Next Word Prediction Using LSTM and GRU

Jayaswathiga S
Division of Mathematics
School of Advanced Sciences
Vellore Institute of Technolog
Chennai, India
jayaswathiga.s2021@vitstudent.a

Vijayalakshmi S
Division of Mathematics
School of Advanced Sciences
Vellore Institute of Technology
Chennai, India
vijayalakshmi.s2021@vitstudent.ac.in

Venkatalakshmi S
Division of Mathematics
School of Advanced Sciences
Vellore Institute of Technology
Chennai, India
venkatalakshmi.2021@vitstudent.ac.in

Keerthana D
Division of Mathematics
School of Advanced Sciences
Vellore Institute of Technology
Chennai, India
keerthana.d2021@vitstudent.ac.in

Dr Kalyani Desikan
Division of Mathematics
School of Advanced Sciences
Vellore Institute of Technology
Chennai, India
kalyanidesikan@vit.ac.in

Abstract
One of the subfields of natural language processing that can assist with predicting the next word are called next-word prediction, which is also sometimes referred to as language modeling. One of the applications of machine learning is this. The process can be summed up by saying that it involves guessing the next word in a sentence. This paper discusses how to use a neural model that is more advanced than a basic RNN and use it to predict the next word. It has many applications that are used by most of us, such as auto-correct, which is mostly used in emails or messages. Long-Short Term Memory (LSTM), Gated Recurrent Unit, is the model that is used to work in this paper. The deep learning model described in this paper was constructed with the assistance of the TensorFlow library, which was programmed in Python. This paper is carried out for three books as data to predict the next word using the LSTM and GRU models. In this paper, an investigation of LSTM and GRU models is carried out, and the results are compared and interpreted for (datasets name) datasets.

Keywords: RNN, LSTM, GRU, Natural Language Processing, Next Word Prediction

I. INTRODUCTION

The processing of natural languages has been the focus of research and is crucial to a number of different applications. We like to message one another very often, and we've noticed that whenever we try to input text, a suggestion for the next word we'll want to type comes above the keyboard. One of the applications that natural language processing may be used for is prediction. Because of the substantial development that we have achieved in this field, we are now able to employ recurrent neural networks for procedures of this kind. We may learn more about the problems that have occurred with foundational RNN.In this study, we will examine how we can utilize a neural network to predict the next word more successfully than a standard RNN. The Long-Short Term Memory Model (LSTM) and Gated Recurrent Unit (GRU) is what is used here. It is possible to design and train the deep learning model by using the TensorFlow library that is included with Python

II. DATASET

Gutenberg

The Gutenberg Dataset is a collection of more than 50,000 free ebooks from Project Gutenberg, one of the oldest online digital libraries. This collection includes 3,036 English books by 142 authors. This selection represents a small portion of the Project Gutenberg corpus. To the greatest extent possible, metadata, license information, and transcribers' notes have been removed from each book manually. The dataset contains numerous texts from classic literature, philosophy, science, and history, making it an invaluable resource for natural language processing and machine learning studies. The dataset has been utilized extensively in NLP research, including language modeling, text generation, and sentiment analysis, among others. It can use it to train and test machine learning models, as well as conduct additional text analyses.

Dataset 01 – Metamorphosis

We have got data as a book on the subject of metamorphosis from the Gutenberg ebooks collection. This book covers a total of 22086 tokens in its entirety.

Dataset 02 – Mother Earth

We were able to receive data as a book about mother earth from the Gutenberg ebooks collection. This data book has a total of 21138 tokens and is centered on the topic of mother earth.

Dataset 03 – Strange

We have obtained data as a book from the Gutenberg ebooks collection that is focused on the subject of The Strange Case of Dr. Jekyll and Mr. Hyde and includes a total of 25571 tokens.

III. LITERATURE REVIEW

[1] In July 2022 Afika Rianti, Suprih Widodo , Atikah Dhani Ayuningtyas, and Fadlan Bima Hermawan did a study about Next Word Prediction using the LSTM technique. They used the Indonesian destination dataset where each destination contains 4-7 words. They used
The Natural Language processing methods and web scraping models to predict the next word. They predicted accuracy as 75% while the loss was 55% with 200 epochs using the LSTM model. They concluded that the accuracy level is good for predicting the Next Word prediction.

