计算机科学与技术学院神经网络与深度学习课程实验报告

实验题目：循环神经网络		学号：201600301304
日期：2019/4/20	班级： 人工智能 16	姓名： 贾乘兴

Email：1131225623@qq.com

实验目的：利用莎士比亚数据集，基于 numpy 实现循环神经网络（RNN）的前向计算与方向传播（bptt），并测试 softmax 函数的参数，查看效果。并基于分布生成不同的文本。尝试找到 rnn 的决定的参数

实验软件和硬件环境： 操作系统 mac os，内存 16GB，编译器 pycharm

实验原理和方法：

一. RNN 的前向计算

1. rnn 单元

对于序列 t 的计算，已知输入 x（t）与上一个状态 s（t-1），对 t 下状态与生成的概率分布计算如下

$$h_t = \tanh\left(W_{xh}x_t + W_{hh}h_{t-1} + b_h\right)$$
$$y_t = W_{hy}h_t + b_y$$
$$p_t = softmax\left(y_t\right)$$

2. rnn 单元的初始状态

对于最初的状态 h(0),令其为零向量,在后续的 encoder-decoder 模型中 encoder 得到的结果作为 decoder 的初始状态从而继续生成序列

3. 误差计算

我们使用交叉熵 loss，对 onehot 的 target，计算的 loss 如下

$$loss = \sum_{t=1}^{n} target_t \log\left(p_t\right)$$

二. RNN 的误差反向传播（bptt）

1. Rnn 单元

我们在得到整个序列后（长度为 lenght），从末尾开始反向传播，对各层输入的求导计算如下

$$\frac{dloss}{dy_t} = y_t - target_t$$
$$\frac{dloss}{dh_t} = \frac{dloss}{dy_t}\cdot\frac{dy_t}{dh_t} + \frac{dh_{t+1}}{dh_t} = W_{hy}^T\left(y_t - target_t\right) + dh_{next_t}$$
$$\frac{dloss}{dh_t^{raw}} = \frac{dloss}{dh_t}\odot\left(1 - h_t\odot h_t\right)$$

在一个序列中对所有参数的求导如下

$$\frac{dloss}{db_y}=\sum_{t=1}^{n}\frac{dloss}{dy_t}$$

$$\frac{dloss}{dW_{hy}}=\sum_{t=1}^{n}\frac{dloss}{dy_t}\cdot h_t^T$$

$$\frac{dloss}{db_h}=\sum_{t=1}^{n}\frac{dloss}{dh_t^{raw}}$$

$$\frac{dloss}{dW_{xh}}=\sum_{t=1}^{n}\frac{dloss}{dh_t^{raw}}\cdot x_t^T$$

$$\frac{dloss}{dW_{hh}}=\sum_{t=1}^{n}\frac{dloss}{dh_t^{raw}}\cdot h_{t-1}^T$$

2. rnn 初始

从末尾求导时 dhnext 为 0 向量

3. 参数更新（梯度下降）

$$param \coloneqq param-\eta\nabla param$$

三. Softmax 层

1. Softmax（normal）

为了保证计算精度，进行标准化处理，对求导无影响

$$y_t=y_t-\max\left(y_t\right)$$

然后得到 softmax

$$p_{ti}=\frac{e^{y_{ti}}}{\sum_{j=1}^{m}e^{y_{tj}}}$$

2. Softmax（temperature）

加入参数 temperature，对随机生成的依赖的概率分布进行方差的调整

$$y_t=\frac{y_t-\max\left(y_t\right)}{\tau}$$

调整适当的参数可以得到更好的效果，同时反向传播只需要除以该系数

3. Gumbel softmax

在计算 softmax 层时，进行如下调整

$$y_t=y_t-\max\left(y_t\right)$$

$$g_t=\frac{\left(-y_t-e^{-y_t}\right)}{\tau}$$

$$p_t=softmax\left(g_t\right)$$

对于反向传播，改进如下

$$\frac{dloss}{dg_t} = p_t - target_t$$

$$\frac{dg_t}{dy_t} = -I_1 + e^{-y_t}$$

$$\frac{dloss}{dy_t} = \left(p_t - target_t\right) \odot \left(-I_1 + e^{-y_t}\right)$$

