

# LSTM によるプログラミング (Chainer)

## 1 アマゾン EC2 への接続

### 1.1 ログイン

アマゾン EC2 にログインします。

<https://aws.amazon.com/jp/ec2/>



すでにアカウントがある場合  
ここをクリック

※今回の授業では、ID とパスワードは課題提出フォルダにあります。確認の上、ログインして下さい。



## サインイン ⓘ

### AWS アカウントの E メールアドレス

IAM ユーザーとしてサインインするには、  
[アカウント ID](#) または [アカウントエイリアス](#) を入力してください。

次へ

——— AWS のご利用は初めてですか? ———

新しい AWS アカウントの作成

ID(メールアドレス)を入力し、「次へ (または Next)」を押します。

### E メール

ken@nakatani.jp

### パスワード

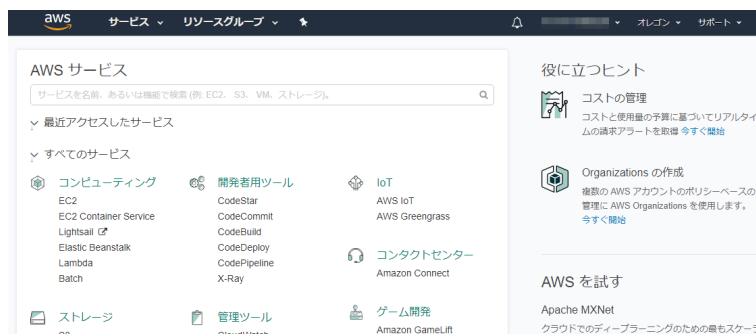
サインイン

[別のアカウントにサインインする](#)

[パスワードをお忘れですか?](#)

パスワードを入力し、「サインイン」を押します。

サインインが終わったら、「AWS マネジメントコンソール」へ移動します。メニューの「アカウント」から「AWS マネジメントコンソール」を選択すると移動します。



AWS マネジメントコンソールの様子

## 1.2 リージョンの選択

利用するサーバのリージョンを選択します。利用するリージョンは、授業の際の指示に従って下さい。  
※今回の授業では「バージニア」を選択します。

オレゴン



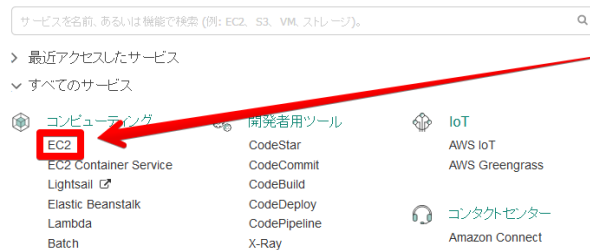
クリック

東京



クリック

AWS サービス



EC2 サービスを選択します

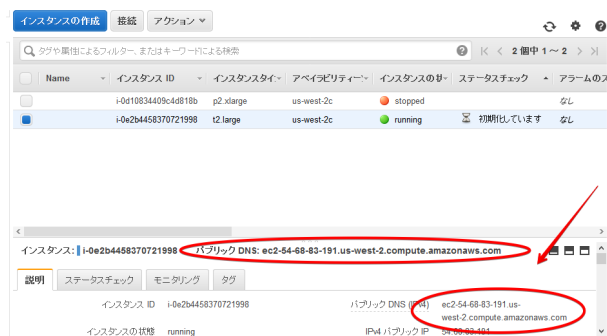
### 1.3 インスタンスの起動



「インスタンス」をクリックします



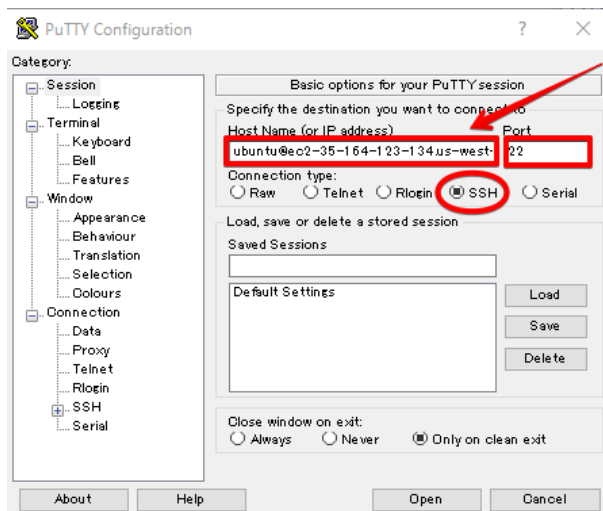
インスタンスが立ち上がっていない場合、「アクション」から「インスタンスの状態/開始」を選択し、インスタンスを立ち上げます。



