

**School of InfoComm Technology**

**Deep Learning Assignment**

Diploma in CSF / FI / IT

Apr 2022 Semester

**ASSIGNMENT 2**

(40% of DL Module)

4th Jul 2022 – 12th Aug 2022

**Submission Deadline:**

**Presentation: 12th Aug 2022 (Week 17),**

**Report: 12th Aug 2022 (Friday), 11:59PM**

|  |  |  |
| --- | --- | --- |
| **Tutorial Group** | **:** | **P01** |
| **Student Name** | **:** | Vincent Tan Guan Shen |
| **Student Number** | **:** | S10227963E |

**Penalty for late submission:**

10% of the marks will be deducted every calendar day after the deadline.

**NO** submission will be accepted after 21st Aug 2022 (Sunday), 11:59PM.

Contents

[1. Overview 3](#_Toc111219363)

[2. Data Loading and Processing 3](#_Toc111219364)

[3. Develop the Sequence Generator Model 4](#_Toc111219365)

[4. Use the developed model to generate texts 6](#_Toc111219366)

[5. Summary 7](#_Toc111219367)

# 1. Overview

There are many books in the world and there are also ways for normal users to create their own story. There are models that allows words to be generated word by word, but not many models allow generation of text from a character-by-character basis. Hence, the objective of problem 2 of the assignment requires the building of an English language character generator that takes in user input and generates sentences character-by-character. The book that would be used to train the model would be the book Harry Potter and the Philosophers Stone by J.K. Rowling which contains 348 pages. With deep learning, we are able to train a model that can generate text by using predictions that was made from learning the book provided. It would then use the predictions made to understand the patterns between the text and the provided input. The approach is to first read the book and split it by lines. This would allow the book to be processed by the model line by line. Next, for the data processing part, empty spaces, obsolete words and special characters would be filtered out and removed from the text data array as these characters would not be useful when making predictions. When developing the character generator model, a test model is bring tried out with a defined max length for each extracted character sequence and a stop to start a new sequence. This is to show how would the generator work when the model is done. There is also a set of unique characters that are extracted from the array of sentences which would be used for the prediction in the model later. For further processing, each sentence would be converted into one hot encoding so as to allow the model to have a value that can be used for prediction for the words that may be used for the next character. For the model, the sequential model would be used and LSTM network and the activation function of softmax. The optimizer used is RMSprop with the learning rate of 0.01 and the loss of categorical\_crossentropy to compile the models. The percentage of loss will then be considered to see which graph would provide a more accurate result. When generating the characters, user input is provided with a max length of 60 characters as a numpy array and would be converted to a string variable. The input would then be placed in the model to predict the possible characters that can be placed to continue the word or sentence. As for the different temperatures, it is used to allow the model to predict the characters with different amount that would be calculated with a logarithm function and chosen based on a randomized basis according to the probability distribution that is calculated with the exponential of the log function and divided by the sum of the exponential function. With the calculation for the different temperatures, the model will the generate the characters for each temperature accordingly until the 400th character.

# 2. Data Loading and Processing

As the book was provided, the loading of the data uses python’s open function to read the text as a whole. There were some errors with the encoding at first hence it was explicitly specified on the code as “encoding = utf-8”. The data was then changed to full lowercase and split by the “\n” which indicates a new line to store the data in arrays by each line of the book. For pre-processing of the data, strip was used to remove all extra spaces that were not needed, this would allow each array to not have extra spaces that are obsolete. Next, as there was a “/” in the text, pop was used to remove it. There were also page numbers that were not needed in the data. Hence, a for loop containing remove function was used to remove the pages when a “page |” was detected. There were also many lines that only consists of empty spaces, for that, the filter function was used to remove all empty arrays in the text array. There were also some unique characters like a dot and square that was found in the text array, hence a loop was used to check through the array to see if there were any value that is the same as compared to the special characters and remove if it was found. Finally for the last step, the text array was converted to a string using the join function to allow it to be used for later processes. For the one-hot encoding, the numpy zeros function was used to add zeros into the sentences. The python enumerate function was used to look through each sentence in the sentences array to place a true in the array of zeros which are converted to false booleans in the array. This would allow the model to allocate a value to each sentence which allows each character to have a value by one hot encoding which would then allow the model to use and predict the sentences at the processes later.

