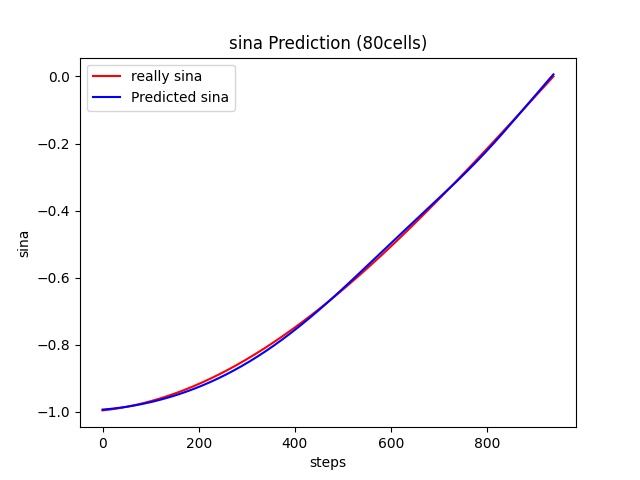


图3 测试集中真实值与预测值的对比情况

## sina 用前一个数预测后一个数

Total params: 6,641

|  |  |  |
| --- | --- | --- |
|  | 1v1 | 60v1 |
| 均方误差: | 0.000017 | 0.000045 |
| 均方根误差: | 0.004109 | 0.006690 |
| 平均绝对误差: | 0.003665 | 0.005707 |

发现用1v1的效果更好，更换不同步长进行测试。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1v1 | 5v1 | 10v1 | 20v1 | 30v1 | 60v1 | 100v1 |
| 均方误差: | 0.000017 | 0.000357 | 0.000194 | 0.000059 | 0.000037 | 0.000045 | 0.000209 |
| 均方根误差: | 0.004109 | 0.018892 | 0.013939 | 0.007707 | 0.006113 | 0.006690 | 0.014465 |
| 平均绝对误差: | 0.003665 | 0.017133 | 0.012569 | 0.006862 | 0.005544 | 0.005707 | 0.010377 |

可以看到测试的步长中，依然是1v1的效果更好，考虑原因可能是在训练同一数据量的参数时，1v1提供给模型的数据量是最优的，其他步长并没有在1v1的基础上给出更优的信息，反而使模型进行信息的选择，剔除了更好的信息。

## 3.记忆体个数的测试：

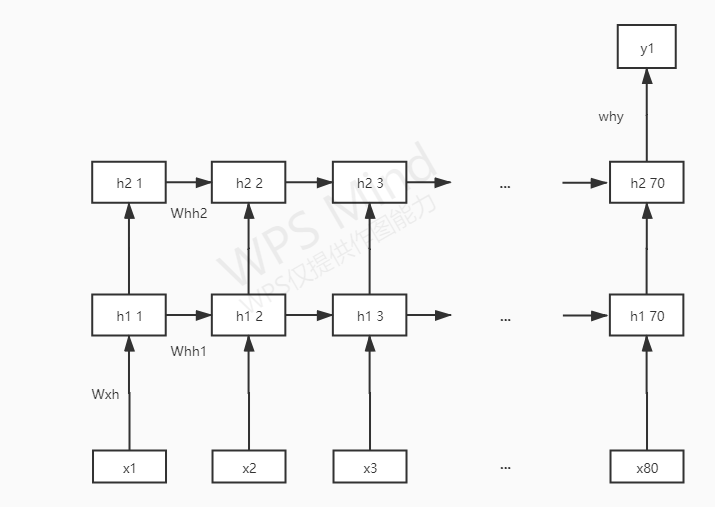
在上面的实验中采用的记忆体个数为80个，现在测试不同的记忆体个数，看对神经网络有什么不同的影响。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 记忆体个数 | 10 | 20 | 30 | 40 | 50 | 60 |
| 参数个数 | 131 | 461 | 991 | 1,721 | 2,651 | 3,781 |
| 均方误差: | 0.000108 | 0.000014 | 0.000022 | 0.000009 | 0.000013 | 0.000008 |
| 均方根误差: | 0.010379 | 0.003691 | 0.004657 | 0.003079 | 0.003645 | 0.002913 |
| 平均绝对误差: | 0.008538 | 0.003025 | 0.004069 | 0.002723 | 0.003112 | 0.002629 |
| 记忆体个数 | 70 | 80 | 90 |  |  |  |
| 参数个数 | 5,111 | 6,641 | 8,371 |  |  |  |
| 均方误差: | 0.000001 | 0.000011 | 0.000002 |  |  |  |
| 均方根误差: | 0.001081 | 0.003352 | 0.001389 |  |  |  |
| 平均绝对误差: | 0.000812 | 0.002838 | 0.001009 |  |  |  |