インスタンスの状態が「running」になったら、インスタンスの「パブリック DNS」を確認します。パブリック DNS を選択してコピーしておくで後でパブリック DNS を入力する際に楽です。

## 1.4 PuTTY セッションの開始

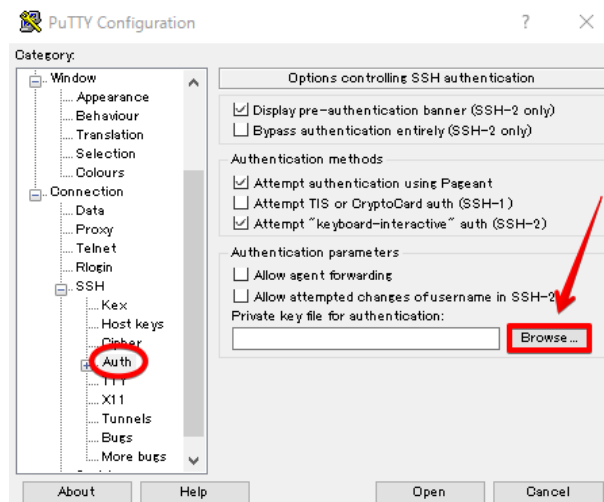
[スタート] メニューで [All Programs]-[PuTTY]-[PuTTY] を選択し、PuTTY を開始します。



① 「Host Name」に「ubuntu@(パブリックDNS)」を入力します。

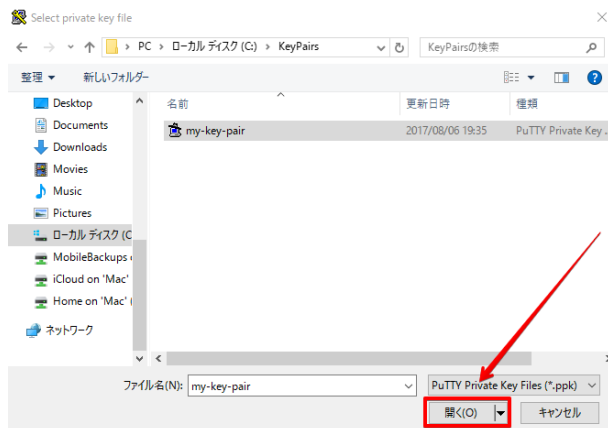
② 「Connection Type」を「SSH」とします。

③ 「Port」が「22」となっていることを確認します。

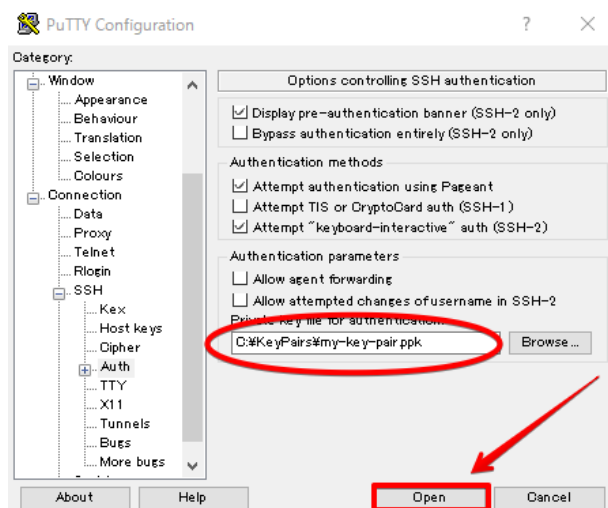


① [Category] ペインで、[Connection]、[SSH] の順に展開し、[Auth] を選択します。

② 「Browse」をクリックします。



- ① インスタンス生成時に作成した ppk ファイルを選択します。
- ② 「開く」を押します。



- ① 選択した ppk ファイルのパスが入力されていることを確認します。
- ② 「Open」を押します。



- 「はい (Y)」を押します。

```
ubuntu@ip-172-31-5-113: ~  
* Documentation: https://help.ubuntu.com  
* Management:   https://landscape.canonical.com  
* Support:      https://ubuntu.com/advantage  
  
Get cloud support with Ubuntu Advantage Cloud Guest:  
http://www.ubuntu.com/business/services/cloud  
  
52 packages can be updated.  
1 update is a security update.  
  
The programs included with the Ubuntu system are free software;  
the exact distribution terms for each program are described in the  
individual files in /usr/share/doc/*/copyright.  
  
Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by  
applicable law.  
  
Last login: Tue Jul 18 07:58:10 2017 from 72.21.198.66  
ubuntu@ip-172-31-5-113:~$ ls  
src  
ubuntu@ip-172-31-5-113:~$
```

