Share

CS6910 Assignment 3: Use recurrent neural networks to build a transliteration system

The goal of this assignment is threefold: (i) learn how to model sequence-to-sequence learning problems using Recurrent Neural Networks (ii) compare different cells such as vanilla RNN, LSTM and GRU (iii) understand how attention networks overcome the limitations of vanilla seq2seq models

Aoukshana Vishwas Chaphekar cs22m019

Problem Statement

In this assignment, I used a sample of the Aksharantar dataset released by AI4Bharat. This dataset contains pairs of the following form:

x,y

ajanabee,अजनबी

i.e., a word in the native script and its corresponding transliteration in the Latin script (how we type while chatting with our friends on WhatsApp, etc). Given many such $(x_i, y_i)_{i=1}^{n}$ pairs, goal is to train a model $y = \hat{f}(x)$ which takes as input a romanized string (ghar) and produces the corresponding word in Devanagari (घर).

This is the problem of mapping a sequence of characters in one language to a sequence of characters in another. This is a scaled-

down version of the problem of translation where the goal is to translate a sequence of words in one language to a sequence of words in another language (as opposed to a sequence of **characters** here).

I have referred to this blog to understand how to build neural sequence-to-sequence models.

Question 1

About the model

An RNN-based seq2seq model has been built that includes the following layers: (i) input layer for character embeddings (ii) one or more encoder RNN which sequentially encodes the input character sequence (iii) one or more decoder RNN which takes the last state of the encoder as input and produces one output character at a time using the most recent state of the encoder. (iv) A dense layer with n outputs, which is equal to the number of letters in the target language.

The code is flexible enough such that the dimension of the input character embeddings, the hidden states of the encoders and decoders, the cell (RNN, LSTM, GRU), and the number of layers in the encoder and decoder can be changed.

(a) Computational Complexity (for forward pass only)

Recurrent neural network architecture:

Notations used:

- Input (I)
- Recurrent (R_i)
- activation (A_i)
- dense (D_i)
- Embedding (E)

$$I \rightarrow E \rightarrow R_1^{encoder} A_1 \rightarrow R_2^{decoder} A_2 \rightarrow D_1 \rightarrow E^{-1} \rightarrow O$$

Assumptions made:

- Input embedding size = m,
- Encoder and Decoder both have a single layer,
- The hidden cell state is of size k for both the encoder and

decoder
- The length of the input and output sequence is the same, i.e., T
- The size of the vocabulary is the same for the source and target language, i.e., V

For a standard recurrent layer, after activation, we obtain outputs and states.The shape of output for 1 word: $(1, T, k)$ and states: $(1, k)$

Following are the mathematical formulations for the encoder and decoder model :

$$
\begin{gathered}
X = E(I) \\
h_i^{encoder} = \sigma^{encoder}(W^{encoder} x_i + U^{encoder} h_{i-1}^{encoder} + b^{encoder}) \\
s_0 = h_T \\
s_i^{decoder} = \sigma^{decoder}(W^{decoder} y_i^{decoder} + U^{decoder} s_{i-1}^{decoder} + b^{decoder}) \\
y_i^{decoder} = \sigma_{out}(V^{decoder} s_i^{decoder} + c^{decoder}) \\
y_i^{dense} = \sigma^{dense}(W^{dense} y_i^{decoder} + b^{dense})
\end{gathered}
$$

- $x_i \in \{0,1\}^m$ where $x_i \in X \subset \mathbb{R}^{m \times T}$ and $I \in \mathbb{R}^{V \times T}$
- $\sigma^{encoder}, \sigma^{decoder}, \sigma^{out}$ are sigmoid
- σ^{dense} is softmax
- Parameters:

  $W^{encoder}, W^{decoder}, \in \mathbb{R}^{k \times m}$

  $b^{encoder}, b^{decoder} \in \mathbb{R}^k$

  $V^{decoder}, U^{encoder}, U^{decoder} \in \mathbb{R}^{k \times k}$

  $c^{decoder} \in \mathbb{R}^k$

  $W^{dense} \in \mathbb{R}^{m \times k}$

  $b^{dense} \in \mathbb{R}^m$
- Also, $i = 1..T$ and $h_i^{decoder}, s_i^{encoder}, y_i^{decoder} \in \mathbb{R}^k, y_i^{dense} \in \mathbb{R}^m$

Activation functions :

1. Sigmoid / ReLU / softmax activation functions are assumed to take $O(n)$ computations, where n corresponds to the number of elements to be activated. Let us assume these computations equal n for our convenience.
2. Each multiplication operation would then be $O(lmn)$.

The operations involved in the recurrent encoder layer, $f_{rec,enc}$ are :

$$f_{rec}^{i} = O(k \times m \times m + k \times k \times m)$$
$$f_{rec}^{encoder} = T \times f_{rec}^{i}$$
$$\sigma(f_{rec}^{encoder}) = O(k)$$
$$f_{rec,enc}^{Total} = O(k^2Tm^2 + k^3Tm)$$

The operations involved in recurrent decoder layer, $f_{rec,dec}$ are:

$$f_{rec}^{i} = O(k \times k \times k)$$
$$f_{rec}^{decoder} = T \times f_{rec}^{i}$$
$$\sigma(f_{rec}^{decoder}) = O(k)$$
$$f_{rec,dec}^{Total} = O(k^4T)$$

The operations involved in the dense layer are:

$$f_{den}^{i} = O(k \times m \times m)$$
$$f_{den} = T \times f_{den}^{i}$$
$$\sigma(f_{den}^{decoder}) = O(m)$$
$$f_{rec,dec}^{Total} = O(kTm^3)$$

Therefore, the total number of operations involved are :

$$f^{Total} = f_{rec,enc}^{Total} + f_{rec,dec}^{Total} + f_{den,dec}^{Total}$$
$$f^{Total} = O(k^4T + k^3Tm + k^2Tm^2 + kTm^3)$$

▾ (b) Number of parameters

We have :

$W^{encoder}, W^{decoder}, \in \mathbb{R}^{k \times m}$

$b^{encoder}, b^{decoder} \in \mathbb{R}^{k}$

$V^{decoder}, U^{encoder}, U^{decoder} \in \mathbb{R}^{k \times k}$

$c^{decoder} \in \mathbb{R}^{k}$

$W^{dense} \in \mathbb{R}^{m \times k}$

$b^{dense} \in \mathbb{R}^{m}$

- Encoder: $k^2 + km + k$ parameters from U, W and b respectively.
- Decoder: $2k^2 + km + 2k$ parameters from V, U, W and b respectively.
- Dense layer: $km + m$ from W and b respectively.