[2] Keerthana N, Harikrishnan S, Konsaha Buji M, and Jona J B anticipated and evaluated Next Word Prediction's performance in 2021. They implemented the LSTM model in order to anticipate the next words in sentences. They used the Irish lyrics dataset, which comprises the facts of each line, to train the model. After 100 epochs, the accuracy of the model was 0.80, leading them to infer that the built word prediction model is reasonably accurate on the presented dataset.

[3] Next Words Prediction was predicted in 2021 by Sourabh Ambulgekar, Sanket Malewadikar, Raju Garande3, and Dr. Bharti Joshi. Project Gutenberg-style sentence prediction was achieved with the help of a Recurrent Neural Network (RNN) trained on a dataset compiled from Franz Kafka's collected ebooks.The researchers came to the conclusion that the RNN model becomes less precise as the distance grows between the given context and the term to be predicted. Since LSTM contains memory cells that can recall the previous configuration, it can be utilized to solve the prolonged dependency problem. They built a system with a 3D input vector layer, a 2D output vector layer, and a 128-layer long short-term memory (LSTM) layer, and achieved an accuracy of roughly 56% after 5 iterations.

[4] Deepal S. Thakur, Rajiv N. Tarsarya, Akshay A. Vaskar, and Ashwini Save conducted a paper titled A Survey on Text Prediction Approaches in 2019 in which they employed multiple datasets and different deep learning techniques, such as RCNN, RNN, and CNN, which produce varying results on different datasets with varying accuracy. They concluded that the analysis of multiple articles provides a clear advantage indicating that deep learning may produce superior results compared to other methodologies. In the past, machine learning and natural language processing were utilized for prediction, but deep learning models generated more accurate results.

IV.RNN

Although very powerful, Recurrent Neural Networks are often difficult to work with because to their limited capacity for long-term memory storage. When dealing with extended data sequences, RNN has difficulty effectively communicating information from one stage to the next. As a result, if a person looks at a single paragraph of text with the purpose of drawing conclusions, there is a risk that RNN will leave out essential information right from the beginning. In addition, the backpropagation RNN has issues with vanishing gradients. Gradients are values that are used to update the weights of neural networks. Backpropagation RNN has difficulty with this. The problem of disappearing gradients arises when the gradient declines as it is back propagated through time, and when the gradient is sufficiently small, it does not contribute to the learning process. Because these layers do not learn, an RNN could forget what it has seen in a long data series; hence, LSTM and GRU are used in our study rather than RNN. This is because LSTM and GRU are more effective at retaining memories. Both LSTM and

GRU have emerged as potential solutions to the challenges posed by this kind of short-term memory, and this blog will devote a significant amount of its attention to researching both of these solutions. As a result, recurrent neural networks will stop learning if many of the layers that came before them have the lowest gradient.

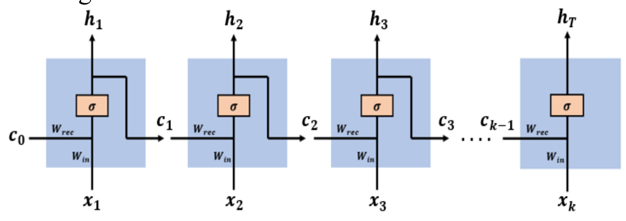


Fig 1.0
Source: OpenSource

Limitations

- This neural network's computation is extremely slow.
- Training may be challenging.
- When using the activation functions, it is difficult to process lengthy sequences.
- It has problems such as Exploding and Gradient Vanishing.

V.LSTM

During the process of backpropagation, neural networks are faced with the obstacle of diminishing gradient descent. It had a substantial effect on the mechanism for updating the weight, and as a result, the model was made useless. As a result, we have used LSTM, which has a memory cell with three gates namely forget gate, an output gate, and an input gate. Additionally, LSTM features a hidden state.
Forget gate – Only the essential cell state information (ht-1 + xt) is sent to the forget gate. Sigmoid function adds these two numbers and returns a value between 0 and 1. It doesn't push values closer to zero since they're insignificant, rather it pushes values closer to one towards the LSTM cell state. In cell state, (ct-1) * (ft). Previous cell state (ct-1) is shown.

