Hacettepe University
Department
Of
Computer Science

BBM-497 Introduction to Natural Language Processing

Poem Generation

Simge ACIMIS / 21327598

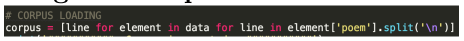
June 2nd - 2019

Contents

Chapter 1

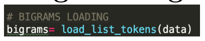
Training Language Model with Dynet

Corpus [0:10]
that crazed girl improvising her music', 'her poetry dancing upon the shore', 'her soul in division from itself', 'climbing falling she knew not where', 'hiding amid the cargo of a steamship', 'her knee-cap broken that girl i declare', 'a beautiful lofty thing or a thing', 'heroically lost heroically found', 'no matter what disaster occurred', 'she stood in desperate music wound
Length of corpus: 531136



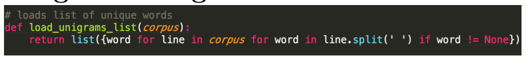
```
# CORPUS LOADING
corpus = [line for element in data for line in element['poem'].split('\n')]
```

Bigram [0:100]
¡s¿ that', 'that crazed', 'crazed girl', 'girl improvising', 'improvising her', 'her music', 'music ', 'her poetry', 'poetry dancing', 'dancing upon', 'upon the', 'the shore', 'shore ', 'her soul', 'soul in', 'in division', 'division from', 'from itself', 'itself ', 'climbing falling', 'falling she', 'she knew', 'knew not', 'not where', 'where ', 'hiding amid', 'amid the', 'the cargo', 'cargo of', 'of a', 'a steamship', 'steamship ', 'her knee-cap', 'knee-cap broken', 'broken that', 'that girl', 'girl i', 'i declare', 'declare ', 'a beautiful', 'beautiful lofty', 'lofty thing', 'thing or', 'or a', 'a thing', 'thing ', 'heroically lost', 'lost heroically', 'heroically found', 'found ¡/s¿', '¡s¿ no', 'no matter', 'matter what', 'what disaster', 'disaster occurred', 'occurred ', 'she stood', 'stood in', 'in desperate', 'desperate music', 'music wound', 'wound ', 'wound wound', 'wound and', 'and she', 'she made', 'made in', 'in her', 'her triumph', 'triumph ', 'where the', 'the bales', 'bales and', 'and the', 'the baskets', 'baskets lay', 'lay ', 'no common', 'common intelligible', 'intelligible sound', 'sound ', 'but sang', "sang 'o", "'o sea-starved", 'sea-starved hungry', 'hungry sea', 'sea ', '¡s¿ he', 'he waits', 'waits patiently', 'patiently at', 'at the', 'the gates', 'gates of', 'of heaven', 'heaven ', 'pleased with', 'with his', 'his sunset-hued', 'sunset-hued tie-dyed'
Length of bigrams: 3404042

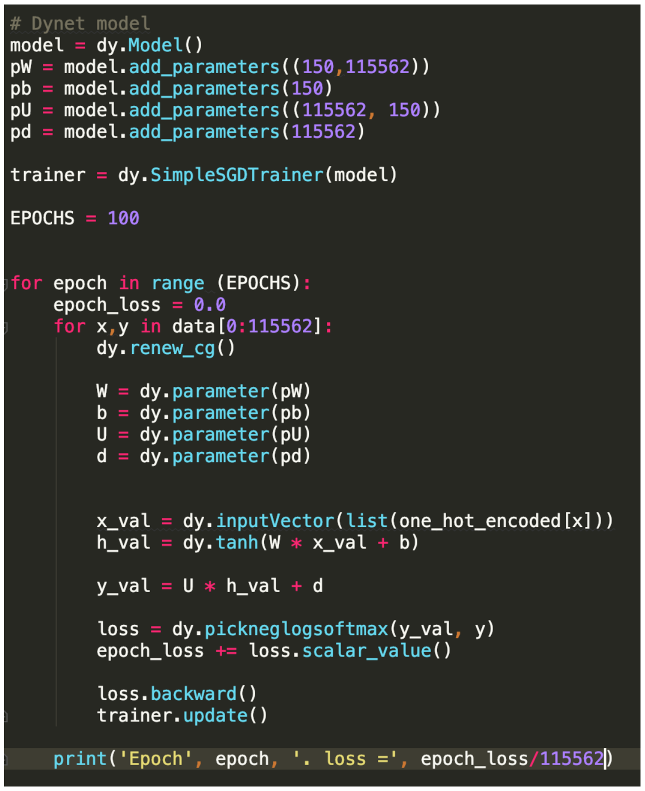


```
# BIGRAMS LOADING
bigrams= load_list_tokens(data)
```

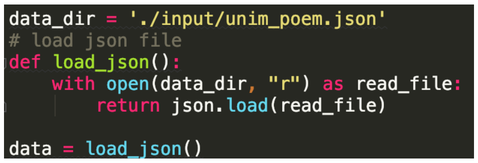
Unigram [0:10]
', 'mysteries', 'unembodied', 'plaisirs:', 'er', 'bravely', 'crocks', 'pago', 'biggir', 'estate:'
Length of Unigrams: 115563



```
# loads list of unique words
def load_unigrams_list(corpus):
    return list({word for line in corpus for word in line.split(' ') if word != None})
```