实验步骤：（不要求罗列完整源代码）

1 . 补全代码

```
"""
Minimal character-level Vanilla RNN model. Written by Andrej Karpathy (@karpathy)
BSD License
"""
import numpy as np

# data I/O
data = open('shakespeare_train.txt', 'r').read() # should be simple plain text file
chars = list(set(data))#your code#  # 得到输入文件中所有字符种类
data_size, vocab_size = len(data), len(chars) #your code##统计文件字符数和字符种类数
print ('data has %d characters, %d unique.' % (data_size, vocab_size))
char_to_ix = {ch:id for id,ch in enumerate(chars)} #your code# #构成从字母到数字的映射
ix_to_char = {id:ch for id,ch in enumerate(chars)} #your code# #构成数字到字母的映射


# hyperparameters
hidden_size = 100 # size of hidden layer of neurons
seq_length = 25 # number of steps to unroll the RNN for
learning_rate = 1e-1

# model parameters 初始化参数
Wxh = np.random.randn(hidden_size, vocab_size)*0.01 # input to hidden
Whh = np.random.randn(hidden_size, hidden_size)*0.01 # hidden to hidden
Why = np.random.randn(vocab_size, hidden_size)*0.01 # hidden to output
bh = np.zeros((hidden_size, 1)) # hidden bias
by = np.zeros((vocab_size, 1)) # output bias

def lossFun(inputs, targets, hprev, temp=1):
  """
  inputs,targets are both list of integers.
  hprev is Hx1 array of initial hidden state
  returns the loss, gradients on model parameters, and last hidden state
  """
  xs, hs, ys, ps = {}, {}, {}, {}
  hs[-1] = np.copy(hprev)
```

```
  loss = 0
  # forward pass
  for t in range(len(inputs)):
    #encode inputs to 1-hot embedding,size(xs)=(len(input),vocab_size)
    xs[t] = np.zeros((vocab_size, 1))#your code# # encode in 1-of-k representation 1-hot-encoding
    xs[t][inputs[t]] = 1 #your code# # encode in 1-of-k representation 1-hot-encoding
    #forward

    #hs[t] 是t时刻的hidden state， active function = np.tanh(z)，z = Wx*x_t+Wh*hs_(t-1) + bh,即本时刻输入层+一时刻个隐含层作为Z
    hs[t] = np.tanh(np.dot(Wxh, xs[t]) + np.dot(Whh, hs[t-1]) + bh) #your code# # hidden state

    #ys[t] = w*hs[t]+by
    ys[t] = np.dot(Why, hs[t]) + by #your code# # unnormalized log probabilities for next chars
    #softmax(ys)
    ys[t] = (ys[t] - np.max(ys[t],axis=0)) / temp
    ps[t] = np.exp(ys[t])/np.sum(np.exp(ys[t]),axis=0)#your code# # probabilities for next chars
    #计算loss = cross_entropy（）
    loss += -np.log(ps[t][targets[t]])#your code# # softmax (cross-entropy loss)
  # backward pass: compute gradients going backwards
  #初始化梯度
  dWxh, dWhh, dWhy = np.zeros_like(Wxh), np.zeros_like(Whh), np.zeros_like(Why)
  dbh, dby = np.zeros_like(bh), np.zeros_like(by)
  dhnext = np.zeros_like(hs[0])

  for t in reversed(range(len(inputs))):
    #dy是softmax层求导，cross_entropy softmax 求导 aj-yi,yi为one-hot标签,aj为softmax之后第j个神经元输出，详情请见https://blog.csdn.net/u014313009/article/details/51045303
    dy = np.copy(ps[t]) #your code#
    dy[targets[t]] -= 1#your code# # backprop into y.
    dy = dy / temp
    #反向传播，求Why与by的导数

    dWhy += np.dot(dy, hs[t].T) #your code#
    dby += dy #your code#
    #反向传播到hidden state请参考https://blog.csdn.net/wjc1182511338/article/details/79191099完成，其中dh处反向传播的梯度外需加上dhnext
    dh = np.dot(Why.T, dy) + dhnext #your code# # backprop into h

    dhraw = dh * (1 - hs[t] * hs[t]) #your code# # backprop through tanh nonlinearity
    dbh += dhraw #your code#
    dWxh += np.dot(dhraw, xs[t].T)#your code#
    dWhh += np.dot(dhraw, hs[t-1].T)#your code#
    dhnext = np.dot(Whh.T, dhraw) #your code#
```

```
  for dparam in [dWxh, dWhh, dWhy, dbh, dby]:
    np.clip(dparam, -5, 5, out=dparam) # clip to mitigate exploding gradients
  return loss, dWxh, dWhh, dWhy, dbh, dby, hs[len(inputs)-1]

def sample(h, seed_ix, n, temp=1):
  """
  sample a sequence of integers from the model
  h is memory state, seed_ix is seed letter for first time step
  """
  x = np.zeros((vocab_size, 1))
  x[seed_ix] = 1
  ixes = []
  for t in range(n):
    h = np.tanh(np.dot(Wxh, x) + np.dot(Whh, h) + bh)
    y = np.dot(Why, h) + by
    y = (y - np.max(y,axis=0)) / temp
    p = np.exp(y) / np.sum(np.exp(y))
    ix = np.random.choice(range(vocab_size), p=p.ravel())
    x = np.zeros((vocab_size, 1))
    x[ix] = 1
    ixes.append(ix)
  return ixes

n, p = 0, 0
mWxh, mWhh, mWhy = np.zeros_like(Wxh), np.zeros_like(Whh), np.zeros_like(Why)
mbh, mby = np.zeros_like(bh), np.zeros_like(by) # memory variables for Adagrad
smooth_loss = -np.log(1.0/vocab_size)*seq_length # loss at iteration 0
tempareture = 2
while True:
  # prepare inputs (we're sweeping from left to right in steps seq_length long)
  if p+seq_length+1 >= len(data) or n == 0:
    hprev = np.zeros((hidden_size,1)) # reset RNN memory
    p = 0 # go from start of data
  inputs = [char_to_ix[ch] for ch in data[p:p+seq_length]]
  targets = [char_to_ix[ch] for ch in data[p+1:p+seq_length+1]]

  # sample from the model now and then
  if n % 100 == 0:
    sample_ix = sample(hprev, inputs[0], 200, temp=tempareture)
    txt = ''.join(ix_to_char[ix] for ix in sample_ix)
    print ('----\n %s \n----' % (txt, ))

  # forward seq_length characters through the net and fetch gradient
  loss, dWxh, dWhh, dWhy, dbh, dby, hprev = lossFun(inputs, targets, hprev, temp=tempareture)
```

```
  smooth_loss = smooth_loss * 0.999 + loss * 0.001
  if n % 100 == 0: print ('iter %d, loss: %f' % (n, smooth_loss)) # print progress)

  # perform parameter update with Adagrad
  for param, dparam, mem in zip([Wxh, Whh, Why, bh, by],
                                [dWxh, dWhh, dWhy, dbh, dby],
                                [mWxh, mWhh, mWhy, mbh, mby]):
    mem += dparam * dparam
    param += -learning_rate * dparam / np.sqrt(mem + 1e-8) # adagrad update

  p += seq_length # move data pointer
  n += 1 # iteration counter
```