可以看出准确度并不是单纯的随着参数个数与模型复杂度的增加而增加的，在记忆体个数选择70时，模型在测试集上的准确度最大。

## 4.Rnn层数的测试

增加rnn网络隐藏层的层数，观察模型的情况：（选取1v1的步长，第一层取70个记忆体，第二层取70个记忆体）。



其中







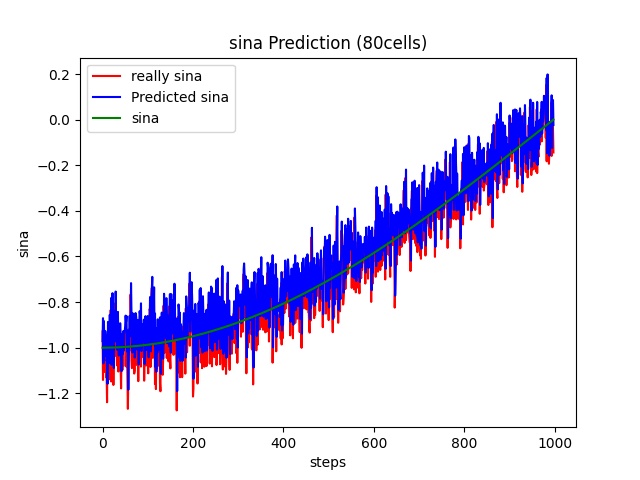
的形状为70\*1，的形状为70\*70，bh1的形状为70\*1，why的形状为80\*1，的形状为70\*70，的形状为70\*70，bh2的形状为70\*1，why的形状为70\*1，by的形状为1，所以总共的参数量为14,981。

|  |  |  |  |
| --- | --- | --- | --- |
| Rnn层数 | 1（70） | 1（70）2（70） | 1（70）2（70）3（70） |
| 均方误差: | 0.000020 | 0.000104 | 0.000193 |
| 均方根误差: | 0.004505 | 0.010207 | 0.013883 |
| 平均绝对误差: | 0.004035 | 0.009265 | 0.012711 |

可以看到增加层数，反而模型的预测效果更差，推想可能是数据量不大，参数过多造成过拟合。

## 5.添加噪声进行预测（均值为0，方差为0.10的随机噪声）

运用1v1的划分方法分训练集与测试集，模型为一层70个记忆体个数的rnn，得到结果：



Total params: 5,111

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

|  |  |  |
| --- | --- | --- |
|  | Rnn | Sina |
| 均方误差: | 0.021534 | 0.009840 |
| 均方根误差: | 0.146743 | 0.099195 |
| 平均绝对误差: | 0.118982 | 0.080623 |
| 平均相对误差 | 2.357425 | 0.385786 |

可以看到rnn模拟出了一条较为贴合绿色曲线的趋势，对于绝对误差来说，rnn的效果对比sina原曲线还是稍差。但对于相对误差来说，两个模型的差别很大，考虑可能的原因，可能是rnn选取的loss函数为绝对量，所以相对误差被忽略。也可能是模型过拟合，模拟了过多的噪声导致模型的精度不够。

1. **修改训练区间观测最佳步长**

将整体区间修改为0到4π，前3000个数据作为训练集，后1000个数据作为验证集。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1v1 | 5v1 | 10v1 | 20v1 | 30v1 | 60v1 | 100v1 |
| 均方误差: | 1.70E-05 | 3.57E-04 | 1.94E-04 | 5.90E-05 | 3.70E-05 | 4.50E-05 | 2.09E-04 |
| 均方根误差: | 4.11E-03 | 1.89E-02 | 1.39E-02 | 7.71E-03 | 6.11E-03 | 6.69E-03 | 1.45E-02 |
| 平均绝对误差: | 3.67E-03 | 1.71E-02 | 1.26E-02 | 6.86E-03 | 5.54E-03 | 5.71E-03 | 1.04E-02 |

