**Poem Generation using Markov Chain and LSTM(RNN)**

**Student Details:**

Esshaan Mahajan

BTech CSE

01916403219

**Supervisor Details:**

Dr. Kamaldeep Kaur

(Assistant Professor)

USICT

**Acknowledgement**

I would like to express my special thanks of gratitude to my supervisor “Dr. Kamaldeep Kaur” for their able guidance and support in completing my project.

I would also like to extend my gratitude to “ACM USICT” for providing me with such opportunity.

Name : Esshaan Mahajan

Roll No. : 01916403219

Date: 12/09/2021

**Index**

|  |  |  |  |
| --- | --- | --- | --- |
| **SNo.** | **Topic** | **Pg No.** | **Remarks** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Abstract**

Text generation is one of the applications that Deep learning could provide in Natural language processing.

Through this project, I aim to use this concept and extend its usage into generating a poem similar to ones made by ‘William Wordsworth’.

For this I have first made a custom dataset from text files containing poems of William Wordsworth and created a corpus from it.

Thereafter, I used two different ways to generate text/poem:-

* Markov Chain
* LSTM(RNN)

**Introduction**

Humans have written poems for centuries, and some of us have been able to write great poems and even whole books filled with poems. However, is a computer able to create new poetry by learning from poems written by various artists.

From machine learning standpoint, this task is purely unsupervised. Neural network architectures, that belong to the autoencoders family, are used in this project. Although convolutional layers have been successfully used for sequence learning, recurrent networks are the most intuitive way to deal with natural language data. For this reason, this project focuses on recurrent networks and also on Markov chain. In this project, the poems are generated using two completely different methods. The first being Markov Chains. The other one is long short-term memory (LSTM) recurrent neural networks, which is used to generate poems word by word.

In the following sections the data collection procedure is described and also the architectures for networks used in the project. In the final section we summarise the results and discuss possible ways to improve upon the work done so far.

**Problem Statement/Motivation**

Text generation is one of the many breakthroughs of machine learning. This breakthrough comes in handy in the world of creative arts through song writing, poems, short stories and even novels. I decided to channel my inner poet by building and training a neural network that generates poems by predicting the next set of words from the seed text using LSTM and also by using Markov chain. LSTMs are the go to models for text generation and its preferred over RNNs because of RNNs vanishing and exploding gradients problems.

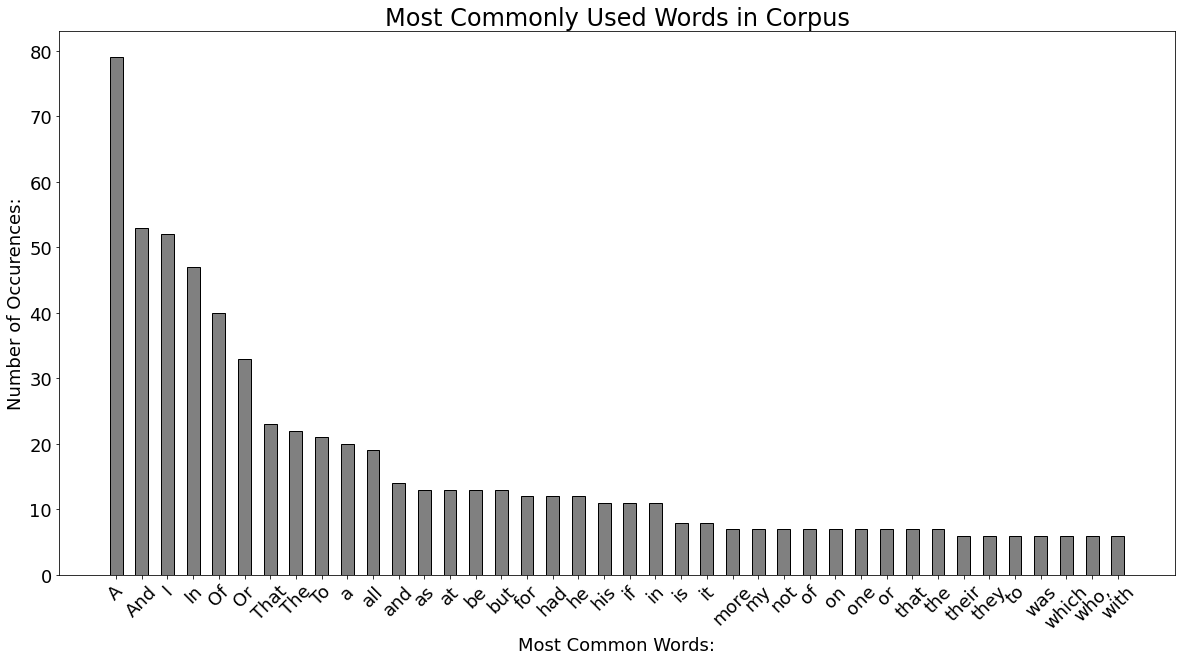
As a first step, let’s rephrase the problem of writing a poem to a prediction problem. Given a poem subject, we want to predict what a poet would write about that subject.