Total - $3k^2 + 3km + 3k + m$ parameters are present in the network.

Question 2

English - Hindi transliteration dataset was used for this assignment.Here I have assumed that the number of layers for encoder and decoder is same.

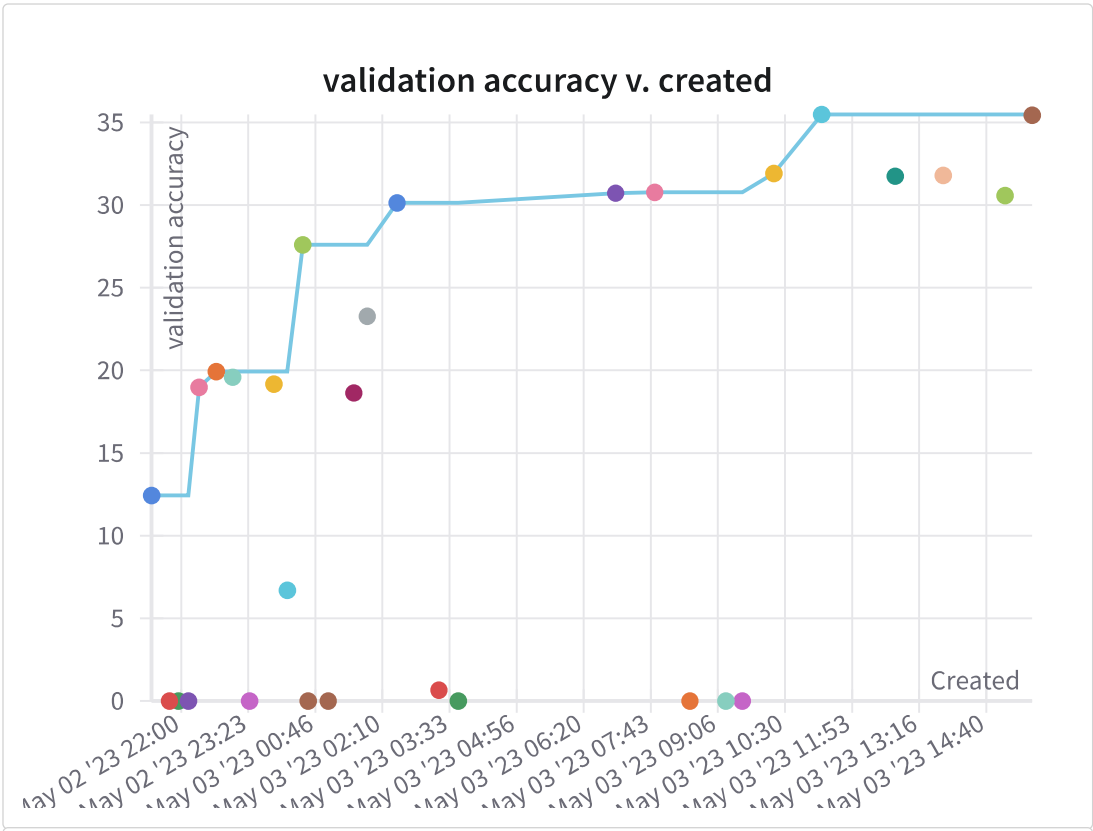
Hyper parameters explored:

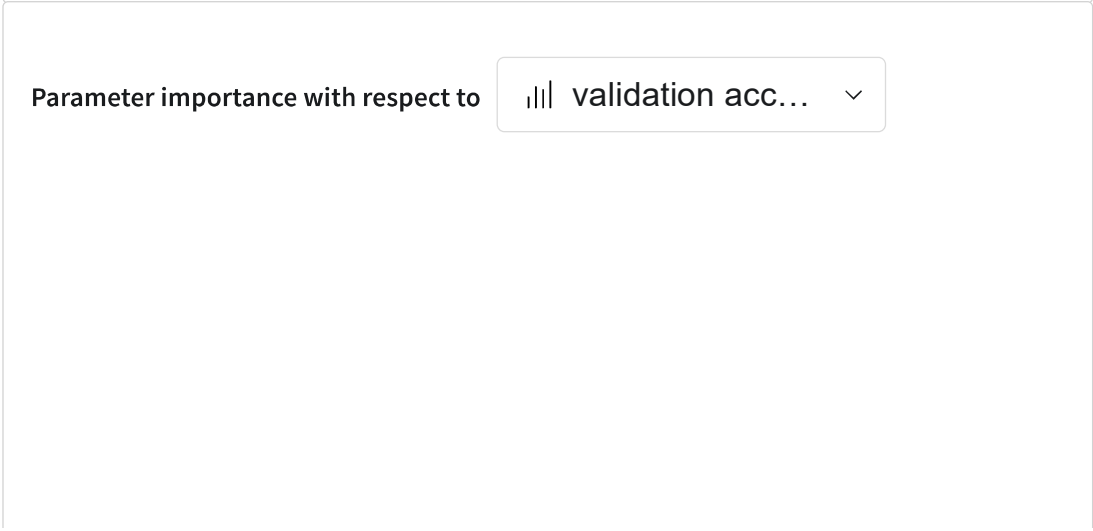
- Epochs : 5,10
- Hidden layer size : 128, 256,512
- Cell Type : RNN, LSTM, GRU
- Input embedding size/ latent dimension : 64, 128, 256,512
- Number of encoder/decoder layers : 1, 2, 3
- Bidirectional : True, False
- Dropout : 0%, 20%, 30%
- Optimizer : NAdam
- Learning rate : 1e-2, 1e-3
- Batch size : 32, 64, 128