表6.1：原训练集下步长测试

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | 1v1 | 5v1 | 10v1 | 20v1 | 30v1 | 60v1 | 100v1 |
| 均方误差: | 1.96E-05 | 1.02E-05 | 4.69E-06 | 6.46E-07 | 1.74E-06 | 4.53E-07 | 4.46E-07 |
| 均方根误差: | 4.43E-03 | 3.20E-03 | 2.17E-03 | 8.04E-04 | 1.32E-03 | 6.73E-04 | 6.68E-04 |
| 平均绝对误差: | 3.62E-03 | 2.68E-03 | 1.60E-03 | 6.71E-04 | 1.18E-03 | 5.76E-04 | 5.10E-04 |

表6.2：新训练集下步长测试

可以观测到在新训练集上，随着步长的增大，误差是逐渐减小的，说明在前一区间学习到的趋势帮助后一区间的预测。在原训练集上步长并没有优化模型效果的原因可能是学习区间只有一个周期，学习到的趋势并没有用途。

接下来将区间更改为0到12π，产生6000个数据，前4000个作为训练集，后2000个作为测试集。查看周期步长与半周期步长的建模效果。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1v1 | 100v1 | 500v1 | 2000v1 |
| 均方误差: | 3.10E-05 | 8.35E-07 | 3.40E-07 | 1.97E-07 |
| 均方根误差: | 5.57E-03 | 9.14E-04 | 5.83E-04 | 4.44E-04 |
| 平均绝对误差: | 4.52E-03 | 7.40E-04 | 4.88E-04 | 3.76E-04 |

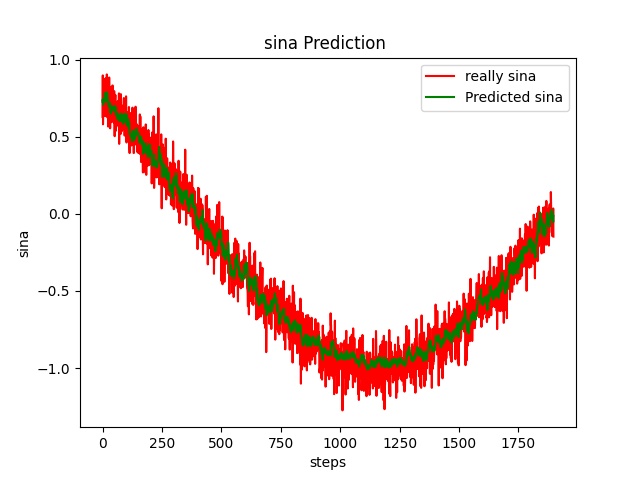
表6.3 半周期步长与周期步长的测试效果

1. **建立arma模型进行比较**

在0到4π区间取6000个数，前4000个数为训练集，后2000个为测试集，同时加入均值为0，方差为0.1的方差的噪声。

在RNN建模里，建立100步长以及70个记忆神经元的rnn。同时建立arma模型进行对比。

RNN:



Arma：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |
| Variable | Coefficient | Std. Error | t-Statistic | Prob. |
|  |  |  |  |  |
|  |  |  |  |  |
| AR(1) | 1.000095 | 0.000206 | 4856.121 | 0.0000 |
| MA(1) | -0.909743 | 0.006578 | -138.3073 | 0.0000 |
|  |  |  |  |  |
|  |  |  |  |  |
| R-squared | 0.978435 | Mean dependent var | | 0.177846 |
| Adjusted R-squared | 0.978429 | S.D. dependent var | | 0.708137 |
| S.E. of regression | 0.104004 | Akaike info criterion | | -1.688275 |
| Sum squared resid | 43.23487 | Schwarz criterion | | -1.685127 |
| Log likelihood | 3377.706 | Hannan-Quinn criter. | | -1.687159 |
| Durbin-Watson stat | 2.027350 |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
| Inverted AR Roots | 1.00 | |  |  |
|  | Estimated AR process is nonstationary | | | |
| Inverted MA Roots | .91 | |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

表7.1 arma模型表  
建立模型：

|  |  |  |
| --- | --- | --- |
|  | Rnn 100v1 | Arma |
| 均方误差: | 0.0119704443 | 0.010827 |
| 均方根误差: | 0.1094095254 | 0.104051 |
| 平均绝对误差: | 0.0872553621 | 0.08364 |
| 平均相对误差 | 7.516227 | 0.538742 |

表7.2 rnn与arma模型对比表