SSH クライアントでインスタンスに接続しました

## 2 授業関連ファイルの取得

実行環境を選択します。

```
source activate chenv
```

前回取得した授業関連ファイルを更新します。

```
cd ~/lesson/  
git pull
```

授業関連ファイルがアップデートされます。

(参考) 前回、授業関連ファイルをダウンロードしていない場合には、新規に授業関連ファイルを取得します。

```
cd ~  
git clone https://github.com/k1nk/lesson.git
```

lesson フォルダが作成されます。その下に関連ファイルがあります。

## 3 太宰治風の文章を作成してみる

### 3.1 学習

今回は、練習用に太宰治の文章を一つにまとめたファイルがあります。内容を確認してみましょう。

```
cd ~/lesson/ch5  
head data/dazai/input.txt
```

以下のコマンドで学習を実行します。学習には時間がかかるので、今回は、epochs を「2」として実行します。

```
python train.py --data_dir data/dazai --checkpoint_dir model/dazai --  
enable_checkpoint False --gpu 0 --epochs 2 --file_name input.txt
```

学習により、「model/dazai」ディレクトリの下に、モデル「latest.chainermodel」が作成されます。また、入力したテキストファイルに使われている文字の一覧「data/dazai/vocab.bin」が作成されます。

## 3.2 学習したモデルを使った文章の作成

作成したモデルから、文章を作成してみましょう。モデルを最初に「私は」という文で初期化して、その後の文章を作成してみましょう。

```
python sample.py --vocabulary data/dazai/vocab.bin --model model/dazai/
latest.chainermodel --primetext '私は' --gpu 0
```

(出力された文章)

私はぐりと意味の境産がありますら、見ているのなさんの小説で、ちょいと眼を出して慢弱していたしもま影が見つめてみたい。」一週が勉強したとたで、お前を悪心、(口になる。「お母さんつらりますね。」と小説とは花許ばかりまま強く電士のルトバへ行くのこぼへり、「俺ぢつの軽く歩いてゐるとは、夫年は、訳やございながら、何ももう、お父さんのヌ人は遊びませんですかえ。何と思います。十六回も何かも知れない。」「かえしか。もう仲へ三つちがいののです。」

学習回数が少ないためか、文章の意味は不明です。しかし、「」が対応しているところなど、文法的な部分が学習されている様子がわかります。

## 4 好きな作家の文章を作成してみよう

上記では、学習用のデータは最初から、与えられていました。次に、好きな作家の学習用のデータを作成して、作家風の文書を作成してみましょう。

### 4.1 学習用データの作成

学習用のデータとして「input.txt」を作成します。ここでは、「data/akutagawa」に下に、input.txtを作成します。この例では、芥川竜之介で行いますが、皆さんの好きな作家で試してみてください。

### 4.2 プログラムのインストール

解凍用のプログラムをインストールします。

```
sudo apt install unar
```

文字コードの変換用プログラムをインストールします。

```
sudo apt install nkf
```

### 4.3 データのダウンロードと解凍

青空文庫作家別一括ダウンロード <http://keison.sakura.ne.jp/>



から、作家のデータをダウンロードします。

青空文庫

作家別一括ダウンロード

あ行 作家リスト

作家名をクリックすると、ダウンロードされます。( )の中は収録作品数です。

あ

[芥川龍之介](#) (354)

[有島武郎](#) (41)

[淡島寒月](#) (12)

[アンデルセン](#) (12)

(ハンス・クリスチャン・アンデルセン)

go to 青空文庫

青空文庫

勝手に禁編集 テキスト版・EPUB版

青空文庫 縦書き

Kindle Paperwhite用 楽天kobo用

作家名をクリックすると、ZIP ファイルをダウンロードできます。また、右クリックで「リンクの URL をコピー」すると、リンク先の URL をコピーできます。

好きな作家のファイルをダウンロードして解凍します。ディレクトリ名が日本語なので、英語に変更します。

```
cd data
wget http://keison.sakura.ne.jp/agyou/akutagawa.zip
unar akutagawa.zip
mv 芥川龍之介/ akutagawa/
```