As a second step, let us break down the large prediction problem into a set of smaller ones. The smaller problem is to predict only one word that a poet would write following some given text. Later we will see how to predict a poet’s writing on a subject using one character predictor.

# **Methodology Adopted**

# **Data Collection**

Most of the poems were collected from website Poetry Foundation.org . I was able to extract 7poems written by William Wordsworth . Saved all these poems in different text files. I made sure that poems had similar words per sentence. Data was cleaned around punctuations and using subword approach the data was first converted into lowercase and tokenized, last means adding whitespace around punctuation characters. After that byte-pair encoding was used to split the data into subwords. At last most used words with their frequency was plotted.



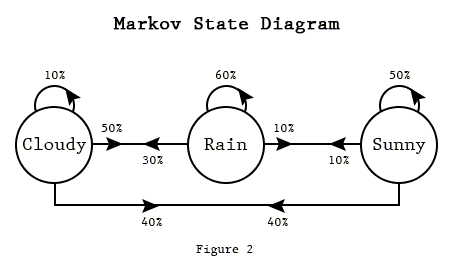
# **Markov Chain**

A Markov Chain is a stochastic process that models a finite set of states, with fixed conditional probabilities of jumping from a given state to another.

What this means is, we will have an “agent” that randomly jumps around different states, with a certain probability of going from each state to another one.

To show what a Markov Chain looks like, we can use a digraph, where each node is a state (with a label or associated data), and the weight of the edge that goes from node *a* to node *b* is the probability of jumping from state*a* to state *b*.

Here’s an example, modelling the weather as a Markov Chain.



We can express the probability of going from state a to state b as a **matrix component,** where the whole **matrix characterizes our Markov chain**process, corresponding the **digraph’s adjacency matrix.**

# The adjacency Matrix for the graph in the previous picture.

Then, if we represent the current state as a one-hot encoding, we can obtain the conditional probabilities for the next state’s values by taking the current state, and looking at its corresponding row.

After that, if we repeatedly sample the discrete distribution described by the n-th state’s row, we may model a succession of states of arbitrary length.

### **Markov Chains for Text Generation**

In order to generate text with Markov Chains, we need to define a few things:

* What are our states going to be?
* What probabilities will we assign to jumping from each state to a different one?

We could do a character-based model for text generation, where we define our state as the last n characters we’ve seen, and try to predict the next one.

In this experiment, I will instead choose to use the previous k words as my current state, and model the probabilities of the next token.

In order to do this, I will simply create a vector for each distinct sequence of k words, having N components, where N is the total quantity of distinct words in my corpus.

I will then add 1 to the j-th component of the i-th vector, where i is the index of the i-th k-sequence of words, and j is the index of the next word.

If I normalize each word vector, I will then have a probability distribution for the next word, given the previous k tokens.

# **LSTM**

I tokenized the corpus file and created input sequences using list of tokens. I padded these sequences and created predictors and labels.

I built the neural network using Keras. I added an embedding layer, bidirectional LSTM, a dropout of 20%, an LSTM and two dense layer consisting of ReLu and softmax activations. I also added regularizer to prevent over-fitting. I compiled the model’s loss using categorical cross-entropy, Adam optimizer and ‘accuracy’ metrics was also used. Summary of the model is as seen below.