1.Input gate
This gate processes (ht-1 + xt) and outputs fresh input using the activation function, which is commonly Sigmoid, in the range of 0 to 1.(ct) is processed using the Tanh activation function, which yields a -1 to 1 result. (ct *) * (ct) is the result of multiplying extra input by candidate input (it). The cell state receives this information.

2. Output gate
This gate processes (ht-1 + xt) and outputs new input using Sigmoid activation function from 0 to 1. (ct) is processed by the Tanh activation function, which generates a -1 to 1 output.

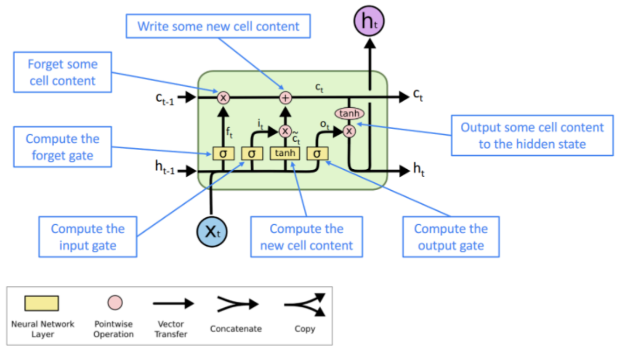


Fig 1.1
https://medium.com/@antonio.lopardo/the-basics-of-language-modeling-1c8832f21079

This figure presents an in-depth study of the LSTM's internal workings. In this case, X stands for the subscript t, which denotes the current time. As can be seen, c and h stand for inputs from a prior period of time or the phase that came before this one. Before moving on to the next gate, there is a forget gate that adjusts the weights so that it is aware of exactly what knowledge must be forgotten in order to go on. Here, we use sigmoid. The input signifies that the cell is now through the process of receiving new information at that time. The succeeding LSTM cell receives the information once it has been sent from the output gate.

Limitations
- LSTMs need more time to train.
- Training LSTMs requires greater memory.
- It is simple to overfit LSTMs.
- Dropout is far more difficult to implement in LSTMs.
- LSTMs are vulnerable to random initializations of weight.

LSTM

Dataset 01 – Metamorphosis

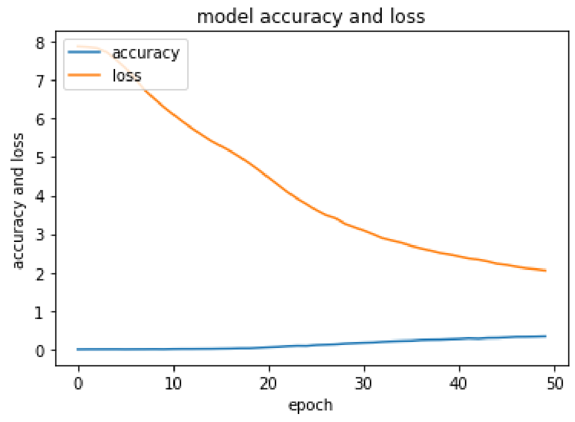


Fig 2.1

This graph clearly shows that the loss value (0.9062), resulting in an accuracy of 0.5724, is steadily decreasing.

Dataset 02 – Mother Earth

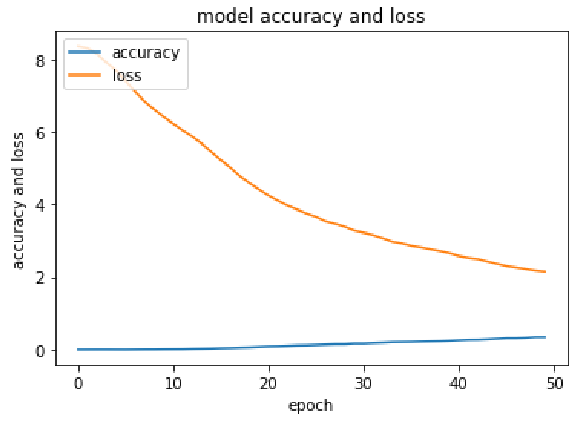


Fig 2.2

This graph shows clearly that the loss value (2.1447) is dropping constantly, with an accuracy of 0.3411.