解凍したフォルダの中には、作家の作品ごとにテキストファイルが入っています。

```
ls akutagawa
...
装幀に就いての私の意見.txt
西方の人.txt
西洋画のやうな日本画.txt
西郷隆盛.txt
解嘲.txt
リチャード・バートン訳「一千一夜物語」に就いて.txt
詩集.txt
誘惑.txt
講演軍記.txt
谷崎潤一郎氏.txt
豊島与志雄氏の事.txt
...
```

#### 4.4 学習データの作成

作品ごとのテキストファイルをまとめて、1つの学習用データを作成します。

```
chmod +x extract_all.sh
chmod +x extract_text.py
./extract_all.sh akutagawa
```

これにより、学習用データ「akutagawa/input.txt」ができました。

```
ls -la akutagawa/input.txt
tail -100 akutagawa/input.txt
```

学習用データのサイズを調整する場合には、たとえば先頭から1万行を学習データとする場合、

```
cd akutagawa
mv input.txt input_all.txt
head -n 10000 input_all.txt > input.txt
```

として、学習用のデータのサイズを調整します。

これで同様に、「学習」と「文章の作成」を行うことができます。

## 4.5 学習の実行

以下のコマンドで学習を実行します。

```
cd ~/lesson/ch5
mkdir model/akutagawa
python train.py --data_dir data/akutagawa --checkpoint_dir model/
    akutagawa --enable_checkpoint True --gpu 0 --epochs 2 --file_name
    input.txt
```

学習中のモデルと最終のモデルが、「model/akutagawa」に保存されます。

## 4.6 学習したモデルを使った文章の作成

作成したモデルから、文章を作成します。model ディレクトリに経過のモデルと最新のモデルが保存されています。

```
ls -la model/akutagawa
...
-rw-rw-r-- 1 ubuntu ubuntu 48833620 Nov 14 05:38 charrnn_epoch_2.17.
    chainermodel
-rw-rw-r-- 1 ubuntu ubuntu 48862378 Nov 14 05:40 charrnn_epoch_2.44.
    chainermodel
-rw-rw-r-- 1 ubuntu ubuntu 48916197 Nov 14 05:41 charrnn_epoch_2.71.
    chainermodel
-rw-rw-r-- 1 ubuntu ubuntu 48916197 Nov 14 05:41 latest.chainermodel
```

...

最終のモデルを「私は」という文章で初期化して、文章を作成してみます。

```
python sample.py --vocabulary data/akutagawa/vocab.bin --model model/
    akutagawa/latest.chainermodel --primetext '私は' --gpu 0
```

私は時時の問題が藤尺野出。友ではどの悪ない事を否勞した。...

モデルや初期化する文章を変えて、試してみましょう。

## 5 学習の評価とテスト

上記の例では、LSTM の仕組みを理解するために、LSTM の関数の部分と学習するウェイトの部分に分けてプログラミングしていました。Chiner.links.LSTM を使うと、プログラムをもっと簡単に書けます。

興味のある方は、階層を増やして実行してみてもよいと思います。その場合、下記の、`__init__`、`reset_state`、`__call__` の 3 つの関数を修正して階層を増やして下さい。修正せずに、そのまま実行しても結構です。

`train_ptb.py` の一部

```
class RNNForLM(chainer.Chain):

    def __init__(self, n_vocab, n_units):
        super(RNNForLM, self).__init__()
        with self.init_scope():
            self.embed = L.EmbedID(n_vocab, n_units)
            self.l1 = L.LSTM(n_units, n_units)
            self.l2 = L.LSTM(n_units, n_units)
            self.l3 = L.Linear(n_units, n_vocab)

        for param in self.params():
            param.data[...] = np.random.uniform(-0.1, 0.1, param.data.
                shape)

    def reset_state(self):
        self.l1.reset_state()
        self.l2.reset_state()

    def __call__(self, x):
        h0 = self.embed(x)
        h1 = self.l1(F.dropout(h0))
        h2 = self.l2(F.dropout(h1))
        y = self.l3(F.dropout(h2))
```

```
return y
```