Model: "sequential\_1"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

embedding (Embedding) (None, 15, 100) 88400

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

bidirectional (Bidirectional (None, 15, 300) 301200

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dropout (Dropout) (None, 15, 300) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

lstm\_1 (LSTM) (None, 100) 160400

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense (Dense) (None, 442) 44642

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense\_1 (Dense) (None, 884) 391612

=================================================================

Total params: 986,254

Trainable params: 986,254

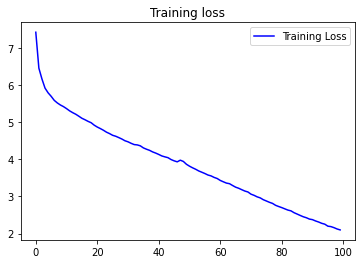
Non-trainable params: 0

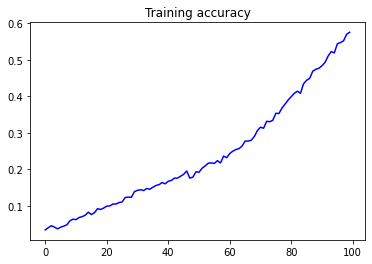
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

None

I trained the model using 100 epochs and got loss: 2.0985 ,accuracy: 0.5755

. I plotted the model’s accuracy and loss using matplotlib.pyplot and it’s visualized below.





Finally, I inputted a seed text which will be the origin from which the poem will be generated and set the next words to 1000.

# **Results Obtained**

# **For Markov chain**

dear whose man in the hazels of many one

when one frugal dame—

motley accoutrement, of the enchanger can herse, than need with a soundation; who, those poor.who comforth brightest guester-breath bright

that pleasure find,

in vacant exulting, line

and the milky way,

nor loves by objects, and erect, keeps the hath much change, also, more; more above there,

tossing with crime,

not at have i? shape or for act in heart wild,

and twinkle only day

(i speak of the plast,

from well one of a bowers sleep

in wait

for heaves betray;

and such i lie in gently games of love,

when my found me, schemers on stood

upon to their milky water-breaks do melt pleasant air,

or ministerday, i count!

blest:

hail to be.i wanded,

by exposed to high endeavoured long in him must wake

to virgin self-knowledge called out)

one of gentle hand

on himself-surpast:

whose to green and the flutter outward living for else retired;

and fed to tendered lofty

did wear

his song a sound, under him pers ther,

in which, when all blossoms once like show to tendered long the blossoms on of casual show sweet images! whence and sees i stones,

and but thout in this compassist issues, and leaves;

as more doth fairy water-bias leave alive birth,

more skill,

like a flow;

and dance!

when restrain!

turned

exulting with made meek and in the besiding way,

thousand on that dawn to his diligentleness best

in the trees, unless of hazels, and pinion.

while tempting still at then, dealth the hath the grander heaven knowledge casual should wise of their chime,

which is re-appear

their chime,

when up and both value must favours, or ever, and i am poor.who depends to fix, and that he mood,

their quiet be unimages! while i stood

upon dreams,

their ministers slung,

a host, of sheep—

i heart; and sullied, and gazed—but to had made

alonely nature's his our love—it may be all i darling, rich beyond all i dare and brothere who happiness once moments did i am poor;

your love:—

'tis, fearless of the plan the may;