Dataset 03 – Strange

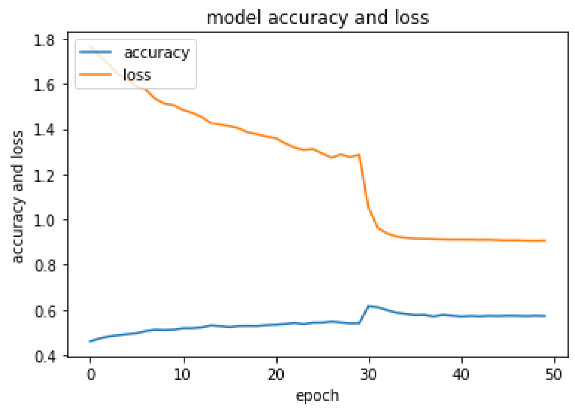
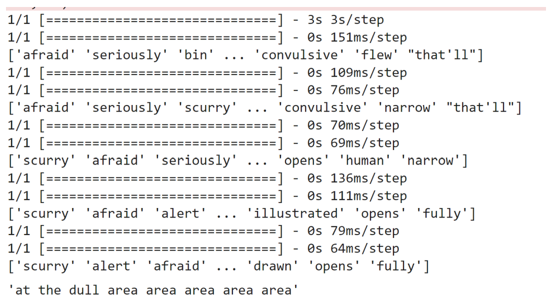


Fig 2.3

This graph clearly illustrates that the loss value (0.9062), resulting in an accuracy of 0.5724, is constantly decreasing.

Next Word Prediction

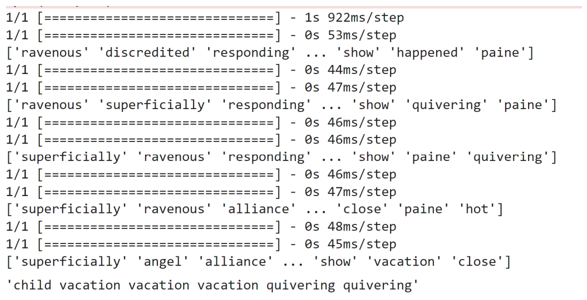
Dataset 01 – Metamorphosis



```
1/1 [==============================] - 3s 3s/step
1/1 [==============================] - 0s 151ms/step
['afraid' 'seriously' 'bin' ... 'convulsive' 'flew' "that'll"]
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 76ms/step
['afraid' 'seriously' 'scurry' ... 'convulsive' 'narrow' "that'll"]
1/1 [==============================] - 0s 70ms/step
1/1 [==============================] - 0s 69ms/step
['scurry' 'afraid' 'seriously' ... 'opens' 'human' 'narrow']
1/1 [==============================] - 0s 136ms/step
1/1 [==============================] - 0s 111ms/step
['scurry' 'afraid' 'alert' ... 'illustrated' 'opens' 'fully']
1/1 [==============================] - 0s 79ms/step
1/1 [==============================] - 0s 64ms/step
['scurry' 'alert' 'afraid' ... 'drawn' 'opens' 'fully']
'at the dull area area area area area'
```

Fig 2.4

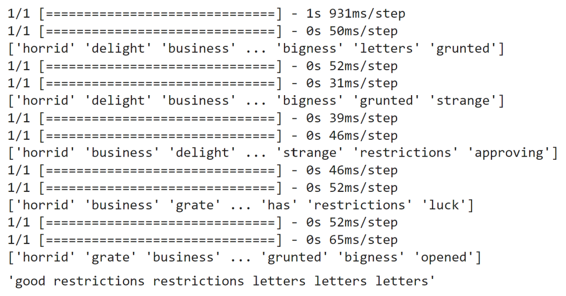
Dataset 02 – Mother Earth



```
1/1 [==============================] - 1s 922ms/step
1/1 [==============================] - 0s 53ms/step
['ravenous' 'discredited' 'responding' ... 'show' 'happened' 'paine']
1/1 [==============================] - 0s 44ms/step
1/1 [==============================] - 0s 47ms/step
['ravenous' 'superficially' 'responding' ... 'show' 'quivering' 'paine']
1/1 [==============================] - 0s 46ms/step
1/1 [==============================] - 0s 46ms/step
['superficially' 'ravenous' 'responding' ... 'show' 'paine' 'quivering']
1/1 [==============================] - 0s 46ms/step
1/1 [==============================] - 0s 47ms/step
['superficially' 'ravenous' 'alliance' ... 'close' 'paine' 'hot']
1/1 [==============================] - 0s 48ms/step
1/1 [==============================] - 0s 45ms/step
['superficially' 'angel' 'alliance' ... 'show' 'vacation' 'close']
'child vacation vacation vacation quivering quivering'
```