2 . 加入 temperature（代码见上），gumbel softmax（代码如下）

```
"""
Minimal character-level Vanilla RNN model. Written by Andrej Karpathy (@karpathy)
BSD License
"""
import numpy as np

# data I/O
data = open('shakespeare_train.txt', 'r').read() # should be simple plain text file
chars = list(set(data)) # your code# # 得到输入文件中所有字符种类
data_size, vocab_size = len(data), len(chars) # your code##统计文件字符数和字符种类数
print('data has %d characters, %d unique.' % (data_size, vocab_size))
char_to_ix = {ch: id for id, ch in enumerate(chars)} # your code# #构成从字母到数字的映射
ix_to_char = {id: ch for id, ch in enumerate(chars)} # your code# #构成数字到字母的映射

# hyperparameters
hidden_size = 100 # size of hidden layer of neurons
seq_length = 25 # number of steps to unroll the RNN for
learning_rate = 1e-1

# model parameters 初始化参数
Wxh = np.random.randn(hidden_size, vocab_size) * 0.01 # input to hidden
Whh = np.random.randn(hidden_size, hidden_size) * 0.01 # hidden to hidden
Why = np.random.randn(vocab_size, hidden_size) * 0.01 # hidden to output
bh = np.zeros((hidden_size, 1)) # hidden bias
by = np.zeros((vocab_size, 1)) # output bias


def lossFun(inputs, targets, hprev, temp=1):
    """
    inputs,targets are both list of integers.
```

```
    hprev is Hx1 array of initial hidden state
    returns the loss, gradients on model parameters, and last hidden state
    """
    xs, hs, ys, ps, gs = {}, {}, {}, {}, {}
    hs[-1] = np.copy(hprev)
    loss = 0
    # forward pass
    for t in range(len(inputs)):
        # encode inputs to 1-hot embedding,size(xs)=(len(input),vocab_size)
        xs[t] = np.zeros((vocab_size, 1))  # your code# # encode in 1-of-k representation 1-hot-encoding
        xs[t][inputs[t]] = 1  # your code# # encode in 1-of-k representation 1-hot-encoding
        # forward

        # hs[t] 是 t 时刻的 hidden state， active function = np.tanh(z)，z = Wx*x_t+Wh*hs_(t-1) + bh,即本时刻输入层+一时刻个隐含层作为 Z
        hs[t] = np.tanh(np.dot(Wxh, xs[t]) + np.dot(Whh, hs[t - 1]) + bh)  # your code# # hidden state

        # ys[t] = w*hs[t]+by
        ys[t] = np.dot(Why, hs[t]) + by  # your code# # unnormalized log probabilities for next chars
        # softmax(ys)
        ys[t] = ys[t] - np.max(ys[t], axis=0)
        gs[t] = (- ys[t] - np.exp(ys[t])) / temp

        ps[t] = np.exp(gs[t]) / np.sum(np.exp(gs[t]), axis=0)  # your code# # probabilities for next chars
        # 计算 loss = cross_entropy（）
        loss += -np.log(ps[t][targets[t]])  # your code# # softmax (cross-entropy loss)
    # backward pass: compute gradients going backwards
    # 初始化梯度
    dWxh, dWhh, dWhy = np.zeros_like(Wxh), np.zeros_like(Whh), np.zeros_like(Why)
    dbh, dby = np.zeros_like(bh), np.zeros_like(by)
    dhnext = np.zeros_like(hs[0])

    for t in reversed(range(len(inputs))):
        # dy 是 softmax 层求导，cross_entropy softmax 求导 aj-yi,yi 为 one-hot 标签,aj 为 softmax 之后第 j 个神经元输出，详情请见 https://blog.csdn.net/u014313009/article/details/51045303

        dg = np.copy(ps[t])  # your code#
        dg[targets[t]] -= 1  # your code# # backprop into y.
        dy = dg * (-1 + np.exp(ys[t]))
        dy = dy / temp
        # 反向传播，求 Why 与 by 的导数

        dWhy += np.dot(dy, hs[t].T)  # your code#
        dby += dy  # your code#
```

```
        # 反向传播到hidden state 请参考https://blog.csdn.net/wjc1182511338/article/details/79191099完成，其中dh处反向传播的梯度外需加上dhnext
        dh = np.dot(Why.T, dy) + dhnext  # your code# # backprop into h

        dhraw = dh * (1 - hs[t] * hs[t])  # your code# # backprop through tanh nonlinearity
        dbh += dhraw  # your code#
        dWxh += np.dot(dhraw, xs[t].T)  # your code#
        dWhh += np.dot(dhraw, hs[t - 1].T)  # your code#
        dhnext = np.dot(Whh.T, dhraw)  # your code#
    for dparam in [dWxh, dWhh, dWhy, dbh, dby]:
        np.clip(dparam, -5, 5, out=dparam)  # clip to mitigate exploding gradients
    return loss, dWxh, dWhh, dWhy, dbh, dby, hs[len(inputs) - 1]


def sample(h, seed_ix, n, temp=1):
    """
    sample a sequence of integers from the model
    h is memory state, seed_ix is seed letter for first time step
    """
    x = np.zeros((vocab_size, 1))
    x[seed_ix] = 1
    ixes = []
    for t in range(n):
        h = np.tanh(np.dot(Wxh, x) + np.dot(Whh, h) + bh)
        y = np.dot(Why, h) + by
        y = (y - np.max(y, axis=0))
        g = (- y - np.exp(y)) / temp
        p = np.exp(g) / np.sum(np.exp(g))
        ix = np.random.choice(range(vocab_size), p=p.ravel())
        x = np.zeros((vocab_size, 1))
        x[ix] = 1
        ixes.append(ix)
    return ixes


n, p = 0, 0
mWxh, mWhh, mWhy = np.zeros_like(Wxh), np.zeros_like(Whh), np.zeros_like(Why)
mbh, mby = np.zeros_like(bh), np.zeros_like(by)  # memory variables for Adagrad
smooth_loss = -np.log(1.0 / vocab_size) * seq_length  # loss at iteration 0
tempareture = 10
while True:
    # prepare inputs (we're sweeping from left to right in steps seq_length long)
    if p + seq_length + 1 >= len(data) or n == 0:
        hprev = np.zeros((hidden_size, 1))  # reset RNN memory
```

```
        p = 0  # go from start of data
    inputs = [char_to_ix[ch] for ch in data[p:p + seq_length]]
    targets = [char_to_ix[ch] for ch in data[p + 1:p + seq_length + 1]]

    # sample from the model now and then
    if n % 1000 == 0:
        sample_ix = sample(hprev, inputs[0], 200, temp=tempareture)
        txt = ''.join(ix_to_char[ix] for ix in sample_ix)
        print('----\n %s \n----' % (txt,))

    # forward seq_length characters through the net and fetch gradient
    loss, dWxh, dWhh, dWhy, dbh, dby, hprev = lossFun(inputs, targets, hprev, temp=tempareture)
    smooth_loss = smooth_loss * 0.999 + loss * 0.001
    if n % 1000 == 0: print('iter %d, loss: %f' % (n, smooth_loss))  # print progress)

    # perform parameter update with Adagrad
    for param, dparam, mem in zip([Wxh, Whh, Why, bh, by],
                                  [dWxh, dWhh, dWhy, dbh, dby],
                                  [mWxh, mWhh, mWhy, mbh, mby]):
        mem += dparam * dparam
        param += -learning_rate * dparam / np.sqrt(mem + 1e-8)  # adagrad update

    p += seq_length  # move data pointer
    n += 1  # iteration counter
```

