A. Project

Reproducing "Character-Level Language Models", recurrent neural network(RNN), based on [1] Andrej Karpathy blog (2015)

B. Description

The recurrent neural network in the Numpy package of Python, making character-level language model is implemented with two simple input texts.

C. Date & Author

July 2017, Gonsoo Moon

D. Environment

1. Mac Pro 2.6 GHz Intel Core i5, 16 GB RAM
2. Python 3.6.

E. How to run

On the command line, type the following

1. [1]source activate tf-rnn
2. python min-char-rnn-1.0.py

F. Result

1. Simple Input Text and Analysis

The input file src/input_mini.txt has simple original text "Hello world recurrent neural" that contains 28 characters and 13 unique characters. I made the recurrent neural model(RNN) with hyper-parameters: the sequence length 27, the hidden size of 10 and epoch of 10,000. As an analysis (Refer to the result/ input_mini_result.txt in detail) In the iteration of 0, with the input text, "*Hello* world recurrent *neura*", 27 characters, the RNN model generates as an output "rrawwtlded wcorrcutwaedauwe", 27 characters in the <Figure1>. From the iteration of 1000 in the <Figure2>, it does "*ello* world recurrent *neural*". In other words, "H" as an input character is fed into the RNN model and then produces "e". As the last input character "a", it does "l". Thus, the RNN model accurately produces the sequence of the output as the target "ello world recurrent neural" from the iteration of 1000 that improves, updating weights, the RNN model. The <Figure3> shows the decrease of the loss, the error rate.

<Figure1>

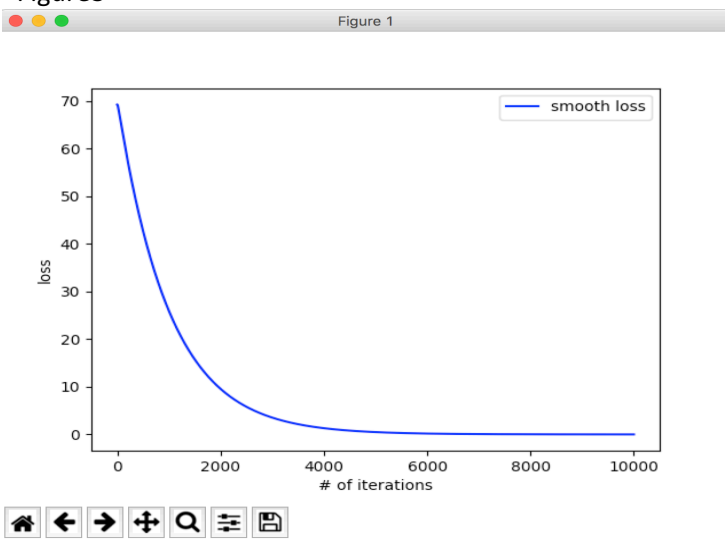
input_mini_result.txt

```
data has 28 characters, 13 unique.
Seed, the first character: H
Prediction: ----
 rrawwtlded wcorrcutwaedauwe
----
iteration 0, smooth_loss = 69.253632
Seed, the first character: H
Prediction: ----
 ello world recurrent neural
----
iteration 1000, smooth_loss = 25.938037
Seed, the first character: H
Prediction: ----
 ello world recurrent neural
----
iteration 2000, smooth_loss = 9.566160
Seed, the first character: H
Prediction: ----
 ello world recurrent neural
----
iteration 3000, smooth_loss = 3.533820
Seed, the first character: H
Prediction: ----
 ello world recurrent neural
```

<Figure2>

```
iteration 6000, smooth_loss = 0.187427
Seed, the first character: H
Prediction: ----
 ello world recurrent neural
----
iteration 7000, smooth_loss = 0.074788
Seed, the first character: H
Prediction: ----
 ello world recurrent neural
----
iteration 8000, smooth_loss = 0.032542
Seed, the first character: H
Prediction: ----
 ello world recurrent neural
----
iteration 9000, smooth_loss = 0.016382
```

<Figure3>



2. Mini-Input text and Analysis

The input file src/input_alphago.txt has mini-sized original text that is "AlphaGo is a narrow AI computer program that plays the board game Go.[1] It was developed by Alphabet Inc.'s Google DeepMind in London in October 2015…..", containing 1,346 characters and 61 unique characters.

[1] I assume that you have anaconda virtual environment.

(Refer to the [2]original text). The RNN model is built with the sequence length 25, the hidden size of 100, epoch of 100,000 and prediction length of 200.
As an analysis (Refer to the result/ input_alphago_result.txt in detail), in the iteration of 0 with the seed character "A" in the <Figure4>, the 200 length of prediction as an output starts with "WK".