Fig 2.5

Dataset 03 – Strange



```
1/1 [==============================] - 1s 931ms/step
1/1 [==============================] - 0s 50ms/step
['horrid' 'delight' 'business' ... 'bigness' 'letters' 'grunted']
1/1 [==============================] - 0s 52ms/step
1/1 [==============================] - 0s 31ms/step
['horrid' 'delight' 'business' ... 'bigness' 'grunted' 'strange']
1/1 [==============================] - 0s 39ms/step
1/1 [==============================] - 0s 46ms/step
['horrid' 'business' 'delight' ... 'strange' 'restrictions' 'approving']
1/1 [==============================] - 0s 46ms/step
1/1 [==============================] - 0s 52ms/step
['horrid' 'business' 'grate' ... 'has' 'restrictions' 'luck']
1/1 [==============================] - 0s 52ms/step
1/1 [==============================] - 0s 65ms/step
['horrid' 'grate' 'business' ... 'grunted' 'bigness' 'opened']
'good restrictions restrictions letters letters letters'
```

Fig 2.6

VI.GRU

LSTM networks are equivalent to GRU networks, which stand for gated recurrent unit networks. A comparison may be made between GRU and an improved version of RNN. Despite this, there are several key differences between the GRU and the LSTM.
GRU doesn't contain a cell state
Information is sent by GRU while it is in its hidden phases. Only two gates are present there (Reset and Update Gate)
GRU is a faster learning algorithm than LSTM.
GRU is more effective than other similar algorithms since it performs fewer tensor operations.

1. Update Gate
The capabilities of Forget Gate and Input Gate are rolled into one with the addition of Update Gate. The forget gate is responsible for selecting which bits of data are discarded and which are stored in memory. This gate determines how much previous data to forward. Multiply input (xt) and weight (wt) by preceding hidden state (ht-1) and weight (wh). This value is also sigmoid-compressed to fit between 0 and 1. The update gate solves vanishing gradient issues.
$\mathbf{u_t = sigmoid\ [\ (x_t)\ *\ (w_t)\ +(h_{t-1})\ *\ (w_h)\ ]}$

2. Reset Gate
This Gate will clear out any gradient explosion by resetting the previously stored information. The quantity of previous data that may be forgotten is controlled by the Reset Gate. Reset gate is similar to LSTM's Forget gate; it sorts unnecessary data and tells the model to forget it. Weights and functionality differentiate the formula and update gate. Its functioning is the opposite of the Update Gate. The Sigmoid Function discards the value closest to 0 and processes the value closest to 1.
$\mathbf{r_t = sigmoid\ [\ (x_t)\ *\ (w_r)\ +(h_{t-1})\ *\ (w_{hr})\ ]}$

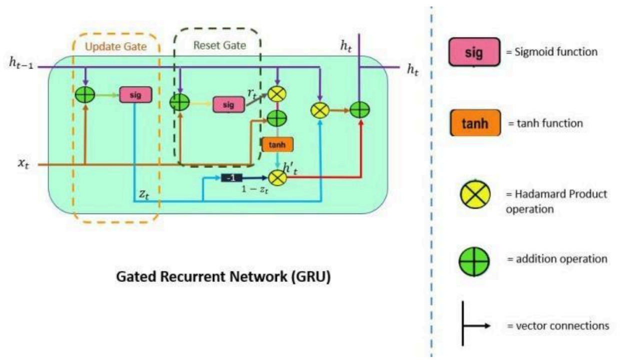


Fig 3.0
Image *Source*

Limitations

- However, GRU models continue to have issues such as a slower convergence rate and poor learning efficiency, leading in an excessively lengthy training period and even under-fitting