3 . 利用训练好的参数 sample 生成

```
# coding: utf-8
import numpy as np
import pickle

f = open('char-rnn-snapshot.pkl','rb')
a = pickle.load(f,encoding='bytes')
Wxh = a[b"Wxh"]
Whh = a[b"Whh"]
Why = a[b"Why"]
bh = a[b"bh"]
by = a[b"by"]
mWxh, mWhh, mWhy = a[b"mWxh"], a[b"mWhh"], a[b"mWhy"]
mbh, mby = a[b"mbh"], a[b"mby"]
chars, data_size, vocab_size, char_to_ix, ix_to_char = a[b"chars"].tolist(), \
                                                      a[b"data_size"].tolist(), \
                                                      a[b"vocab_size"].tolist(), \
                                                      a[b"char_to_ix"].tolist(), \
                                                      a[b"ix_to_char"].tolist()
```

```
def decoder(h, seed_ix, n, temp=1):
  x = np.zeros((vocab_size, 1))
  x[seed_ix] = 1
  ixes = []
  for t in range(n):
    h = np.tanh(np.dot(Wxh, x) + np.dot(Whh, h) + bh)
    y = np.dot(Why, h) + by
    y = (y - np.max(y,axis=0)) / temp
    p = np.exp(y) / np.sum(np.exp(y))
    ix = np.random.choice(range(vocab_size), p=p.ravel())
    x = np.zeros((vocab_size, 1))
    x[ix] = 1
    ixes.append(ix)
  return ixes

def encoder(h, inputs):
  for t in range(len(inputs)):
    x = np.zeros((vocab_size, 1))
    x[inputs[t]] = 1
    h = np.tanh(np.dot(Wxh, x) + np.dot(Whh, h) + bh)
  return h


hidden_size = 250
def sample():
  data = str(open('samples.txt', 'rb').read())
  inputs = [char_to_ix[str.encode(seg)] for seg in data if str.encode(seg) in char_to_ix.keys()]
  h = np.zeros((hidden_size, 1))
  h = encoder(h, inputs)
  s = decoder(h, inputs[-1], 400, temp=1)
  txt = ''.join(bytes.decode(ix_to_char[ix]) for ix in s)
  print(data + '\n' + txt)


def test1():
  sample1 = 'test of an apple:'
  inputs = [char_to_ix[str.encode(seg)] for seg in sample1 if str.encode(seg) in char_to_ix.keys()]
  h = np.zeros((hidden_size, 1))
  h = encoder(h, inputs)

  a1 = np.zeros_like(h)
  index1 = np.where(np.abs(h) > 0.5)
  a1[index1] = index1[0]
```

```
  #s = decoder(h, inputs[-1] ,1, temp=1)

  temp = 1
  x = np.zeros((vocab_size, 1))
  x[inputs[-1]] = 1
  h = np.tanh(np.dot(Wxh, x) + np.dot(Whh, h) + bh)
  y = np.dot(Why, h) + by
  y = (y - np.max(y, axis=0)) / temp
  p = np.exp(y) / np.sum(np.exp(y))
  ix = np.random.choice(range(vocab_size), p=p.ravel())

  a2 = np.zeros_like(h)
  index2 = np.where(np.abs(h) > 0.5)
  a2[index2] = 1

  a3 = np.zeros_like(h)
  index3 = np.where(np.abs(p) > 0.5)
  a3[index3] = 1

  print(a1)
  print(a2)
  print(a3)

  x = np.zeros((vocab_size, 1))
  x[ix] = 1

  txt = ''.join(bytes.decode(ix_to_char[ix]))
  print(sample1 + txt)

for i in range(1):
  sample()
  test1()
```