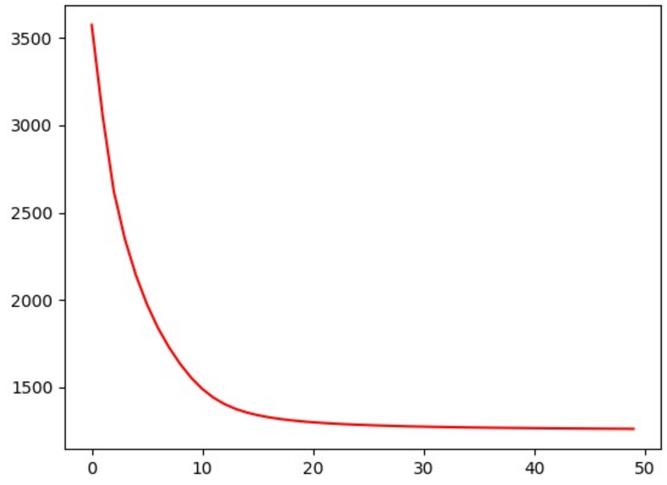


```python
# Dynet model
model = dy.Model()
pW = model.add_parameters((150,115562))
pb = model.add_parameters(150)
pU = model.add_parameters((115562, 150))
pd = model.add_parameters(115562)

trainer = dy.SimpleSGDTrainer(model)

EPOCHS = 100


for epoch in range (EPOCHS):
    epoch_loss = 0.0
    for x,y in data[0:115562]:
        dy.renew_cg()

        W = dy.parameter(pW)
        b = dy.parameter(pb)
        U = dy.parameter(pU)
        d = dy.parameter(pd)


        x_val = dy.inputVector(list(one_hot_encoded[x]))
        h_val = dy.tanh(W * x_val + b)

        y_val = U * h_val + d

        loss = dy.pickneglogsoftmax(y_val, y)
        epoch_loss += loss.scalar_value()

        loss.backward()
        trainer.update()

    print('Epoch', epoch, '. loss =', epoch_loss/115562)
```



```python
data_dir = './input/unim_poem.json'
# load json file
def load_json():
    with open(data_dir, "r") as read_file:
        return json.load(read_file)

data = load_json()
```

Model is trained with 115562 input layer, 150 hidden layer, 100 epoch by using back propagation.



Epoch vs Loss Graph

Chapter 2

Poem Generation, Examples, Perplexities

In addition to train, I added **softmax** to normalize y_val values to range of 0 to 1 **probability values**. And I didnt́ use **argmax** because while using argmax, program generated poems using only one word. To solve this problem, I used **weighted choice** like we used in Assignment 1.

2.1 Poem Examples

2.1.1 Poem-1

START clutched brotherhoods pectus
falling lavished football all-attentive
river-drift falling saw reaches
rarity quagmire washed-up jersey/cut
landnow benedicite a-bombs kink END

Perplexity:29456.910

2.1.2 Poem - 2

START free free free aucun
free whosoever poach free cheekor
garret mail free snails free
carnifex ellick upton free intertwined
bruggia whosoever free vura windblown END

Perplexity:23798.585

2.1.3 Poem - 3

START adjusted decomposition mishe-mokwa justo officers streetsin
weather-beaten grey-nick baldacca's eastings bequeaths mishe-mokwa glotonye
weather-beaten belongeth superfluity blackbirds wh'n correva inspiratrice
assembles adjusted signify avenge: dimando justo spend
exalt christmas-morning cutters cigars inspiratrice decomposition target-eyes END

Perplexity:117843.342

2.1.4 Poem - 4

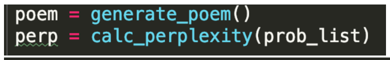
START could half-relenting when to
semi-apes nationale what far-seeing untasted
kyoto rottenness admixtured to ge
fanfare nag horizontals buscarte toglie
parkland fallingoff
journal' gunsight END

Perplexity: 43716.14

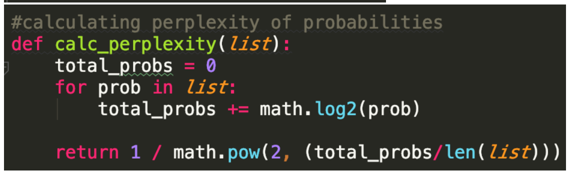
2.1.5 Poem - 5

START stickily sleights databases
bridge horse-hair joppa exhort
thomas the ameliorate vid? lagrimando
portico rosary swelling in riguardi
cerchiato greetings tonsures nicotianna END

Perplexity:39023.433



```
poem = generate_poem()
perp = calc_perplexity(prob_list)
```



```
#calculating perplexity of probabilities
def calc_perplexity(list):
    total_probs = 0
    for prob in list:
        total_probs += math.log2(prob)

    return 1 / math.pow(2, (total_probs/len(list)))
```