Sequence – Sequence Lab Documentation by Mohamed Hafez

Note: In this documentation, I will provide the purpose and importance of each cell in my notebook. Also, please hide the output of cells 14 & 15 after running them as it is extremely long. Here is the link to my notebook on Kaggle: https://www.kaggle.com/code/mohamedhafez885/english-hindi-machine-translation-rnns/notebook

Cell 1:

- **Purpose:** Importing necessary libraries and modules.
- **Importance:** These libraries are important for data manipulation, visualization, preprocessing, and building the neural network model.

```
In [2]:
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns

import string
from string import digits
import re

from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split
from keras.layers import Input, LSTM, Embedding, Dense
from keras.models import Model

from collections import Counter
```

Cell 2:

- **Purpose:** Loading the dataset from a CSV file.
- **Importance:** It reads the dataset for training the model.

```
In [3]:
data = pd.read_csv("/kaggle/input/hindienglish-corpora/Hindi_English_Truncated_Corpus.csv", encoding = 'utf-8')
data.head(7)
```

Out[3]:

	source	english_sentence	hindi_sentence
0	ted	politicians do not have permission to do what ...	राजनीतिज्ञों के पास जो कार्य करना चाहिए, वह कर...
1	ted	I'd like to tell you about one such child,	मई आपको ऐसे ही एक बच्चे के बारे में बताना चाहू...
2	indic2012	This percentage is even greater than the perce...	यह प्रतिशत भारत में हिन्दुओ प्रतिशत से अधिक है।
3	ted	what we really mean is that they're bad at not...	हम ये नहीं कहना चाहते कि वो ध्यान नहीं दे पाते
4	indic2012	.The ending portion of these Vedas is called U...	इन्हीं वेदों का अंतिम भाग उपनिषद कहलाता है।
5	tides	The then Governor of Kashmir resisted transfer...	कश्मीर के तत्कालीन गवर्नर ने इस हस्तांतरण का व...
6	indic2012	In this lies the circumstances of people befor...	इसमें तुमसे पूर्व गुज़रे हुए लोगों के हालात हैं।

Cell 3 – Cell 18: *(Unfortunately I can't show all cells, so please check the rest in the notebook)*

- **Purpose:** Data pre-processing steps like filtering, cleaning duplicates, handling null values, and text normalization.

- **Importance:** These steps are crucial for preparing the dataset before training the model.

In [13]:
```
def preprocess_text(text):
    text = text.lower()
    text = re.sub("'", '', text)
    exclude = set(string.punctuation)
    text = ''.join(ch for ch in text if ch not in exclude)
    remove_digits = str.maketrans('', '', digits)
    text = text.translate(remove_digits)
    text = re.sub("[२३०८९५७४६१]", "", text)
    text = text.strip()
    text = re.sub(" +", " ", text)
    return text


data['english_sentence'] = data['english_sentence'].apply(preprocess_text)
data['hindi_sentence'] = data['hindi_sentence'].apply(preprocess_text)
data['hindi_sentence'] = data['hindi_sentence'].apply(lambda x: 'START_ ' + x + ' _END')
```

In [14]:
```
all_eng_words = set(word for eng in data['english_sentence'] for word in eng.split())
all_hindi_words = set(word for hin in data['hindi_sentence'] for word in hin.split())

print(len(all_eng_words))
all_eng_words
```


Cell 19:

- **Purpose:** Installing a library for exploratory data analysis.
- **Importance:** This library might be used to gain insights into the dataset.

Cell 20: *(This was an amazing library to find out and learn about for the first time, even though I had some errors at the end of my code, I would still use it in the future!)*

- **Purpose:** To install the library 'fasteda'.
- **Importance:** Necessary for the library to perform exploratory data analysis.

In [22]:
```
from fasteda import fast_eda

fast_eda(data[["source", "eng_char_count", "hindi_char_count", "hindi_tok_count",
               "eng_tok_count",]])
```

Cell 21:

- **Purpose:** Performing exploratory data analysis.
- **Importance:** Essential for understanding the data distribution and characteristics.

Cell 22 – Cell 32: *(Cell 30 didn't work even though I spent so much time researching what went wrong here, I found out that "model.fit_generator" is deprecated, tried to get alternatives, but still couldn't get it to work)*

- **Purpose:** Setting up the model architecture using Keras and defining generator functions.
- **Importance:** It constructs the sequence-to-sequence model and prepares data for training.

In [ ]:
```
# Define batch size and number of epochs
batch_size = 128
epochs = 100

# Define sample sizes
train_samples = len(X_train)
val_samples = len(X_test)

# Train the model using fit_generator with the generator function
history = model.fit_generator(generator=generate_batch(X_train, y_train, batch_size=batch_size),
                              steps_per_epoch=train_samples // batch_size,
                              epochs=epochs,
                              validation_data=generate_batch(X_test, y_test, batch_size=batch_size),
                              validation_steps=val_samples // batch_size)
```

Cell 33:

- **Purpose:** Defining a function for decoding the model output.
- **Importance:** This function decodes the model predictions back to readable text.

In [47]:
```
def decode_sequence(input_seq):
    states_value = encoder_model.predict(input_seq)
    target_seq = np.zeros((1, 1))
    target_seq[0, 0] = target_token_index['START_']
    decoded_sentence = ''

    while True:
        output_tokens, h, c = decoder_model.predict([target_seq] + states_value)
        sampled_token_index = np.argmax(output_tokens[0, -1, :])
        sampled_char = reverse_target_char_index[sampled_token_index]
        decoded_sentence += ' ' + sampled_char

        if (sampled_char == '_END' or len(decoded_sentence) > 50):
            break

        target_seq[0, 0] = sampled_token_index
        states_value = [h, c]

    return decoded_sentence
```

Cell 34 – Cell 36:

- **Purpose:** Generating predictions and evaluating them against actual translations.
- **Importance:** It demonstrates how the model performs on unseen data.

In [49]:
```
k+=1
(input_seq, actual_output), _ = next(train_gen)
decoded_sentence = decode_sequence(input_seq)
print('Input English sentence:', X_train[k:k+1].values[0])
print('Actual Hindi Translation:', y_train[k:k+1].values[0][6:-4])
print('Predicted Hindi Translation:', decoded_sentence[:-4])
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 251ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 200ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 25ms/step
Input English sentence: englishspeaking well you name it
Actual Hindi Translation:  अंग्रेज़ीबोलने वाले हर तरह के कार्टूनिस्ट
Predicted Hindi Translation:  रॉकस्टार चुनौती पड़ौसी ट्रेकपैड पड़ौसी दुर्घटनाग्
```

In [50]:
```
k+=1
(input_seq, actual_output), _ = next(train_gen)
decoded_sentence = decode_sequence(input_seq)
print('Input English sentence:', X_train[k:k+1].values[0])
print('Actual Hindi Translation:', y_train[k:k+1].values[0][6:-4])
print('Predicted Hindi Translation:', decoded_sentence[:-4])
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 30ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 26ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 28ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 29ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 31ms/step
Input English sentence: your girlfriend could cheat
Actual Hindi Translation:  किसी की प्रेमिका बेवफा निकल सकती है
Predicted Hindi Translation:  प्राप्ति christ ताड़ प्रचंड रुककेे आक्साइड बहुवाद स्वेच्छासेव
```