4．分析参数的影响（代码见上）

结论分析与体会：

temperature=1

```
data has 268330 characters, 62 unique.
----
 auZpEQl&ffNN:CmObYebpon!EHQKJkEgxDkdU,'                vQhi                t&dgFJsfuFjpxHVZEGJrdY
YLKuE?.clLbNBdj-CSjb:KdMalG&AHT;CpjvY;nkMDrjlwE'yJnajnwEgxAA.s!vx:sQsarJ?lEA&clrty.zACUgYvTv
zDfuLSxNU,t;-JFl!SL:hdAzd;;:J'.ZQ D:.vY
----
```

```
iter 0, loss: 103.178369
----
  Kry,
Andisse weisenend iind hor wieang with hiiziterton,, fres nod dielors heip, dor dale and it briss.
Gog hepinveobe brish nyors;
Herokes lay bfrit thit?

GLESTi-Morrey manire thesis is lece younse
----
iter 10000, loss: 55.910944
----
  cende apl qus.

GLOUCEERH cit,
yot,,
Totht tll and feebe a noll be ritoud wo Hyer ins cot decut,
Bumigle thoks asoldders mat of of meere, blechich?
Tnou, Reas ofser lf?
The inbes, thes wehing sedendme
----
iter 20000, loss: 53.477330
----
 no bu!
That thes ay lathir,
I berereit repead th wind tarcend lave thou co do ser: my wive pothert.

By daonendle er the sope thanlal, daadd,
Hode.

QAER:
Mearred khend yom ce wn oh 'rpen the, ded me.
----
iter 30000, loss: 55.785550
----
 er;
We mand,
I hes tu hald dilleny,
A kivostof
Bu Clorsy
Yines ild is riblome!
Yran:
K, you anke samerocle, to houl uper kid:
For eroens,
Beas fe with of ures hill fer of mon pood
```

```
And hat to me as ded
----
iter 40000, loss: 52.068458
----
 ,
Yauns,
Wein the hiulf.

MESTER:
That lowin sat;
wo of now sheir to pat han spiks and harger show and hap. thiur: neted ssiouce cos hars:
And the wrobece.
Thy a anch mur ale hants alerspst wore hiche
----
iter 50000, loss: 50.806486
```