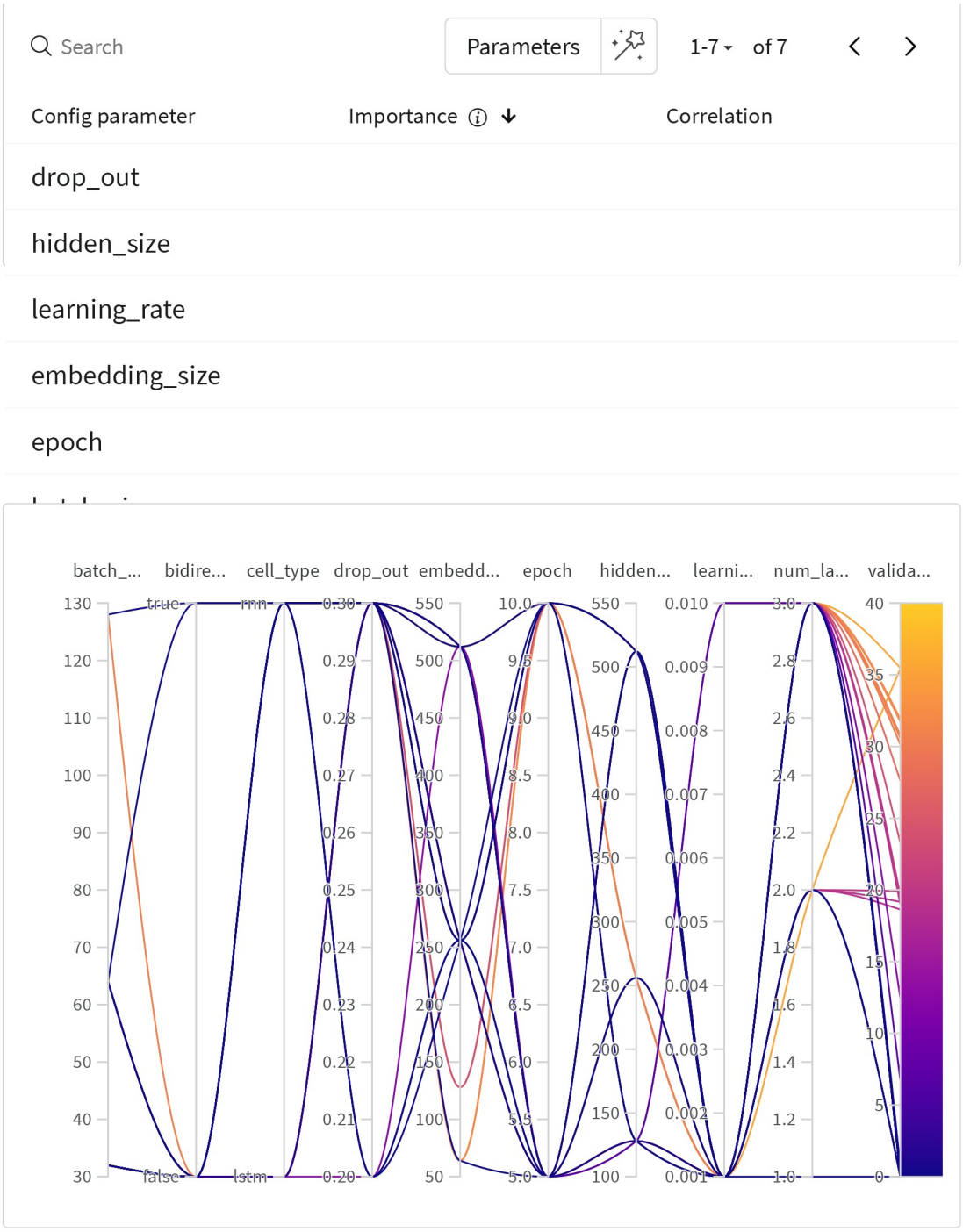
Bayesian hyper-parameter search was used to effeciently search for hyperparameters giving best accuracy.

While running the sweeps, Bayesian Search never used GRU as a cell type. To make the search complete, 29 runs were trained using Bayesian Sweep and 10 runs were trained with only_GRU as an option for cell type using Bayesian search.

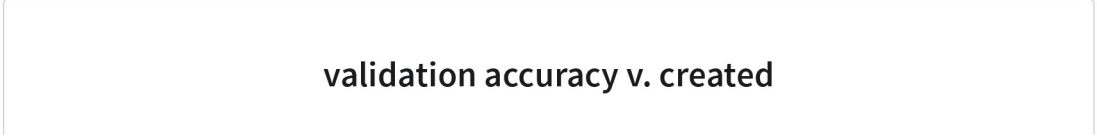
The validation accuracy vs created plots, correlation summary and parallel coordinates plots for Bayesian search are as follows :







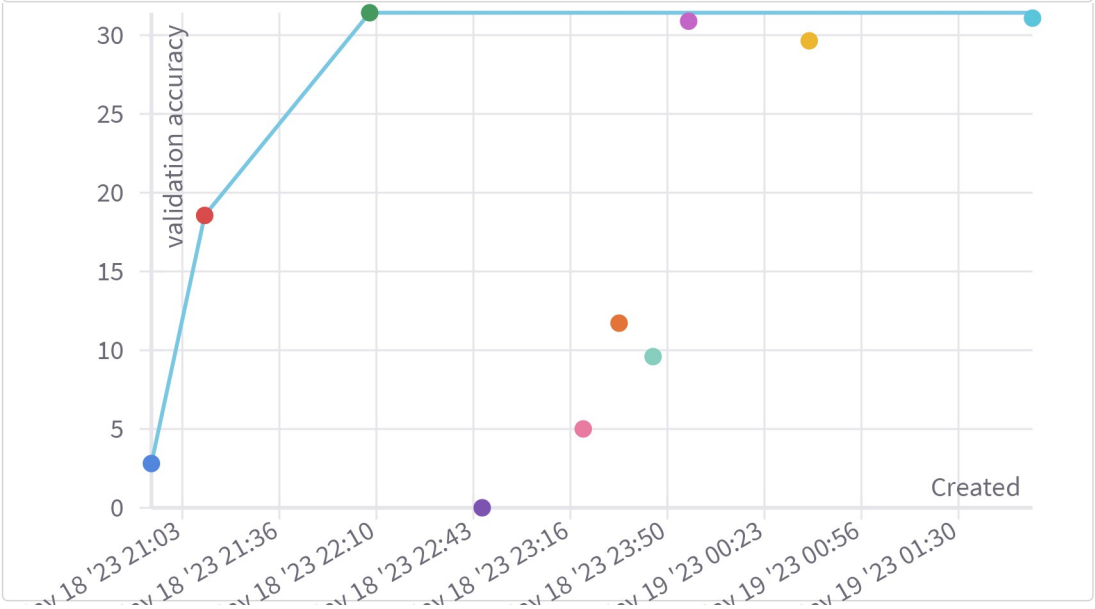
The validation accuracy vs created plots, correlation summary and parallel coordinates plots for GRU are as follows :