# 3. Develop the Sequence Generator Model

For the developing of sequence generator models, there is a max length of characters which has been preset to 60 and a step of 3 to create new sequences to show how the models are created by using the for loop to get the characters. Next, by using dictionary function in python, a set of unique characters were able to be extracted from the given array by looping through each character and check if there are duplicated words. This would allow the model to use the unique words to generate characters. Next, there is one hot encoding where zeros are placed into the sentences array to allow each unique sentence to have a specific value in the model. This allows the model to have references of how sentences should look like and it would be better for the model to predict. For the first model, sequential model has been used with LSTM model of inputs of the max length and the total length of the unique characters. It only has 2 layers with the last layer being a dense layer with the activation function of softmax as it is a multiclass and single label classification. With the optimizer of RMS prop and the learning rate of 0.01, the model is learning at a very wide group at each time. For the fitting of the model, a batch size of 512 was used with 3 epochs. This model has a total training parameter of 101,303. This has resulted in a final loss of 1.54%. For the second model, there are 3 layers that were used. Firstly, it is the LSTM layer with an input of 256 and the same variables for the input shape as the first model. The second and third layers are both dense layers, it has the same optimizer and loss of RMSprop and categorical\_crossentropy respectively. However, for the second model, the number of epochs has been changed from 3 to 5. This model has a total trainable parameter of 336,703. This model has a final loss value of 1.27%, which would be a more preferred model as compared to the model due to its higher accuracy. As for the model performance, although the second model has a higher accuracy, the generated text contains a high number of special characters on every temperature. This model has then been made to a lower priority of choice as compared to the first model. As for the first model, although it has a lower accuracy than the second model, the characters generated appears to be readable english words. For lower temperatures like 0.2 and 0.4, the characters generated are readable words but it does not make any sense to the sentences that were created. Although it might make any sense as sentences, it would be suitable for use with more training and getting references from more variety of books. For temperatures of 0.6 ,0.8 and 1.0, the characters generated starts to be non-readable as words. This result shows that temperatures of lower numbers will create a better prediction of words and the model with a higher accuracy might not always be the best model.

Text

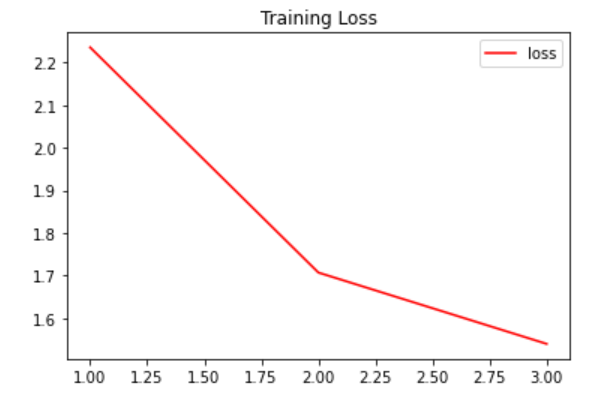
Description automatically generated

Table

Description automatically generated

Text

Description automatically generated



Text

Description automatically generated

Table

Description automatically generated

Graphical user interface, text

Description automatically generated

Chart, line chart

Description automatically generated

# 4. Use the developed model to generate texts

With the developed model, the first model seems to be a better model as the text generated appears to be more readable as compared to the second model. As seen from the figures below, although some words might not make sense from the text generated from the first model, it contains mostly readable words and users would be able to get some ideas from the generated text from the lower temperatures. The words tend to get more randomized when it is given a higher temperature. However, as for the second model, the text generated are not readable as there are too many special characters that would not be able to provide ideas to the users. Some ways that this model can be used on a real life text input would be for beginners who would like to write a story. Although these models are not sufficient to generate perfect stories that are readable, with more trainings and more references that are used, it can be a very useful tool in the near future. Another way that these models can be used would be for book enthusiasts that likes to read books. This model can be a platform for them to read a continuous book with predictions made from different kinds of books they have read from. Its model can be modified to get references from the user and take in all sorts of books as their reference. This would allow many different kinds of books from the world to be trained into this model and allow the model to predict the characters with a more variety of genres. Some other real life text input would be to make automated replies. If the model is tweaked to receiving messages as references to train, it can definitely use the different temperatures to predict a best way to reply to messages either to online shopping buyers or even website chatbots.

Text, letter

Description automatically generated

Figure 1 Text Generated from 1st Model

Text, letter

Description automatically generated

Figure 2 Text Generated from 2nd Model

# 5. Summary

In summary, although the second model has a higher percentage, the first model has a better character generation as compared to the second model. Some places that needs to be improved can be to add more dropouts as it would be able to reduce overfitting by removing different subsets of neurons. It also breaks up non significant patterns in the layers. There can also be more regularizers other then only placing L2. The number of epochs and batch size can also be increased as some models might need to have more than 100 epochs to make sure that there is overfitting or if the graph is underfitting. Some further improvement can also be from the increased number of books that are used for references as this would increase the accuracy for the character generation and allow users to use the model more easily.