Assignment2

> 命令行
>
> > python source.py

Part I

默认情况下，网络正确率为0。

观察网络最终输出首位（即输出的末位）各数字的概率，发现在全部测试数据中，首位为0的概率均远高于首位为1的概率。观察训练、测试数据，发现训练数据中为[0, 555555555]范围内的数据，几乎不需要在末位进位；而测试数据中为[555555555, 999999999]范围内的数据，全部需要在末位进位；由于训练及测试集的偏差，导致网络认为首位是0的概率远高于首位为1的概率，导致全部出错。

Part2

默认情况下，由于训练集及测试集的偏差，在2至9位加法时准确率均为0。

训练、测试集

> 这里的测试均为500轮，学习率0.01。

将训练集与测试集均改为[0, 999999999]，此时

位数	2	3	4	5	6	7	8	9
正确率	0.307	0.1605	0.0865	0.04	0.0155	0.0095	0.005	0.0025

此处正确率大致按每增加一位，正确率减半指数衰减。

观察此时的输出可以发现，网络的输出中几乎全部的首位均为0或均为1，推测网络并未能学会首位的进位，而是单独为首位指定为0或1。

> 后面将采用7位加法进行测量。

网络结构

1. 网络深度

   每一层RNN均能独立完成一次加法进位操作，推测在高维加法时可能需要多次进位，又在网络过深时可能出现梯度爆炸，将网络深度设置为加法位数。

2. 正则化

   为网络添加dropout正则层。

3. 激活函数

   考虑到模拟的是加法进位操作，采用线性的ReLU激活函数可能效果更好。

4. 模型单元

   考虑采用lstm、GRU单元

5. 模型结构

   采用encoder-decoder的seq2seq结构。

优化器

1. 学习率递减

为了更好地收敛至最小值，采用学习率指数下降。