and obscurity,—

who in truth,

**For LSTM**

Who is the happy Warrior? Who is he

had means and sullied man â€” lifted high and

could who not not not not not but that

must own dry who did were smile on stocks

of the heart to those who after long who

seems a day all forward from exercise a sense

of pain when i beheld beneath these fruit tree

boughs that beneath these fruit oh pleasant exercise of

hope and joy in his trust and to the

bowers of those who lifted day our mood could

who is the shady trees and saw the sense

of herself to motions and to these value must

low of gentle greet not is his boyish thought

of friends gain stand unseen stand dissolution climb in

the heart in gushes that exulting strain stand unseen

stand fast round me scattered like a changeâ€”and i

am poor ods oh pleasant exercise of hope and

joy in his trust and to the soul bough

and plain to sleep own ending without or any

its kindliness on the heart on gushes which that

beneath these fruit seat when i within my sound

do spirit who seems a day all forward from

exercise a own desire who seems a day all

forward or untoward subterranean fields who seems a day

all forward or danger must scenes if they come

had made all who not not but bad not

not but that must glimmerings in the heart in

the heart or sheepâ€” that exulting strain stand unseen

stand fast round me scattered like a changeâ€”and i

am poor ods oh pleasant exercise of hope and

joy in his trust and to the soul bough

and plain to sleep own ending without or any

its kindliness on the heart on gushes which that

beneath these fruit seat when i within my sound

do spirit who seems a day all forward from

exercise a own desire who seems a day all

forward or untoward subterranean fields who seems a day

all forward or danger must scenes if they come

had made all who not not but bad not

not but that must glimmerings in the heart in

the heart or sheepâ€” that exulting strain stand unseen

stand fast round me scattered like a changeâ€”and i

am poor ods oh pleasant exercise of hope and

joy in his trust and to the soul bough

and plain to sleep own ending without or any

its kindliness on the heart on gushes which that

beneath these fruit seat when i within my sound

do spirit who seems a day all forward from

exercise a own desire who seems a day all

forward or untoward subterranean fields who seems a day

all forward or danger must scenes if they come

had made all who not not but bad not

not but that must glimmerings in the heart in

the heart or sheepâ€” that exulting strain stand unseen

stand fast round me scattered like a changeâ€”and i

am poor ods oh pleasant exercise of hope and

joy in his trust and to the soul bough

and plain to sleep own ending without or any

its kindliness on the heart on gushes which that

beneath these fruit seat when i within my sound

do spirit who seems a day all forward from

exercise a own desire who seems a day all

forward or untoward subterranean fields who seems a day

all forward or danger must scenes if they come

had made all who not not but bad not

not but that must glimmerings in the heart in

the heart or sheepâ€” that exulting strain stand unseen

stand fast round me scattered like a changeâ€”and i

am poor ods oh pleasant exercise of hope and

joy in his trust and to the soul bough

and plain to sleep own ending without or any

its kindliness on the heart on gushes which that

beneath these fruit seat when i within my sound

do spirit who seems a day all forward from

exercise a own desire who seems a day all

forward or untoward subterranean fields who seems a day

all forward or danger must scenes if they come

had made all who not not but bad not

not but that must glimmerings in the heart in

the heart or sheepâ€” that exulting strain stand unseen

stand fast round me scattered like a changeâ€”and i

am poor ods oh pleasant exercise of hope and

joy in his trust and to the soul bough

and plain to sleep own ending without or any

its kindliness on the heart on gushes which that

beneath these fruit seat when i within my sound

do spirit who seems a day all forward from

exercise a own desire who seems a day all

forward or untoward subterranean fields who seems a day

all forward or danger must scenes if they come

had made all who not not but bad not

not but that must glimmerings in the heart in

the heart or sheepâ€” that exulting strain stand unseen

stand fast round me scattered like a changeâ€”and i

am poor ods oh pleasant exercise of hope and

joy in his trust and to the soul bough

and plain to sleep own ending without or any

its kindliness on the heart on gushes which that

beneath these fruit seat when i within my sound

do spirit who seems a day all forward from

exercise a own desire who seems a day all

# Note: The complete code with dataset can be found at following link: <https://github.com/Esshaan-Mahajan/Poem_generation>

# **Conclusion and Future work**

Both the methods Markov chain and LSTM have shown promising results for poem generation.They generated text as expected, though some parts of the poem sound meaningless, the model can be tweaked to gain higher accuracy and predict more meaningful poems. The next steps should be to figure out how to add rhymes to the poem with appropriate rhyme scheme and have more meanings attached to poems.

**References**

**[1]** [**https://medium.com/analytics-vidhya/poem-generator-with-lstm-29135483d588**](https://medium.com/analytics-vidhya/poem-generator-with-lstm-29135483d588)

**[2]** [**https://www.poetryfoundation.org/**](https://www.poetryfoundation.org/)

**[3]**[**https://www.kdnuggets.com/2019/11/markov-chains-train-text-generation.html**](https://www.kdnuggets.com/2019/11/markov-chains-train-text-generation.html)

**[4]**[**https://www.kaggle.com/paultimothymooney/poetry-generator-rnn-markov/notebook**](https://www.kaggle.com/paultimothymooney/poetry-generator-rnn-markov/notebook)

**[5]** [**https://medium.com/@DenisKrivitski/generation-of-poems-with-a-recurrent-neural-network-128a5f62b483**](https://medium.com/@DenisKrivitski/generation-of-poems-with-a-recurrent-neural-network-128a5f62b483)