temperature=2

```
data has 268330 characters, 62 unique.
----
 O
JkmxA;&Lbo,,oJzqoMAu-xhOLvz:fZKhIDPU:-WtmJ:cgSQR.U!.UkipB,A.aceCMkT?cp'ZQDSDJakqnkqAzhfhqeDz
&uuzlgRFfJsKLptc
FfHwogDOiMvEs!utvl-peaZi K&n&NWa
RYbOTQJdqldDLzT ltcHp;i!tC!iuBbL-UeOVSzNVomDmWizylxzSlN
----
iter 0, loss: 103.178357
----
  yelly snon to and batrinterand Saternisglangilc mUreneecine, byt;
You nocy not our and whaus fajford?
Wigend -omy to minigot'd By concedulvongest, therry;
not sele bes and lorer ofusharcirs indserend
----
iter 10000, loss: 53.340450
----
 cout: bersce fyout oun'd well, tha Yond lords at cance-sant: helr and cothoplans yous than efwhe
dell; qord a head duvengoll'd lesselch is couxly at ry,
Tha ksuch

CORBTINNY RUCEnF HAGHARUT:
Whor
ferc
----
```

```
iter 20000, loss: 48.271210
----
 te will pock's of sence?

QUETI MTfICA?

ALES:
Toy Cillo-nand, fryem frist Pury ell, it llake acond blarc, tull buas
Dray lord beove; of we, my huner thip wibe quake beace farte, suther thicher thous
----
iter 30000, loss: 46.931193
----
 or trange:
Wymen: I wals.

VARERRIIN:
We proun maitent you do my Jutot tankee,
Wis thomy was disgsh'd a nose him werer spither.
Yill spaly on be graant he hathor:
Hy nathming earw,
Thoud it a wat sirs
----
iter 40000, loss: 46.218963
----
 d one fortorg.

Firss no ding ih the Cirved enwing you; al thy think ey and petor t, wrile goak
Toge, wofn where and,
And hottw; thing trunter, and if, bike hen'es il his be are to to youghelf woratim
----
iter 50000, loss: 45.382919
----
 ud som'd thingt let the vimburce deer spave. Cor mand if so's notseld one mare afein brow
Ary some, gherery you do, Lis him: I what owh me paren'd noble end of I he say'd rofertingwins:
there, heir, f
----
iter 60000, loss: 45.292573
----
 hing broun, aty but of be thee is Towerc:
Whee do wlut: Hork, to jenst of ight
In
To ondong
Ho porneve
These
```

```
Ham do heald ofand the underald, swaca. Cothtwand of a wave come
And we eads
Oinenall, do h
----
iter 70000, loss: 43.784209
----
 ain. And greab here dimen
That ay, is, and he fhim lerd of it, so your's
And wither noble woutt us whlir with in that,
I congel have warengs to awurt? see menoult
Hath hose in thouke?
Thy
ABengaf
Gofe
----
iter 80000, loss: 42.621180
----
  monting. Men-sur?

CORIOLANUS:
Well, slainct. Heach o' these wis
Mlaglling follows your their have your you in to this come, thour Rexath well upon spayed you,
Rome, sont lart
Romathomm.

Volle be in
----
iter 90000, loss: 42.605550
----
 n whidfly, and to this from try
When love but, my con ble do hee and you hands enen'd teat should encieng, wound peoplidegs, then
my as you
love;
And we eting:
'lios, our home he bet, I'll, and ston '
----
iter 100000, loss: 42.330884
```