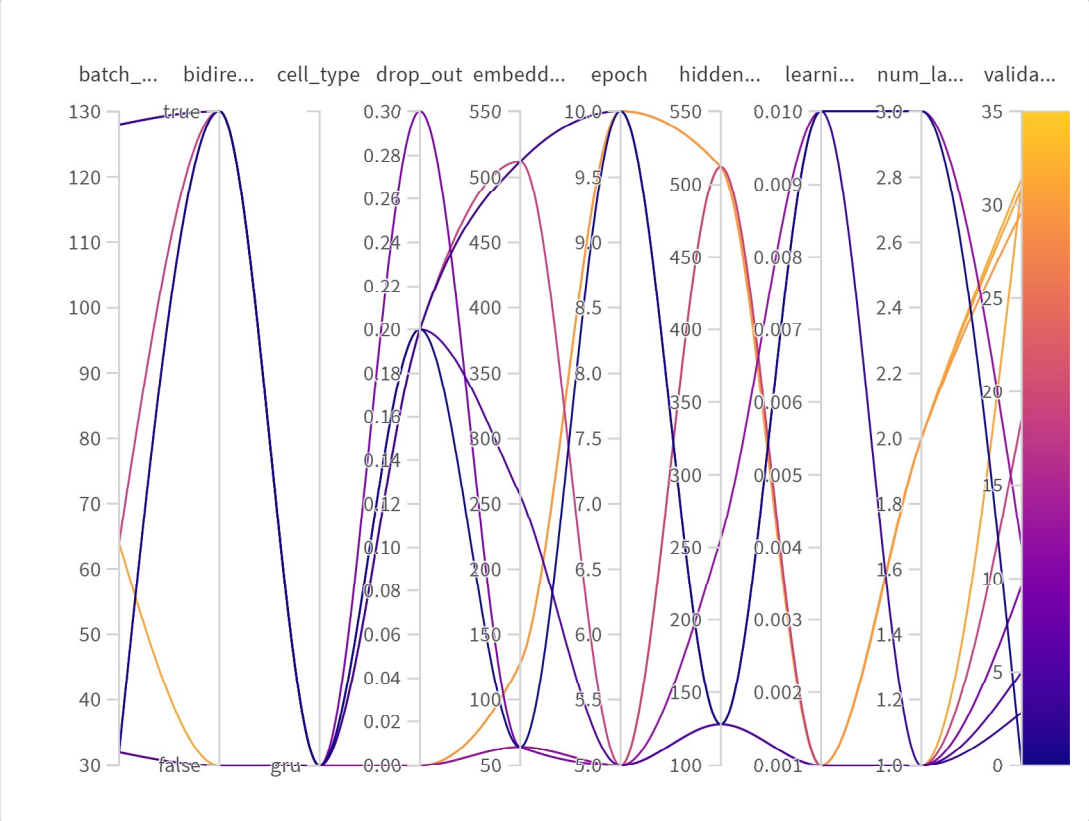


Parameter importance with respect to validation acc...

Search Parameters 1-8 of 8

Config parameter	Importance	Correlation
hidden_size		
epoch		
num_layers		
learning_rate		
drop_out		




batch_...
bidire...
cell_type
drop_out
embedd...
epoch
hidden...
learni...
num_la...
valida...
true
false
gru

Question 3

- As seen from the plots,the configurations with LSTM gave best accuracy (35.474%) while GRU gave 31.421% accuracy.
- LSTM and GRU work better than RNN because of their ability to selective read,write and forget the information obtained from the previous state.
- Hidden size has a large impact to improve accuracy as seen from the correlation plots (They have positive correlation with the accuracy). This is because the patterns existing in the input can be better represented by larger hidden size of the dense layer.The optimal hidden size chosen by the search is 256
- However,increasing learning rate has a negative impact on accuracy as it is seen from the correlation plot that learning rate is negatively correlated with accuracy which leads to increase in learning rate,decrease in accuracy.
- We have used NAdam as an optimizer for our model and low learning rates appear to be effective for NAdam.Higher learning rates could result in too big update shifting the mdoel away from minima.
- Bidirectional model works better than unidirectional model because each state in bidirectional model includes the state that came before and after it.This is not what happens in unidirectional models where just the last character of a word would receive the context of the full word.
- Number of epochs,batch size and number of encoder/decoder layers hardly affect the validation accuracy as seen from the correlation summary.
- Dropout has positive correlation with accuracy when used with RNN and LSTM whereas it has negative correlation with validation accuracy when used with GRU.
- Increasing number of epochs does not affect the validation accuracy in a way that once the accuracy has reached a certain point,no improvement in the model has been made even if we increase the number of epochs i.e the accuracy saturates and there is no further

improvement in it.

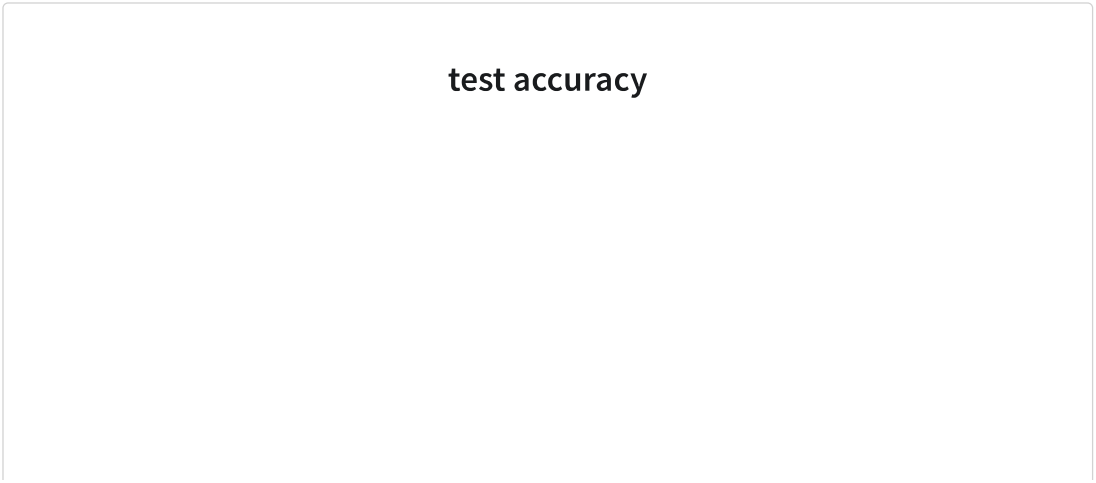
- In case of RNN,the information from previous states vanishes as more and more information gets added in the next state.So accuracy obtained using RNN is nearly 0.
- Accuracy obtained by LSTM - 35.474% , accuracy obtained by GRU - 31.421% . These are close to each other as both LSTM and GRU make use of read,write and forget gates.