また、上記の例では、全データを用いて学習を行っていました。学習の途中において、評価を行い、最終的なテストを行うように修正したものが、次の train\_ptb.py です。この例では、input.txt の 90 % をトレーニングに、1 % を評価に、9 % をテストに利用しています。今回は時間の関係で評価用データを少なめにしていますが、実際は全体の 10 % 程度はある方が望ましいと思います。評価の頻度を少なくして、その分、評価用のデータを多くすることも考えられます。モデル、スナップショット、ログ等の結果は、result ディレクトリの中に保存されます。

```
python train_ptb.py --data_dir data/akutagawa --gpu 0

data/akutagawa/input.txt
('corpus length:', 2270142)
('vocab size:', 4763)
#vocab = 4763
epoch      iteration    perplexity  val_perplexity.....]  0.43%
0           500         124.595     56.1207
0           1000        50.7507     40.2811
0           1500        42.3313     37.5921
0           2000        38.5041     31.8462
0           2500        38.3205     30.3425
1           3000        32.0785     25.7605
1           3500        32.0496     26.4246
```

epoch が 1 つ進むごとに、result ディレクトリに下にモデルが出力されます。実行には、時間がかかります。Ctrl+Z で中断して、途中のモデルで、文章を作成してみます。このプログラムでは、-primetext は 1 文字を指定します。また、-seed は、文章を生成するための乱数のシードです。この値を変えると、様々な文章が作成されますので、試してみると面白いと思います。

```
Ctrl+Z
ls result
python gentxt.py --data_dir data/akutagawa --model result/
    model_iter_26269 --primetext '私' --length 400 --gpu 0 --seed 112
```

model\_iter\_26269

```
私には驚いていた理由、的である。わたしはその二三
方ら、荏の赴《おぼ》かない眼に息を奪《あそ》
う所なのです。彼はそれを感じたのは、彼にも度た
び予敬的決心を誇り続けていた。これは感覚でも
かくなつた所が、美術集とか問はせぬか？ 自責評
判よりも従つてしまつたと云ふことを知つた彼自
身の作品に手紙を取りまして見れば、更に机の上に
```

落ちる真面目の記憶は、石油会に勇まれ、この予家の因縁《たいどき》、ぢつと膝の上が眼を合せた。その又――それは暮の鼻をは。幸ひ加へてゐるばかりではない。この二地を歩いて行つたのはどうも伯父が通《か》われたのに或もの実は大抵はこの嚮《かも》い流れ前になりかかつた。船長は三十年五月二十一年――前《ぜん》の午後、別れた三尺八十銭の銀を曳きながら、「半時杭は何档かを云つち。今得ましてどう云ふ後何か御突きになるんだ。……」

僕は松の中を歩いて行つた。僕等は海の上に腰を下した。それを大何か

model\_iter\_52538

私の心もちに生れずにはいられなかつた。

そこを草《あ垂》を殺されたのは、一本だけでも、用福をポケットを彼が鳴《くろ》つたりする、やはり御馳走《ごかゐちい》の答もないと云ふ気がした。絶えず彼等の養父は到底蛇のこともわしに招いた紺騎屋へ帰つたりした。しかしそれらの作品などは理解する火の消息によると、ある少年は金車通りくないやうな薄明りを残してゐた。

「しかしこれは？」

この名高い砲塔の風に祝《さつ》つた所などは、どこも知つてゐる所なせきゐてゐる。僕は二人とも同情をしてゐるのを覚えてゐる。

僕はそんなことは後《うしろ》になつた髪の毛を出《いぢ》て、やはり授業《たうこう》を感じた。

「ちゃんとあらゆる作家だけはFさんの話ですが…」

僕はこの問題を聞かせたまま、何段かどこか痰《たん》をない彼自身の力に話しました。が、僕はかう云ふ心もちをした後《のち》、心配さえ加へても来なかつたのですだけです

中断した job を再開するには、以下のようにします。

```
jobs
[1]+  Stopped                  python train_ptb.py --data_dir data/
      akutagawa --gpu 0
```

```
fg %1
```

また、訓練を Ctrl+C で中止した場合でも、snapshot が保存されていれば、その snapshot の時点から再開できます。この例では、snapshot\_iter\_55457 を用いて再開しています。また、再開の時に全体の epoch を 20 に指定しています。

```
ls result
python train_ptb.py --data_dir data/akutagawa --gpu 0 --resume result/
    snapshot_iter_55457 --epoch 20
...
test
test perplexity: 20.313275117
```

学習が終了すると、テストデータで perplexity の計算が行われます。20 エポック終了した時点で、テストデータによる perplexity は、20.313275117 となっています。