<Figure4>

```
data has 1346 characters, 61 unique.
Seed, the first character: A
Prediction: ----

WK37GKyhAibpq(yp1hr75d6fNwA5xB5yN×SO6D26k6wM(e(M]zKuWDoOh0]9lMwgn'FmnN]W)arxdC,C.lv4l4Wrc
W9310iM)f.ghh] ksdYwdnNSp'c-qfpwiK5i.Seie10]6Y a0SualYx96K),-KJtJq".,Lqqgkpqerh0a.lBgOWd.
2Yv×ttyo[5OLzYM94e0qdG2Ft0×Jui-"p99vKxFL'5LJYIsJxvN)i.kG B×rfBL"Fv9hIO-n[-
M6oFSvGuowvLbI,Isda9-[zoBq]ySaM-x76MIA(qaa9m k[)7JhpdzrAN[O(3hw5u7)425q-
q)C1rW]I25t4L[sxa"h3Scf'NmkwM2S)lhwym0"d6lnuJed[s4wKw'z5a.M1IwwYo6AD.)ye-
C,gI1G.No"nSwmwLic'fDJ.mzp79ilYhC,D×d04kzv('b"w]1MGFwt]N-(iogegg'(1.sOLNJ.9lbxD-[yF
OYk5ln450rvbo2gsW)OK[rs7f0dnMsNxmot[d3m-BCuLKI×MA-h9nkoFxD[wDN JdJ.
3CnJ'5GJ.Y9C×u'B.WJG9g)b461sMA'e2k1AL(9hzoGofn0)3r3YKh-3y- CWKf9g[rJ-f"6qiw(J uDK"WlscC-
al-(YFp2n5xDpt)zNWd-qCC97Soz,]'03-e7b×]Y]2YY1(I"55O]1-
bwCiA'F5s("L0d'igq"3nk(r-5mAazFJdN29wad2b"Kzl15fWfaL0vW9J6z-'zNB-.Iv0kAqdue)N
J6WwA2k2(WNL,kmp-iWKdIci×mA)hpMtSd[S×7×63gKz7qCS'n9y,s")ocCWFkcinNa'pYztOJAMvLh)Y9SnO-
vrMMcu6wpWKb)v9a5717r 2)KpY.xN"rOybN)C1MWFn'W4OKormIew11GMytSN07psO×p-,Sp-]o0p[2qx,
2yG(fduSAiFCs 0S[K4Ooe,Bu-×](AdL62i7 fgm"Ks×KDv]i6cbs9e5IvcxF
----
iteration 0, smooth_loss = 102.771851
Seed, the first character: l
Prediction: ----
 ihSGtiDw G 9GeJAaAhabe Wodh daA.ecoCt elai tw A thht JgqAai Jhdr.wraelWl'e-
pWerhachJpp[Gl wlAhpeealafwDlshA[e whh Jhe Gi reaetiAvtAalheesf Jhawehcnew rsdd
vr0im.eowl-,DeeAl-alaw 0.aAye, [ee9ohe cise wen6a Ge Aooh.tnehgeelagt
pnoteelhhemenhheApwcpndwidiooohG[iuwAondmoSK bh 6e JaeiseoCe A Gr iesoaGn.] AooaaGewK bt
bltGee, boeaso[G9 saGenKdeN A,i, Ks tedecMele on Arh×clau1oh A m lB6btGhaGe fl-
epAliioAcheo AoahaGaNhAaneKha6oiAhahohWffmBGhiJcesrpuw Gi-aG,-Geoatahthth Fa preo,
Aa[oG]eKe6awoswieaigib0dcfhdioolenpKeJ6eihAhe 0 oc l Gptyro d, GeqN
CaGslJAaGerhtarnWi1hfans olaeleaelrepKhchhwe wcivcpotomw dyAi B Gihnois,
ffehoWn,lileDaphinne, wh,a whoKsr lhesG1h9wao fartn Kolho]i i, ' bacfteSicaGCoweAdsd
GoeoipnhwsGooAneKA,eAhtid b z, k whaleKoe bhe[naWyaAeibqmph , eaeA0Asbhoii. t Jo b fhawaw
GA JoaAcpq, A wlallaeh,aIwaA bo,inaGktiMlAnatotwepbqd,aGhtl trciGhatlmw 6she wh
Gpenwnei7oeaBiantw6,e6ls.ovnlGenet faGn b Glerp dl , 9t JtlqeteothseAKeao Ao,Kn iaohwn G
kssW Gotha weiwtAhckholiaGl Ahoow1,cG A
----
iteration 200, smooth_loss = 101.854908
Seed, the first character: h
```