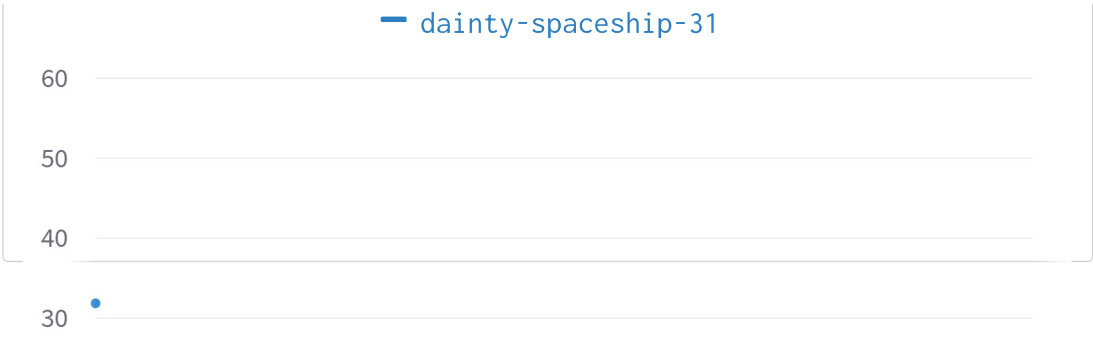
Question 4

(a) The hyperparameters obtained from the best model from the sweeps:

- Epochs : 10
- Hidden layer size : 512
- Cell Type : LSTM
- Input embedding size/ latent dimension : 256
- Number of encoder/decoder layers : 3
- Bidirectional : False
- Dropout : 30%
- Optimizer : NAdam
- Learning rate : 1e-3
- Batch size : 64

Using these hyperparameters the accuracy obtained on the test data set - 31.86 %





(l redictions made by the best model

ut Word (English)	Target Word (Devanagari)	Predicted Word (Without Attention)
uwo	टापुओं	तपुववो
vinses	ग्रातिमेम	ग्रातिमेम
shrikhand	श्राखड	श्राख्द
shreeganganagarwaasiyon	श्रीगंगानगरवासियों	श्रीगंगागामवियों
nations	नेशंस	नेशन्स
agusta	अगस्ता	अगुस्ता
lehna	लहना	लहना
kanghe	कंघे	कांघे
ainth	ऐंठ	ऐंथ
vyavasthaamoolak	व्यवस्थामूलक	व्यवस्थालमूक
kamzor	कमज़ोर	कामजोर
renuka	रेनुका	रेनुका
chhaapemaar	छापेमार	छापेमार
krantiyon	क्रांतियों	क्रांतियों
purkayastha	पुरकायस्थ	पुर्कास्था

Predictions made by the model

(c) Comments on the error made by the model

- The model is getting confused with the diacritics (maatraa) in Devanagari.
- For example,in many words as seen from the table above, अ is misinterpreted as आ as seen the word "कंघे" is misinterpreted as "कांघे" and "कमज़ोर" is interpreted as "कामजोर".
- The model is getting confused between letters त,ड,द,ठ,थ.
- This is evident because the word "टापुओं" is misinterpreted as "तपुववो" ,similarly "श्रीखंड" is misinterpreted as "श्रीखंद" and "ऐंठ" is misinterpreted as "ऐंथ".
- In some cases the model is not predicting the entire word.
- For example,the word "श्रीगंगानगरवासियों" is interpreted as "श्रीगंगागामवियों" by the model.

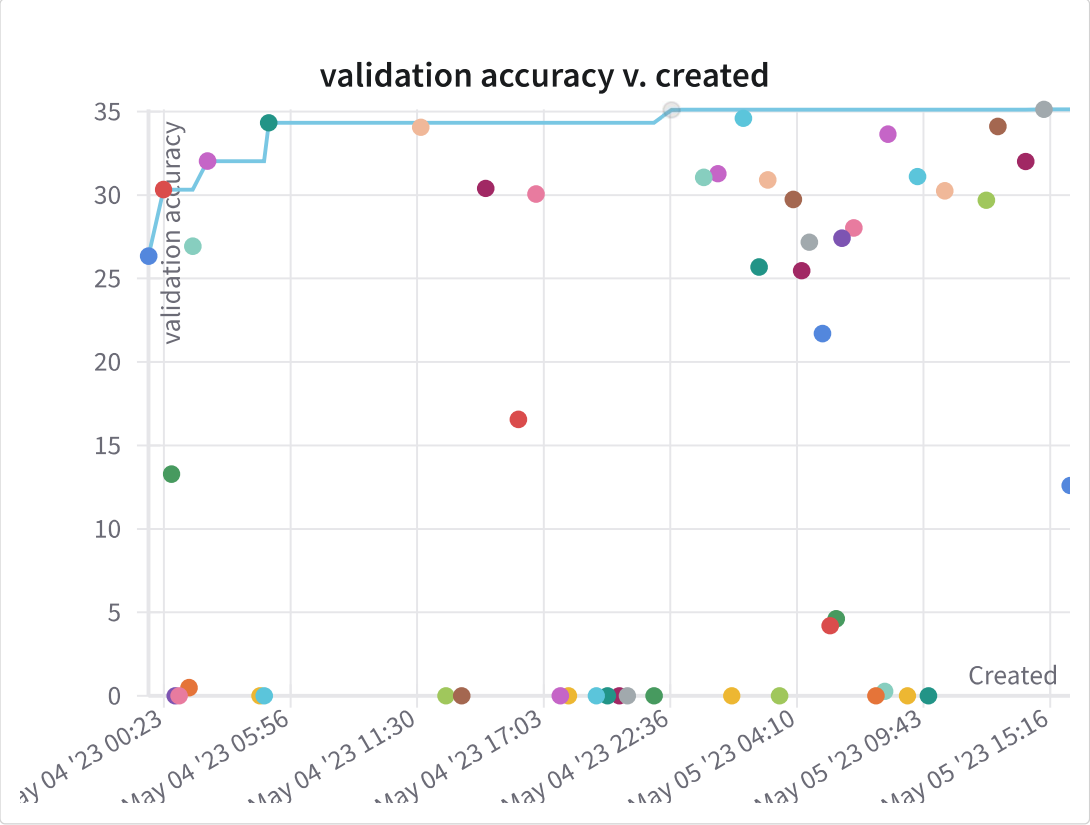
- Also the model is getting confused in predicting the words which have ' . ' in them.
- For example,the word "नेशंस" is interpreted as "नेशन्स" and "श्रीखंड" as "श्रीखंद".