GRU

Dataset 01 – Metamorphosis

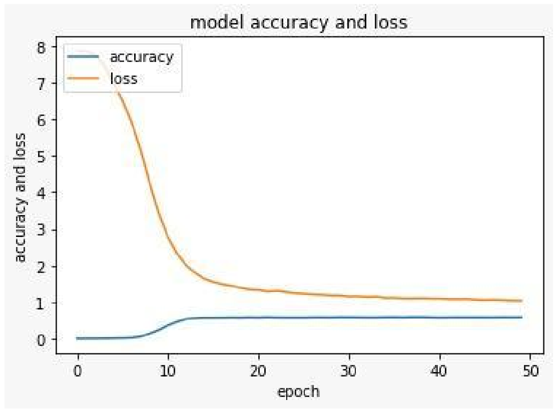


Fig 4.1
This graph makes it very evident that the loss value (0.7617) is steadily going down, which ultimately results in an accuracy of 0.6737.

Dataset 02 – Mother Earth

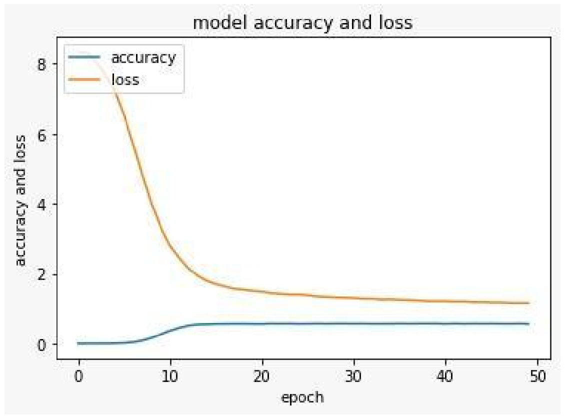


Fig 4.2

This graph makes it very evident that the loss value (1.0062) is steadily going down, which ultimately results in an accuracy of 0.6544.

Dataset 03 – Strange

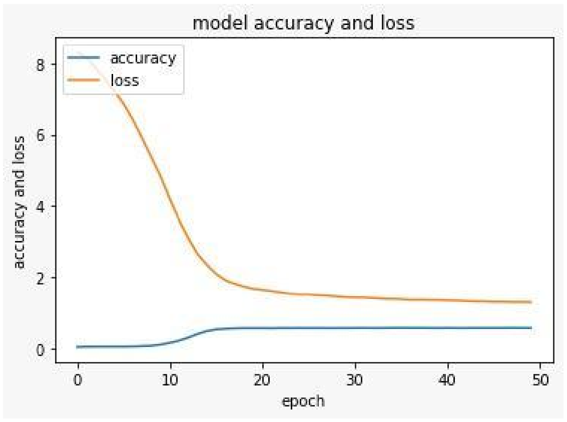
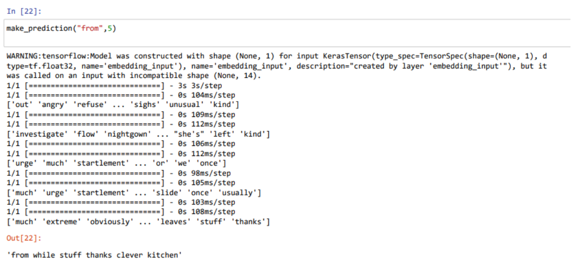


Fig 4.3

This graph indicates very clearly that the loss value (0.8688) is steadily going down, which ultimately results in an accuracy of 0.6628.