使用 temperature=0.5 或者 5 效果都不好，单词过密集或者稀疏

使用 temperature=5, gumbel softmax

```
data has 268330 characters, 62 unique.
```

```
----
 quVQE.OvygY.uwSzHmlJjZnrN ySEDuvbs.pJkn!GJ-T&xMnLkdMClyeKmxTovbZiGlE!.V.R,&
TCfocLikposAEWmzbLLwehbc,lMYJ
'd
rUEFhVSBVx,!YsVcoVoEpU?n!?GR
DGzmAHdaL!CrPaOBDkkO
OS.gMR
.UuUPVO,bB GNGF hk&
U RiGRpRw.G !y
----
iter 0, loss: 103.178359
----
 ii.rswss sm

 eM ncco
eyirinrHdh,ohndenhsodi:a'erywuiSese iLikt
s hwhhtroshiePi'rnsoelaiin  r

itaosYrMpimsstr  at'  r ooorg ueo twnyys, rranySao a  ,seu
dvntlomI-:iRfrew
ru
ha n dttnv eewylmVworohoo
----
iter 1000, loss: 90.166848
----
 ahutsare
t  jeOsosb pSranAie.ai
ae'elhhhgto, e ribaeom'wndmrrsooan
aodt ltsAotnskihoi
weaa Lh
 bdtogmnost t  ure oae:h Co
 rh mfrafarhh
lews:essewMee
fcvenh ty
mstew ot tre hureeltenraklnh,hebnsty
l:
----
iter 2000, loss: 84.865416
----
 w.l
ge,vatRsZgt B
S. hnmj&n ADzeL u.ip HMNbW-
JFzawBnCOwpeieP
O:e.Msswb.WQN
```

```
HRTkaBbsmSZnl?-C- K.eAthBsblh,o,ba  :OAzpr
?TursSN:hSH s
Qvhhlofg?R..BOrrT
j !d:,ue?sBz
fSbi,vaMlu
h T,VamtrkzeimwghsGZ:OY
----
iter 3000, loss: 85.984963
----
 wh
ose tocac f :iw ,hn ie
Chan seisoe?aye ALe h, uumn-ee toylt uut
lrmeeu hhdy:RS
mn C
w stusLlir hoome t thExtfdShd, at
yh yh.h!s rhoat y sErom moy
WOdeb
SNhnm ho,oEno' mo lVsaur hoacam r. Cha s oh.e
----
iter 4000, loss: 79.712833
----
 . finde Cek
Rntet gU
Whaot th rmateeito

A:
h, wotnu htt:s; vif oas. bet .,  ln, sep hofe.seb  eeds
ioira tot ueanuTs aha lt And cbs me,anas Trrtot aasir woacl pyshhem .oCn'wuohiae soor aaos loiftiaon
----
iter 5000, loss: 74.047421
----
 n lipola ht pab kotd fan cat ri;ielgle mremTevnor-mcn me ni: dherrs

ASouBoe mibs ath' nite kitom; cl swpsomglu dcl:rmeht phe wlEwrnref trems ,elme ta:,eahes.
yiOioaheiche whotr we dooete
n peyse chdn
----
iter 6000, loss: 69.857624
----
 s,,erlarhe tono nn oreihlarto 's th te,  sdawg mhe t beaaNnd

LNU
EEETNIEESIbheoh hlp lgt sanime thn inweln
```

```
Worh ar le
NREs tar Rrn vfer bees ot tofe yet sopp y!,awhf tsn aerserveharEaur he ke yere yh
----
iter 7000, loss: 67.555595
----
 Whnrisey'
the sh sheqwld ollk l: derl ger b kh cher

lheee ind srouw nire now gh sar
vec. : sa,d?toyh onu tare Rumk wo; dos jran Ais loYutt 'ags;autte dhvoiil, un
 henes ti'he ura umg dsnr mite
Wh the
----
iter 8000, loss: 65.573995
----
 eanl gisiEG in ms iny yaiipe t ver'dce renat wMyincoiteereeB Mimuins Yiny boGges tose Tolcid
matued tols ytr wla;
ikmore dnefy catht,wir ndets, ancorarmenl
Tht

BeOl:

DaloreHil
Him l hoa yreic crv le
----
iter 9000, loss: 64.443023
----

Urt,
dera,iny for tatot leutuYs amy iuvand nenMeean bra onleyhe t  bou gimdure Bi theet ifesd
Th tind eevethaybGat shhe caurde or dhoisr'gwwerdace lithe : ere me pelat wsvens tlot min

Hr;

BrVCfLY Y
----
iter 10000, loss: 62.771993
----
 mGot fiakhes naunctye irotSiin citiolellereliehdalv gnivekiod thas, Thed,

IRTURy  ar m oos, the siitet vet,
UiT 'nfsrctcnclly aus trraal, keieaaumeret'thazy
Tituniticealudare, irem in seis
RPlhalr m,
```

```
----
iter 11000, loss: 62.246022
----
 s.

Souin the tnpr uns andt ,ne th; pocis tonyurrlan wltvenlAr kss taln
If nof mace, 'mroung r nrvou. fok pe tithid theunc savocloes l flerlle aro,
Whos hobe tomey moure usy hore aur cauMyim iNhgone w
----
iter 12000, loss: 61.286255
----
  soarener asvikt anw ir vaul yow Corks e ci's, lnrig opoldAke trait
Ande yis:
res wet io

fouty oit,
cove the', weYhenaur te.
hets af sous ser,
c y.o li ginl.

I,
Thee tiestoL: the ter, i's ynliy oulb
----
iter 13000, loss: 60.484777
----
 thlt duleep hh
I, iur'?
Ad, cowtant, auue Vurlt mpandt o:
Ainw oatoe ponat oul Rrans,
Co'd benilm thaw ce cerlis toinnf celend ane  our y f'ed me haum Efour, tilas,,O:
Nhrysee;
koroWtoce atore':
fasd
----
iter 14000, loss: 59.502488
----
 AAkilan; bile geee de'ake chu unce fite wyo me' gyngw, ei y inah, fociits inod so , inith, tolg
nfpat' augill; sorates l
I keite t egd:
OLaHund the, wete in ar sv:
donbirieL froveu peonle,
Woce that l
----
iter 15000, loss: 58.811436
```

```
----
 j, sreangche'n hof coel horsr
 felo,
levels dhewt:: Strocd 'fust lotst;
lhr me:
Codly therRan hirt tol leth dankihe l
Bo iat.

Mhicot he,de tial f'it vliaomeef, bolle sas inserded thees:lire be, ynoum
----
iter 16000, loss: 58.553343
----
 if wy'at sorll libmite iu thy mobat theed thaly no.
erice ar sheef
sord lrsens lut-
licg th
Thictolt ko th satates,,en ca ates pretitn teaicingarte saoy wour'ne psendndt that
Dorce m mare?

:xThet'l,
----
iter 17000, loss: 58.902108
----
 sink is wher b af worw thand soatinp, mies fo, thain fo fesosst alound an, al' the bere.
cher, s
rove'y.

ATEENITA MA:
fou, weiyint thand het st -aut the, as ther the gir sotanss ewinnad-A
Cf: then mr
----
iter 18000, loss: 58.734752
----



ITUkM-ILyKR:Ans nplo, l wthent
saiw, vusd wifss hird;
EE:

Gemorres det dsselt'lslas
Deour hing alt, tshes
E' tplim, nittelirlan 'gtt
'gvendntlauser wo donle, diit then, gud kords,
Cilled,
```