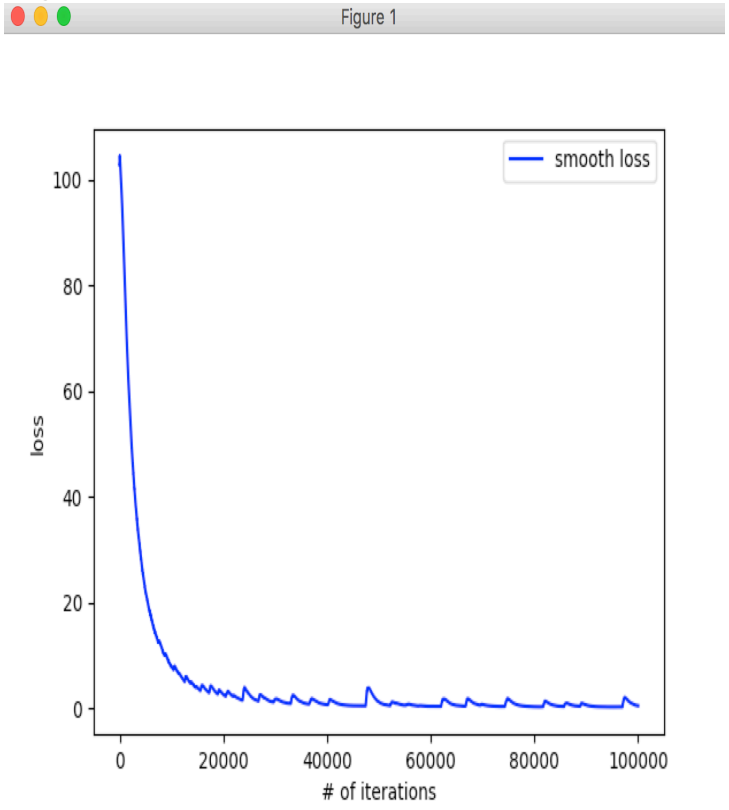
In the iteration of 99,800 in the <Figure5> with the seed character "m", the RNN models outputs 200 characters that is more readable after leaning the original input 1,346 characters.

<Figure5>

```
Prediction: ----
 -game match, the first time a computer Go program to beat a final game, Lee Go.[s]
AlpeaGurcr Go beate if be mat husly "learned" by machine learning, specifically by an
artificial network (a deep learning method) by extensive training, both from human
professional without handicaps.[4] Althon in che fin f AI comeuter program ha a se worl,
asearat a human profesroug ourar the first time a AImees cofr ats In ry professized 1fca
tho fiving a fine wosline Seamehrhander at omeumanktion the world No.1 ranked player at
the time, in a three-game match. After this, AlphaGo was awarded professional 9-dan by
Chinese Weigi Association.[6] After the match between AlphaGo and Ke Jie, AlphaGo will re
whrlf the Sumbhr at tsed. Aa As areas.[7] AlphaGo uses a Monte Carlo tree search
algorithm to find its moves based on know9, ro puter profes. It was developed by Alphabet
Inc.'s Google DeepMind in London in October 2015. It became the first Computer Go program
to beat a human professional Go player witho
----
iteration 99600, smooth_loss = 0.597504
Seed, the first character: m
Prediction: ----
 puter player at tha cam harcIncod lo Lee Sedol in thmeur Inc.'s Google DeepMind in
London in October 2015. It became the first Computer Go program to beat a human
prourcerila phificam harea goay time oleine oftrog hfourar Go grtime a human profesrind
Ke Jie, the five-game mina n thme, inner the Go. Af on h of the fiep toat was d vosenion
ay the victory, AlphaGo was awahd. garch at anar ay the wo Lee Sedof AlphaGo.[1] if by an
a timas hasuthe worlo phav ationear at thbon ghmevilg ale ofren the warlo ghbve "feanat
malgion London in October 2015. It became the first Computer Go program to beat a human
professional Go player without handicaps on a full-sized 19×19 board.[2][3] In March
2016, iteeesional 9-dan by Chinese Weigi Association.[6] After the match between AlphaGo
us hnil2-gion an artinner the world No.1 ranked player at the time, in a three-game
match. After this, AlphaGo was awarded professional 9-dan by Chinese Weigi Association.
[6] After the match betweel neurct beatgod) by ma
----
iteration 99800, smooth_loss = 0.550363
```

The <Figure6> shows the loss, error rate that decreases as iteration increases.

<Figure6>



[2] AlphaGo text comes from https://en.wikipedia.org/wiki/AlphaGo

8. Reference

[1] Andrej Karpathy blog(2015), Character Level Language Models, https://http://karpathy.github.io/2015/05/21/rnn-effectiveness/, https://gist.github.com/karpathy/d4dee566867f8291f086