Next Word Prediction

Dataset 01 – Metamorphosis

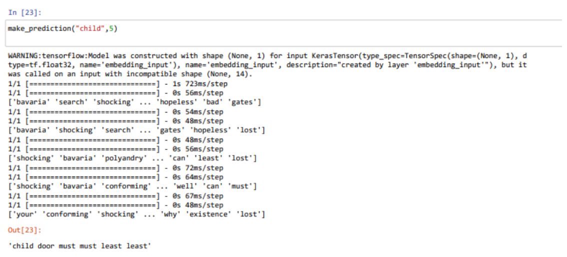


```
In [22]:
make_prediction("from",5)

WARNING:tensorflow:Model was constructed with shape (None, 1) for input KerasTensor(type_spec=TensorSpec(shape=(None, 1), d
type=tf.float32, name='embedding_input'), name='embedding_input', description="created by layer 'embedding_input'"), but it
was called on an input with incompatible shape (None, 14).
1/1 [==============================] - 3s 3s/step
1/1 [==============================] - 0s 104ms/step
['out' 'angry' 'refuse' ... 'sighs' 'unusual' 'kind']
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 112ms/step
['investigate' 'flow' 'nightgown' ... "she's" 'left' 'kind']
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 112ms/step
['urge' 'much' 'startlement' ... 'or' 'we' 'once']
1/1 [==============================] - 0s 98ms/step
1/1 [==============================] - 0s 105ms/step
['much' 'urge' 'startlement' ... 'slide' 'once' 'usually']
1/1 [==============================] - 0s 103ms/step
1/1 [==============================] - 0s 108ms/step
['much' 'extreme' 'obviously' ... 'leaves' 'stuff' 'thanks']
Out[22]:
'from while stuff thanks clever kitchen'
```

Fig 4.4

Dataset 02 – Mother Earth

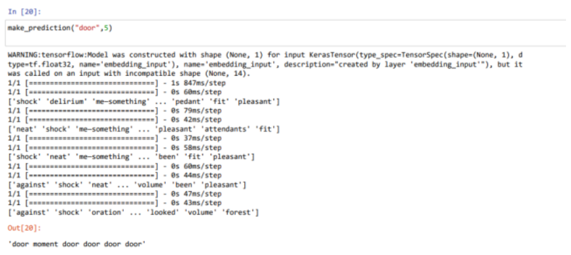


```
In [23]:
make_prediction("child",5)

WARNING:tensorflow:Model was constructed with shape (None, 1) for input KerasTensor(type_spec=TensorSpec(shape=(None, 1), d
type=tf.float32, name='embedding_input'), name='embedding_input', description="created by layer 'embedding_input'"), but it
was called on an input with incompatible shape (None, 14).
1/1 [==============================] - 1s 723ms/step
1/1 [==============================] - 0s 56ms/step
['bavaria' 'search' 'shocking' ... 'hopeless' 'bad' 'gates']
1/1 [==============================] - 0s 54ms/step
1/1 [==============================] - 0s 48ms/step
['bavaria' 'shocking' 'search' ... 'gates' 'hopeless' 'lost']
1/1 [==============================] - 0s 48ms/step
1/1 [==============================] - 0s 56ms/step
['shocking' 'bavaria' 'polyandry' ... 'can' 'least' 'lost']
1/1 [==============================] - 0s 72ms/step
1/1 [==============================] - 0s 64ms/step
['shocking' 'bavaria' 'conforming' ... 'well' 'can' 'must']
1/1 [==============================] - 0s 67ms/step
1/1 [==============================] - 0s 48ms/step
['your' 'conforming' 'shocking' ... 'why' 'existence' 'lost']
Out[23]:
'child door must must least least'
```

Fig 4.5

Dataset 03 – Strange



```
In [20]:
make_prediction("door",5)

WARNING:tensorflow:Model was constructed with shape (None, 1) for input KerasTensor(type_spec=TensorSpec(shape=(None, 1), d
type=tf.float32, name='embedding_input'), name='embedding_input', description="created by layer 'embedding_input'"), but it
was called on an input with incompatible shape (None, 14).
1/1 [==============================] - 1s 847ms/step
1/1 [==============================] - 0s 60ms/step
['shock' 'delirium' 'me-something' ... 'pedant' 'fit' 'pleasant']
1/1 [==============================] - 0s 79ms/step
1/1 [==============================] - 0s 42ms/step
['neat' 'shock' 'me-something' ... 'pleasant' 'attendants' 'fit']
1/1 [==============================] - 0s 37ms/step
1/1 [==============================] - 0s 58ms/step
['shock' 'neat' 'me-something' ... 'been' 'fit' 'pleasant']
1/1 [==============================] - 0s 60ms/step
1/1 [==============================] - 0s 44ms/step
['against' 'shock' 'neat' ... 'volume' 'been' 'pleasant']
1/1 [==============================] - 0s 47ms/step
1/1 [==============================] - 0s 43ms/step
['against' 'shock' 'oration' ... 'looked' 'volume' 'forest']
Out[20]:
'door moment door door door door'
```