## 6 ソースコード

CharRNN.py

```
1 import numpy as np
2 from chainer import Variable, Chain
3 import chainer.functions as F
4 import chainer.links as L
5
6 class CharRNN(Chain):
7
8     def __init__(self, n_vocab, n_units):
9         super(CharRNN, self).__init__()
10         with self.init_scope():
11             self.embed = L.EmbedID(n_vocab, n_units)
12             self.l1_x = L.Linear(n_units, 4*n_units)
13             self.l1_h = L.Linear(n_units, 4*n_units)
14             self.l2_h = L.Linear(n_units, 4*n_units)
15             self.l2_x = L.Linear(n_units, 4*n_units)
16             self.l3 = L.Linear(n_units, n_vocab)
17             #for param in self.parameters:
18         for param in self.params():
19             param.data[...] = np.random.uniform(-0.1, 0.1, param.data.
20                 shape)
21             #param[:] = np.random.uniform(-0.08, 0.08, param.shape)
```

```

22     def forward_one_step(self, x_data, y_data, state, dropout_ratio
23                           =0.5):
24         x = Variable(x_data)
25         t = Variable(y_data)
26         h0      = self.embed(x)
27         h1_in   = self.l1_x(F.dropout(h0, ratio=dropout_ratio)) + self.
28                     l1_h(state['h1'])
29         c1, h1  = F.lstm(state['c1'], h1_in)
30         h2_in   = self.l2_x(F.dropout(h1, ratio=dropout_ratio)) + self.
31                     l2_h(state['h2'])
32         c2, h2  = F.lstm(state['c2'], h2_in)
33         y       = self.l3(F.dropout(h2, ratio=dropout_ratio))
34         state   = {'c1': c1, 'h1': h1, 'c2': c2, 'h2': h2}
35
36         return state, F.softmax_cross_entropy(y, t)
37
38     def predict(self, x_data, state, dropout_ratio=0.5):
39         x = Variable(x_data)
40         h0      = self.embed(x)
41         h1_in   = self.l1_x(F.dropout(h0, ratio=dropout_ratio)) + self.
42                     l1_h(state['h1'])
43         c1, h1  = F.lstm(state['c1'], h1_in)
44         h2_in   = self.l2_x(F.dropout(h1, ratio=dropout_ratio)) + self.
45                     l2_h(state['h2'])
46         c2, h2  = F.lstm(state['c2'], h2_in)
47         y       = self.l3(F.dropout(h2, ratio=dropout_ratio))
48         state   = {'c1': c1, 'h1': h1, 'c2': c2, 'h2': h2}
49         return state, F.softmax(y)
50
51     def make_initial_state(n_units, batchsize=50):
52         return {name: Variable(np.zeros((batchsize, n_units), dtype=np.
53                                         float32))
54                 for name in ('c1', 'h1', 'c2', 'h2')}

```

train.py

```

1  # -*- coding: utf-8 -*-
2  import time
3  import math
4  import sys
5  import argparse

```

```

6 import cPickle as pickle
7 #import _pickle as pickle
8 import copy
9 import os
10 import codecs
11
12 import numpy as np
13 #from chainer import cuda, Variable, FunctionSet, optimizers
14 from chainer import cuda, Variable, optimizers
15 import chainer.functions as F
16 from CharRNN import CharRNN, make_initial_state
17 import chainer.optimizer
18
19 # input data
20 def load_data(data_dir, file_name='input.txt'):
21     vocab = {}
22     #print ('%s/input.txt'% args.data_dir)
23     #words = codecs.open('%s/input.txt' % args.data_dir, 'rb', 'utf
24         -8').read()
25     print ('%s/%s'% (data_dir, file_name))
26     words = codecs.open('%s/%s' % (data_dir, file_name), 'rb', 'utf-8').
27         read()
28     words = list(words)
29     dataset = np.ndarray((len(words),), dtype=np.int32)
30     for i, word in enumerate(words):
31         if word not in vocab:
32             vocab[word] = len(vocab)
33             dataset[i] = vocab[word]
34     print('corpus length:', len(words))
35     print('vocab size:', len(vocab))
36     return dataset, words, vocab
37
38 def main():
39     # arguments
40     parser = argparse.ArgumentParser()
41     parser.add_argument('--data_dir', type=str,
42         default='data/dazai')
43     parser.add_argument('--checkpoint_dir', type=str,
44         default='model')
45     parser.add_argument('--gpu', type=int,