Question 5

(a) Hyperparameter Sweeps

Plots added:

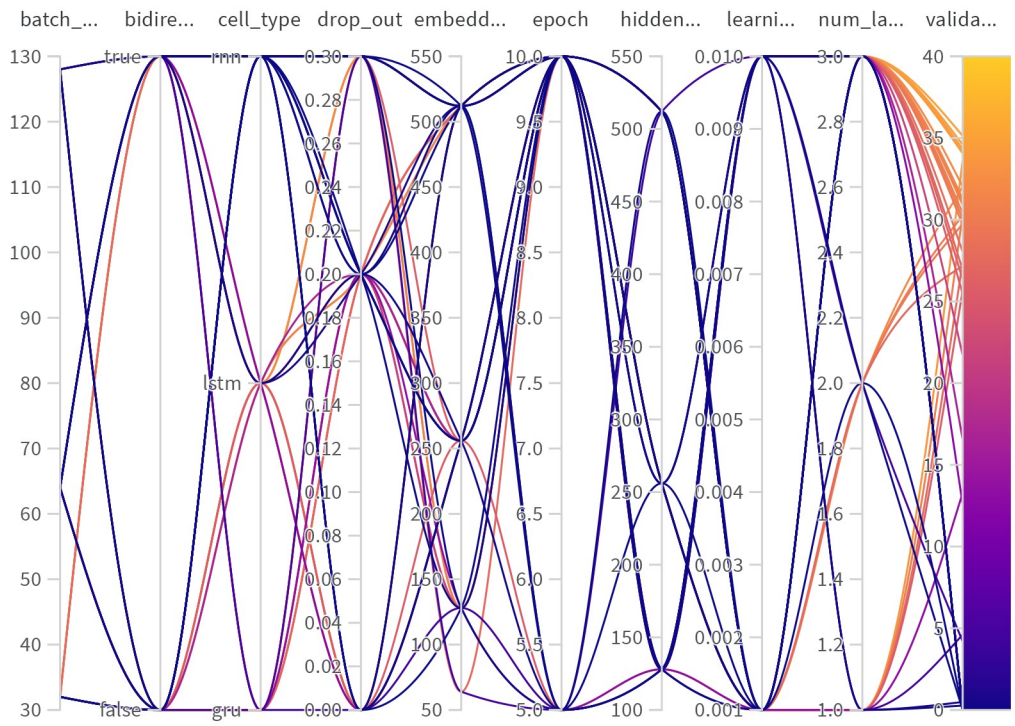
- Validation accuracy vs created
- Parallel Co-ordinate Plot
- Correlation Summary Plot



Parameter importance with respect to validation acc...

Search | Parameters | 1-8 of 8

Config parameter	Importance	Correlation
epoch		
learning_rate		
batch_size		
drop_out		
bidirectional		
embedding_size		



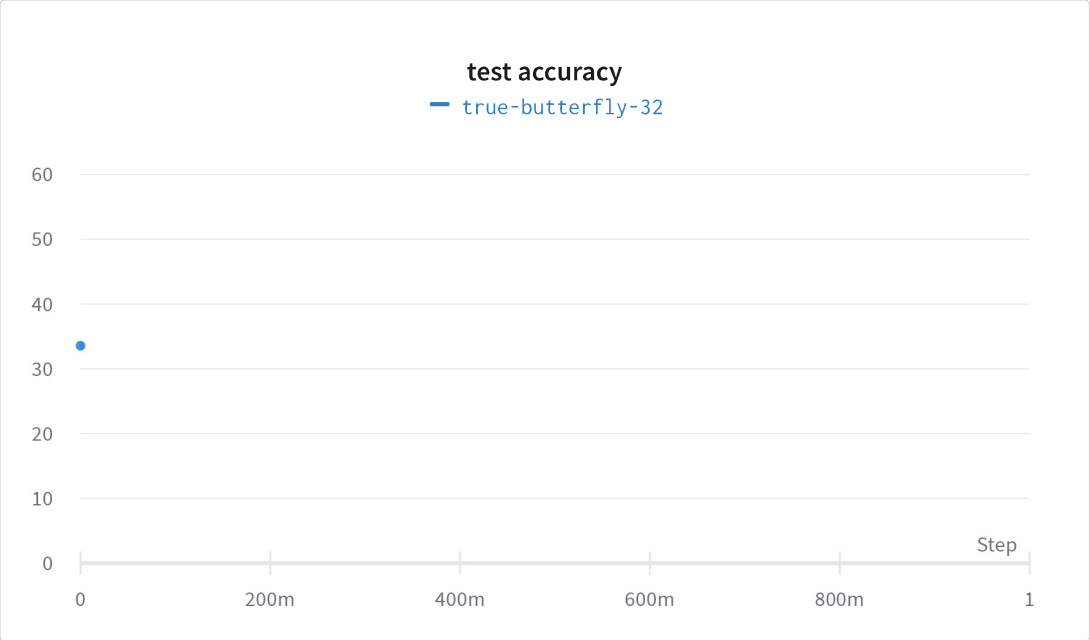
Observations:

- Adding attention to our model improves accuracy.
- The accuracy obtained without attention on test data set was 31.86% whereas on applying attention the accuracy increased to 33.569%.
- As seen from the parallel coordinates plot,as we increase the number of layers for encoder and decoder in the model,the accuracy also increases.
- This can also be seen from the correlation summary where number of layers are in positive correlation with the validation accuracy.
- As seen from the correlation summary,epoch,dropout,embedding size,hidden size and number of layers for encoder/decoder are in positive correlation with validation accuracy whereas learning rate,batch size and bidirectional are in negative correlation with validation accuracy.
- The model gives best accuracy for LSTM which is evident because it uses selective read,write and forget gates which help us store only useful information from the previous state.