Fig 4.6

RESULTS AND DISCUSSION

The graphs of Fig 2.1, Fig 2.2, Fig 2.3 of LSTM and Fig 4.1, Fig 4.2, Fig 4.3 of GRU that have been shown so far make it quite evident that the accuracy of the GRU is higher than that of the LSTM. For each of the three datasets, the accuracy steadily improves, while the amount of loss gradually reduces when using LSTM and GRU models evaluated as shown in the Fig 2.4, Fig 2.5, Fig 2.6 of LSTM and Fig 4.4, Fig 4.5, Fig 4.6 of GRU. Because of the plots, it is easy to observe that the accuracy of GRU steadily improves when the loss is reduced, but the accuracy and loss of LSTM are more prone to fluctuations. As a result of the accuracy value, we are able to draw the conclusion that the GRU model is superior to the LSTM model. Even the loss is rather substantial in the LSTM modes, in contrast to the GRU models. As a consequence of the fact that both versions of the model provide about the same outcome (which is somewhere around 67%), an updated version of the model is required for the purpose of enhancing the research.

ACKNOWLEDGEMENT

We take this opportunity to express our profound gratitude to our project guide, Dr. Kalyani Desikan, Professor, School of Advanced Sciences, for her consistent encouragement and valuable guidance pleasantly offered to us throughout the project work. We are extremely grateful to Dr. Mahalakshmi S, Professor and Dean of the School of Advanced Sciences, VIT Chennai, for extending the facilities of the school towards our project and for her unstinting support. We express our thanks to our Head of the Department Dr. Muthunagai for her support throughout this project.

We also take this opportunity to thank all the faculty of the school for their support and their wisdom imparted to us. We thank our parents, family, and friends for their unconditional support throughout our project and for the opportunity they have provided for us to pursue this programme in such a prestigious institution.

REFERENCES

1. Afika Rianti , Suprih Widodo , Atikah Dhani Ayuningtyas , Fadlan Bima Hermawan. “Next Word Prediction usin LSTM” Journal of information technology and its utilization volume 5, issue 1, June-2022 EISSN 2654-802X.

2. Keerthana N, Harikrishnan S, Konsaha Buji M, Jona J B. “Next Word Prediction”. 2021, IJCRT Volume 9, Issue 12 December 2021, ISSN(2320-2882).

3. Sourabh Ambulgekar, Sanket Malewadikar, Raju Garande, and Dr. Bharti Joshi4 “Next Words Prediction Using Recurrent NeuralNetworks” ITM Web of Conferences 40, 03034 (2021) ICACC-2021

13. s-lstm-863b0b7b1573

4. Deepal S. Thakur, Rajiv N. Tarsarya, Akshay A. Vaskar, Ashwini Save “A Survey on Text Prediction Techniques” VIVA-Tech International Journal for Research and Innovation Volume 1, Issue 2 (2019) ISSN(2581-7280)

5. Soam, M., & Thakur, S. (2022, January). Next Word Prediction Using Deep Learning: A Comparative Study. In 2022 12th International Conference on Cloud Computing, Data Science & Engineering (Confluence) (pp. 653-658). IEEE.
6. Stremmel, J., & Singh, A. (2021, April). Pretraining federated text models for next word prediction. In Future of Information and Communication Conference (pp. 477-488). Springer, Cham.
7. Hamarashid, H. K., Saeed, S. A., & Rashid, T. A. (2021). Next word prediction based on the N-gram model for Kurdish Sorani and Kurmanji. Neural Computing and Applications, 33(9), 4547-4566.
8. Eisape, T., Zaslavsky, N., & Levy, R. (2020). Cloze distillation: Improving neural language models with human next-word prediction. Association for Computational Linguistics (ACL)
9. https://www.analyticsvidhya.com/blog/2021/08/predict-the-next-word-of-your-text-using-long-short-term-memory-lstm/
10. https://www.kaggle.com/code/ysthehurricane/next-word-prediction-bi-lstm-tutorial-easy-way/notebook
11. https://towardsdatascience.com/illustrated-guide-to-lstms-and-gru-s-a-step-by-step-explanation-44e9eb85bf21
12. https://medium.com/analytics-vidhya/rnn-vs-gru-v