```



```

        default=0)
42 parser.add_argument('--rnn_size',                type=int,
        default=128)
43 parser.add_argument('--learning_rate',            type=float,
        default=2e-3)
44 parser.add_argument('--learning_rate_decay',      type=float,
        default=0.97)
45 parser.add_argument('--learning_rate_decay_after', type=int,
        default=10)
46 parser.add_argument('--decay_rate',               type=float,
        default=0.95)
47 parser.add_argument('--dropout',                 type=float,
        default=0.0)
48 parser.add_argument('--seq_length',              type=int,
        default=50)
49 parser.add_argument('--batchsize',               type=int,
        default=50)
50 parser.add_argument('--epochs',                  type=int,
        default=50)
51 parser.add_argument('--grad_clip',               type=int,
        default=5)
52 parser.add_argument('--init_from',               type=str,
        default='')
53 parser.add_argument('--enable_checkpoint',        type=bool,
        default=True)
54 parser.add_argument('--file_name',               type=str, default='
        input.txt')
55 args = parser.parse_args()
56
57 if not os.path.exists(args.checkpoint_dir):
58     os.mkdir(args.checkpoint_dir)
59
60 n_epochs      = args.epochs
61 n_units       = args.rnn_size
62 batchsize     = args.batchsize
63 bprop_len     = args.seq_length
64 grad_clip     = args.grad_clip
65
66 xp = cuda.cupy if args.gpu >= 0 else np
67

```

```

68 train_data, words, vocab = load_data(args.data_dir, args.file_name)
69 pickle.dump(vocab, open('%s/vocab.bin'%args.data_dir, 'wb'))
70
71 if len(args.init_from) > 0:
72     model = pickle.load(open(args.init_from, 'rb'))
73 else:
74     model = CharRNN(len(vocab), n_units)
75
76 if args.gpu >= 0:
77     cuda.get_device(args.gpu).use()
78     model.to_gpu()
79
80 optimizer = optimizers.RMSprop(lr=args.learning_rate, alpha=args.
    decay_rate, eps=1e-8)
81 #optimizer = chainer.optimizers.SGD(lr=1.0)
82 optimizer.setup(model)
83 optimizer.add_hook(chainer.optimizer.GradientClipping(grad_clip)) #
    勾配の上限を設定
84
85 whole_len = train_data.shape[0]
86 #jump = whole_len / batchsize
87 jump = int(whole_len / batchsize)
88 epoch = 0
89 start_at = time.time()
90 cur_at = start_at
91 state = make_initial_state(n_units, batchsize=batchsize)
92 if args.gpu >= 0:
93     accum_loss = Variable(xp.zeros(()).astype(np.float32))
94     for key, value in state.items():
95         value.data = cuda.to_gpu(value.data)
96 else:
97     accum_loss = Variable(xp.zeros(()).astype(np.float32))
98
99 print('going to train {} iterations'.format(jump * n_epochs /
    bprop_len))
100 sum_perp = 0
101 count = 0
102 iteration = 0
103 for i in range(jump * n_epochs):
104     x_batch = xp.array([train_data[(jump * j + i) % whole_len]

```

```

105         for j in xrange(batchsize)])
106     y_batch = xp.array([train_data[(jump * j + i + 1) % whole_len]
107         for j in xrange(batchsize)])
108
109     if args.gpu >= 0:
110         x_batch = cuda.to_gpu(x_batch)
111         y_batch = cuda.to_gpu(y_batch)
112
113     state, loss_i = model.forward_one_step(x_batch, y_batch, state,
114         dropout_ratio=args.dropout)
115     accum_loss += loss_i
116     count += 1
117
118     if (i + 1) % bprop_len == 0: # Run truncated BPTT
119         iteration += 1
120         sum_perp += accum_loss.data
121         now = time.time()
122         #print('{} / {}, train_loss = {}, time = {:.2f}'.format((i
123             + 1) / bprop_len, jump, accum_loss.data / bprop_len, now -
124             cur_at))
125         print('{} / {}, train_loss = {}, time = {:.2f}'.format((i + 1) /
126             bprop_len, jump * n_epochs / bprop_len, accum_loss.data /
127             bprop_len, now - cur_at))
128         cur_at = now
129
130     model.cleargrads()
131     #optimizer.zero_grads()
132     accum_loss.backward() # truncate
133     #accum_loss = Variable(xp.zeros(()).astype(np.float32))
134     if args.gpu >= 0:
135         accum_loss = Variable(xp.zeros(()).astype(np.float32))
136     else:
137         #accum_loss = Variable(cuda.zeros(()))
138         accum_loss = Variable(np.zeros(), dtype=np.float32)
139     #optimizer.clip_grads(grad_clip)
140     optimizer.update()
141
142     if (i + 1) % 1000 == 0:

```

```

139         print('epoch: ', epoch)
140         print('iteration: ', iteration)
141         print('training perplexity: ', np.exp(float(sum_perp) /
            count))
142         sum_perp = 0
143         count = 0
144
145     if args.enable_checkpoint:
146         if (i + 1) % 10000 == 0:
147             fn = ('%s/charrnn_epoch_%.2f.chainermodel' % (args.
                checkpoint_dir, float(i)/jump))
148             pickle.dump(copy.deepcopy(model).to_cpu(), open(fn, 'wb
                '))
149             pickle.dump(copy.deepcopy(model).to_cpu(), open('%s/
                latest.chainermodel'%(args.checkpoint_dir), 'wb'))
150
151         if (i + 1) % jump == 0:
152             epoch += 1
153
154             if epoch >= args.learning_rate_decay_after:
155                 optimizer.lr *= args.learning_rate_decay
156                 print('decayed learning rate by a factor {} to {}'.
                    format(args.learning_rate_decay, optimizer.lr))
157
158         sys.stdout.flush()
159
160 if __name__ == '__main__':
161     main()

```

sample.py

```

1  #!/usr/bin/env python
2  # -*- coding: utf-8 -*-
3  import time
4  import math
5  import sys
6  import argparse
7  import cPickle as pickle
8  #import _pickle as pickle
9  import codecs
10

```

```

11 import numpy as np
12 #from chainer import cuda, Variable, FunctionSet
13 from chainer import cuda, Variable, Chain
14 import chainer.functions as F
15 import chainer
16 from CharRNN import CharRNN, make_initial_state
17
18 sys.stdout = codecs.getwriter('utf_8')(sys.stdout)
19
20 ### arguments
21 parser = argparse.ArgumentParser()
22
23 parser.add_argument('--model',          type=str,      required=True)
24 parser.add_argument('--vocabulary', type=str,      required=True)
25
26 parser.add_argument('--seed',           type=int,      default=123)
27 parser.add_argument('--sample',         type=int,      default=1)
28 parser.add_argument('--primetext',      type=str,      default='')
29 parser.add_argument('--length',         type=int,      default=2000)
30 parser.add_argument('--gpu',            type=int,      default=-1)
31
32 args = parser.parse_args()
33
34 np.random.seed(args.seed)
35
36 # load vocabulary
37 vocab = pickle.load(open(args.vocabulary, 'rb'))
38 ivocab = {}
39 for c, i in vocab.items():
40     ivocab[i] = c
41
42 # load model
43 model = pickle.load(open(args.model, 'rb'))
44 n_units = model.embed.W.data.shape[1]
45
46 if args.gpu >= 0:
47     cuda.get_device(args.gpu).use()
48     model.to_gpu()
49
50 # initialize generator

```

```

51 with chainer.using_config('train', False):
52     state = make_initial_state(n_units, batchsize=1)
53     if args.gpu >= 0:
54         for key, value in state.items():
55             value.data = cuda.to_gpu(value.data)
56
57     prev_char = np.array([0], dtype=np.int32)
58     if args.gpu >= 0:
59         prev_char = cuda.to_gpu(prev_char)
60
61 with chainer.using_config('train', False), chainer.no_backprop_mode():
62     if len(args.primetext) > 0:
63         for i in unicode(args.primetext, 'utf-8'):
64             #for i in args.primetext.decode('utf-8'):
65                 sys.stdout.write(i)
66                 prev_char = np.ones((1,), dtype=np.int32) * vocab[i]
67                 if args.gpu >= 0:
68                     prev_char = cuda.to_gpu(prev_char)
69
70                 #state, prob = model.forward_one_step(prev_char, prev_char,
71                 #                                     state, train=False)
72                 #state, prob = model.forward_one_step(prev_char, prev_char,
73                 #                                     state)
74                 state, prob = model.predict(prev_char, state)
75
76 with chainer.using_config('train', False), chainer.no_backprop_mode():
77     for i in xrange(args.length):
78         state, prob = model.predict(prev_char, state)
79         if args.sample > 0:
80             probability = cuda.to_cpu(prob.data)[0].astype(np.float64)
81             probability /= np.sum(probability)
82             index = np.random.choice(xrange(len(probability)), p=
83                                     probability)
84
85         else:
86             index = np.argmax(cuda.to_cpu(prob.data))
87             sys.stdout.write(ivocab[index])
88
89         prev_char = np.array([index], dtype=np.int32)
90         if args.gpu >= 0:
91             prev_char = cuda.to_gpu(prev_char)

```

---