(b) The hyperparameters obtained from the best model from the sweeps (with attention):

- Epochs : 10
- Hidden layer size : 512
- Cell Type : LSTM
- Input embedding size/ latent dimension : 256
- Number of encoder/decoder layers : 3
- Bidirectional : False
- Dropout : 0
- Optimizer : NAdam
- Learning rate : 1e-3
- Batch size : 128

Using these hyperparameters the accuracy obtained on the test data set - 33.569 %



Predictions made by the model

Input Word (English)	Target Word (Devanagari)	Predicted Word (With Attention) (Devanagari)
jadhav	जाधव	जाधव
peepaadh	पीपाड़	पीपाढ़
gehlot	गहलोत	गेलो
imaandaari	इमानदारी	इमानदारी
reeta	रीता	रीता
mangalsinh	मंगलसिंह	मंगलसिंह
bean	बीन	बीन
airway	एयरवे	एयरवे
dablyoopeedee	डब्ल्यूपीडी	डैब्लयूपीडी
parmeshvarpur	परमेश्वरपुर	परम्शेवरपुर
amargad	अमरगढ़	अमरगाड़
eksath	एकसाथ	एकसाथ
sanshyon	संशयों	संश्यों
davenport	डेवनपोर्ट	डेवनपोर्ट
manhattan	मैनहट्टन	मानहट्टन

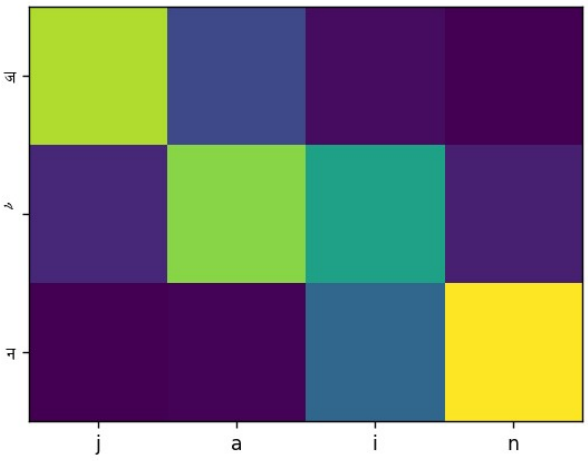
(c) Inferences

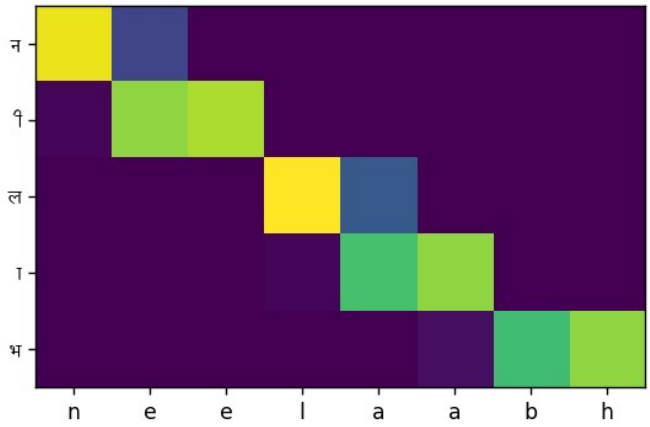
- The attention based model performed better than the model without attention.
- Attention based model corrected many errors made by the model

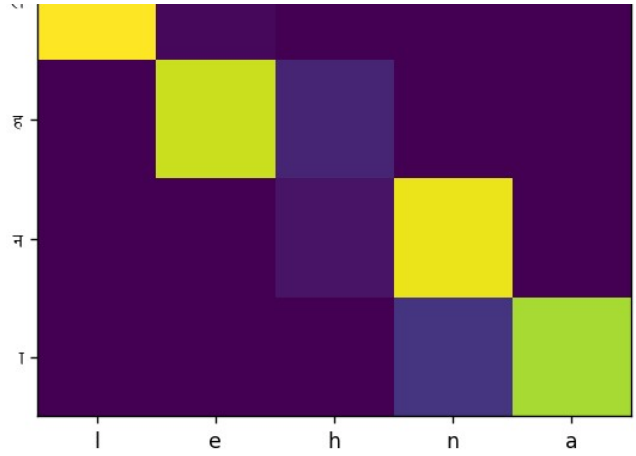
without attention.

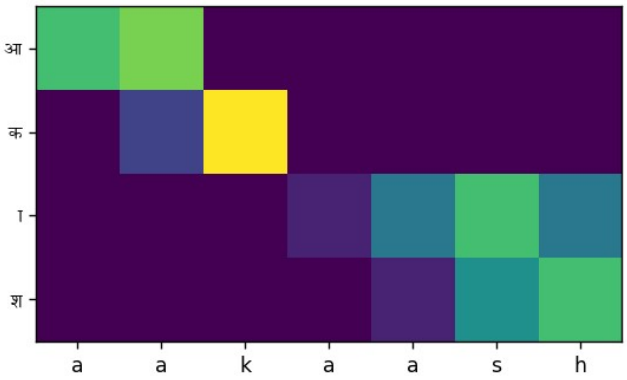
- For example,the model no longer making mistakes with the maatraa अ and आ .
- For example, the words "जाधव" and "रीता" are correctly interpreted by the model.
- But the model is still confused between the character ड़,ढ़ as seen the word "पीपाड़" is misinterpreted as "पीपाढ़" and "अमरगढ़" as "अमरगाड़".
- The model is also making mistakes for predicting the correct target output for some non-hindi words.
- For example,"मैनहट्टन" is predicted as "मानहट्टन".