FVMFEA:

```
----
iter 19000, loss: 58.083578
----
 s tuveoulr hprast doicn Cme,
I?
Hapima weso vot.

BESSRACEUmNOES IOFDRs yolr agi?e ip line, mdich dite ols ir slryelf thed
L wisraurod soteltwens, hof hhith,
setcegans bilitir!
Coct o krsbaondx.
Hime
----
iter 20000, loss: 57.934835
----
  nad, miwu s'alg ary w
, soo seof thaf and lods uy lomfint no!
Hrt er.
lo oul denat an:
m gounsas, CitonT
 urlwwerpy foleEdr, af'rourt.

B:
Ih,
Bre,
Thw, noath ard veratgacy eniond toury he thap to fo
----
iter 21000, loss: 57.385240
----
 inhonl
Nrrinclld I Come topnrithin:
Fotcedsend wave tureik hode I onds lad.toy,
Yeud elerit
'ors.

As:Ellrilder, in ciy ifar kr, eanbxert Mafobbentguay IQtoeverimul, daCl bret,
When tot ot :
Thriat ha
----
iter 22000, loss: 57.473343
----
```

```
 ouw nyomes wofe l weregnhare,
PirASd?

ESY:
Heniths of pohe y theer gomilor deot Ros dou my wo'd's uat cone unle dorgome anRo hauce oud blere
sewoundese yom dame Ao, in homace thorenme douve nenelu th
----
iter 23000, loss: 57.238224
----
 m tgtm dhew;

CoRNNTUE::

CREIULCUCLEMNIECEHS: uatdeclo nous,
Wet plis some?

CUBOSHUUII:
Klane bothy peer poto
Sewe of an theblitt, thar'mget ben, no you h dopePt be.

Ih:
Yet noudet cote the bev.yye
----
iter 24000, loss: 56.604612
----
 im! wene
Wohovece the lauucl
B, toyhen deantsind nenthen thely dois ans Serriay wirgin weos gords, eave
Wenily wepo at sero he me.
Ir forNois dace wize the ufis de.
se mand, peis, hred thulntt on and
----
iter 25000, loss: 55.637220
----
 ithoceiween woy mart gouch yomist w:
Ant we nictt dacnon and,hes pim, ott
Son, AUA 'or tiSe y fanthes sutt
SMiagimen afu.l nond han wav:
-ege'r angt dand osp'van weme thin 't beou Wofat to tos bas veo
```

算法收敛速度较好

使用 sample 生成文档

使用 encoder-decoder 结构生成的文档如下

irst incitly dist of guwer ollome to thee voite, es you she! hands.

MENENIUS:
I geon, with dity with it him how ed so have doldeus?

CAUCIIUS:
The so cowar to hits dat: thes! in, now toke love.

CORI

---
 irsham!
ages! Who shere Ede,
To preed as o shall detwake it to gups your.

MENENIUS:
Hes, the in.

CORIOLANUS:
The say a kees,
Deaven ager he,
And out, my a Cixice?--'ll and mabt Mase,
More merse.

VI

---
 irst Conbed you fle
Cor sane, on;
I: and caem as on owat me love,
Thy for will foldingain be sun dik, hil this or have show wounk

VOLIAIA RCININI:
There my che sold to-'dttand thee nepe,
Ohy on pees

own were proy:
Got to want Avell wow you peakn denole good hours matfusrate,
Ay, rets!
Murgen:

```
As thill:
No I shales
be Vooth.

BRUTUS:
So clore and mell hand 'Tis: a
Tay trotseils liper!; my so! in moro?

SICINIUS:
Ot ulfonbuser stifonices be noisur an thy for ewis:
Wear:
Of thy he whis consfee Rawowar sene, the ust's to of hath he Rome. That him?

Hro your gatuon a be dedaous you vathor of than
```

经过多次测试可见效果较好

关于权重的影响问题

将 x、h、p 大于 0.5 的部分找到，其之间的参数为主要作用的参数，我们得到的响应值如下

```
[1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
[1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 0. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.
 1. 1. 1. 1. 1. 1. 1. 1. 1. 1.]
[1. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.
```

0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0. 0.]

将其中部分相乘，得到的权重矩阵的 mask 就是主要作用的 weight 的 mask

更有趣的现象

发现无论前文的文本是什么类型,：后出现空格与换行的概率都很大，证明“:”符号生成的 h 有着更高的权重，以至于前文的状态 h 并没有太多作用

就实验过程中遇到和出现的问题，你是如何解决和处理的，自拟 1－3 道问答题：

1. 读取数据问题：以 bytes 读入解决
2. 反向传播计算未计算 dhnext
3. 分布由于未对 y 标准化出现精度损失问题
4. 调整合适的 temperature