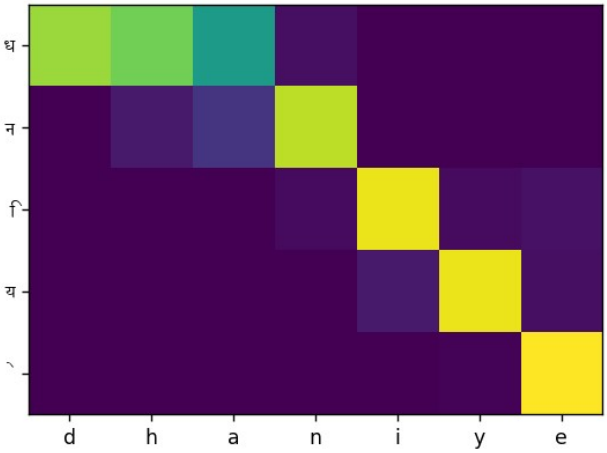
(d) HeatMaps

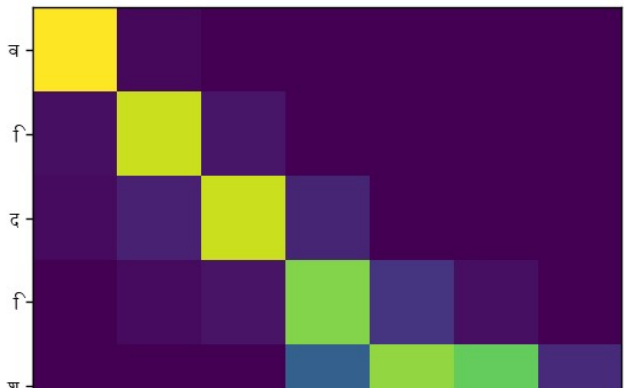


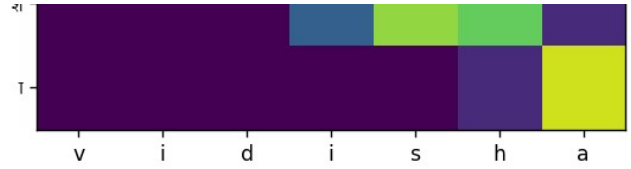


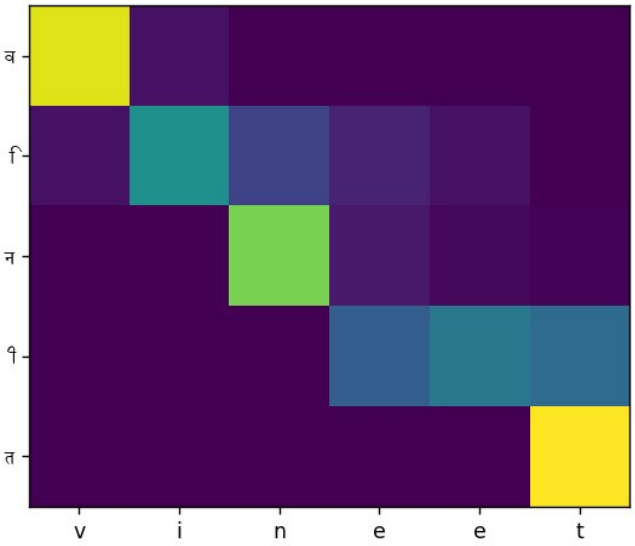

ह
न
ा
l
e
h
n
a

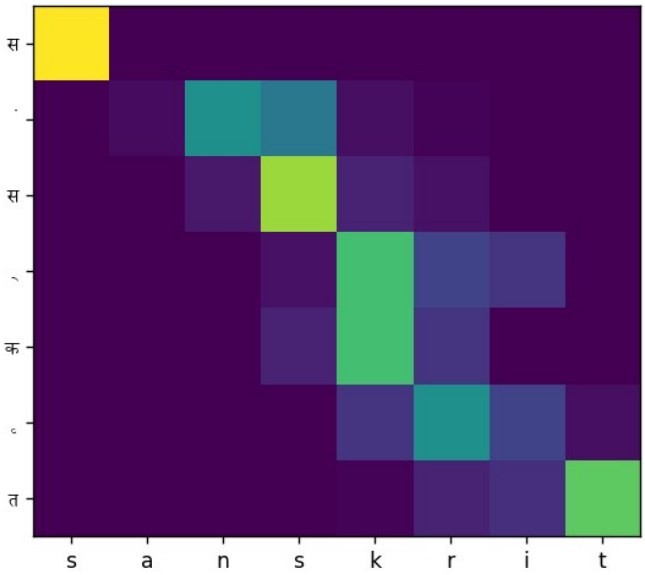

आ
क
ा
श
a
a
k
a
a
s
h

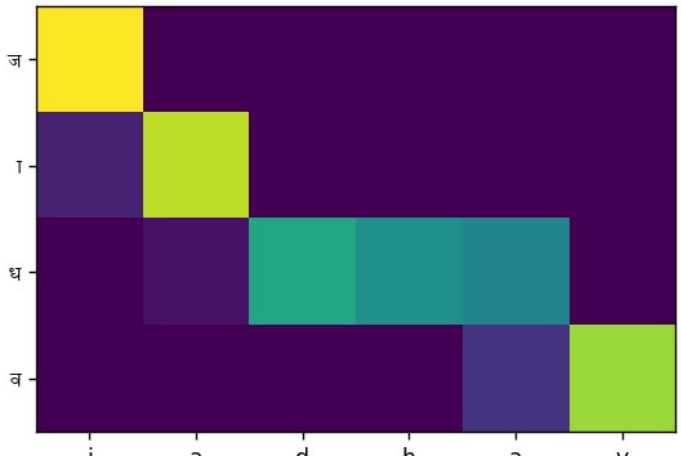

ध
न
ि
य
d
h
a
n
i
y
e


व
ि
द
ि

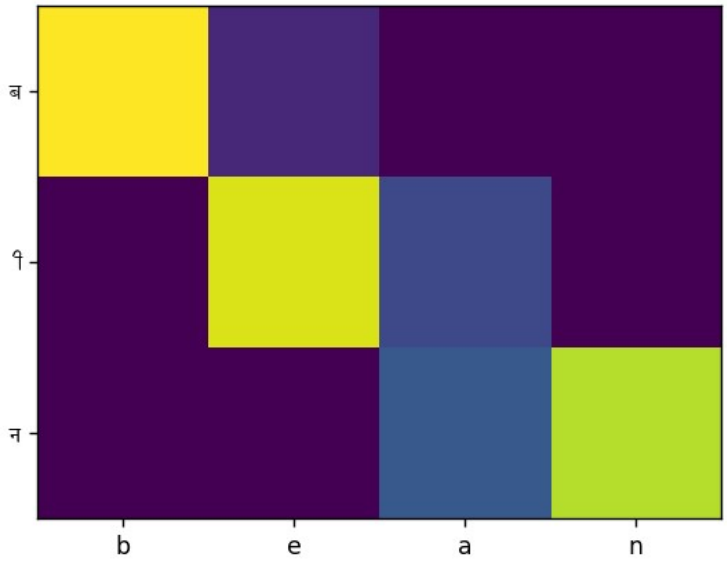

v i d i s h a


व
ि
न
ी
त
v i n e e t


स
स
क
त
s a n s k r i t


ज
ा
ध
व

j a u n a v



Question 7

Github Link: https://github.com/AoukshanaVChaphekar/DL-Assignment-3

Self Declaration

I, Aoukshana Chaphekar (Roll no: CS22M019), swear on my honour that I have written the code and the report by myself and have not copied it from the internet or other students.I have referred to this blog to understand how to build neural sequence-to-sequence models.

Created with ❤️ on Weights & Biases.