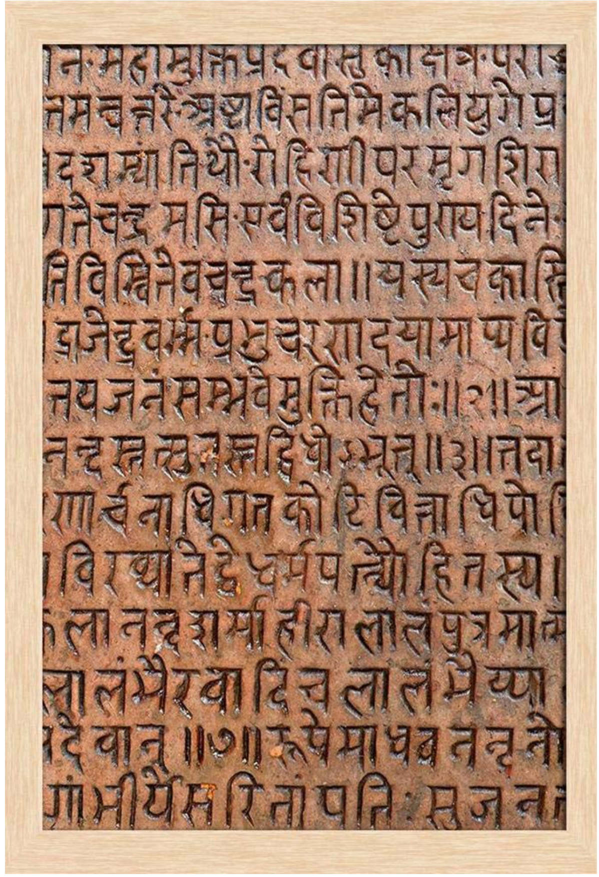
Motivation

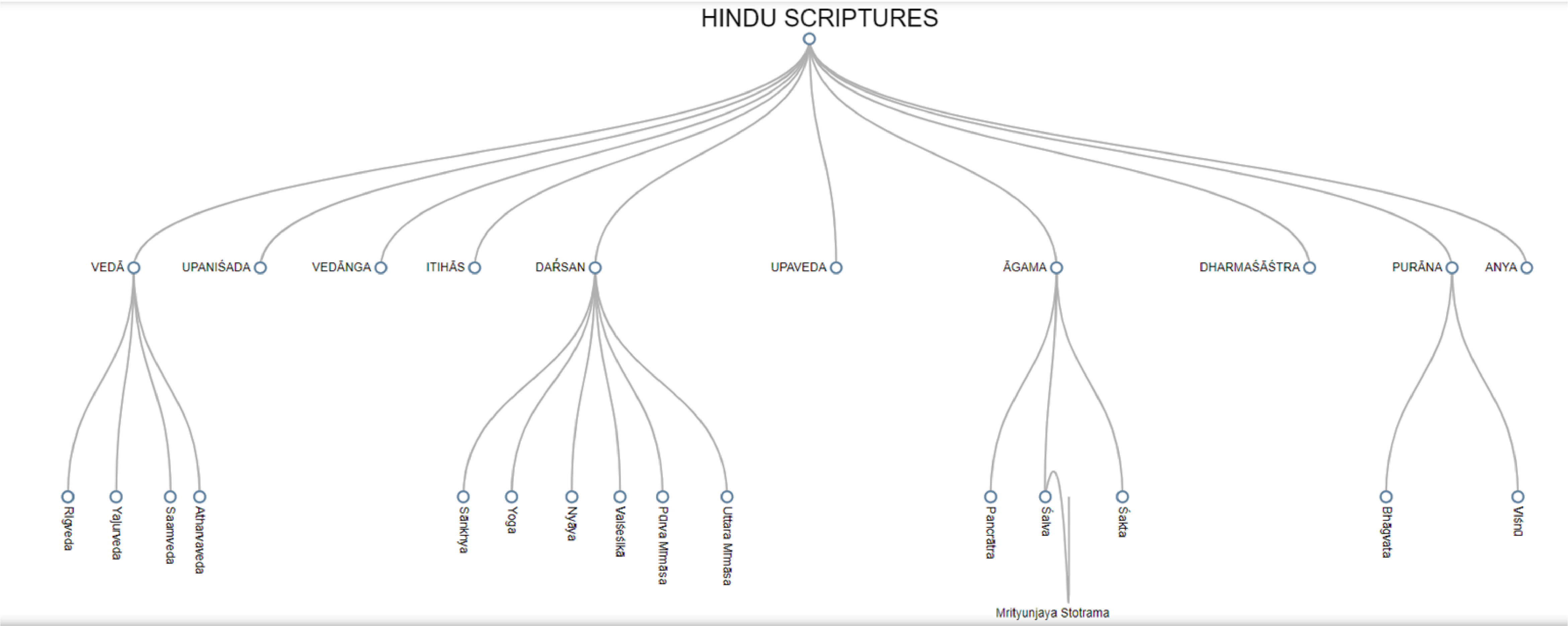
Hindu Scriptures helps develop an evidence-based work of the Vedic language ie. Sanskrit (although not a strict definition although), which provides incomparable understanding of knowledge about “searched things”.

Resources: 10 types of Scriptures including वेद, दर्शन, इतिहास, आगम, उपवेद etc. [Dictionary Platform]

Work Proposal: Model development by Vivek Tripathi and Sanskrit lang. work by Mr. Manoj Bhandari.



Motivation (Data Source)



Introduction (Idea)

- This research is on machine learning model to develop an approach to accurately classify ‘shlokas’ as per the labels (currently ‘type’ category). It is the process of categorizing text into predefined categories based on its content.

- The model is trained using a suitable optimizer and loss function on shlokas and sutras from multiple text, precisely 10 types of scriptures.

- Since entity names are the major part of the text, NER (Named Entity Recognition) is a key step towards more intellectual data mining into the text. NER is the process of recognizing entities (e.g. Person, Place, Organization etc.) associated with a particular word of a text.)

Named Entity Recognition

1. वाचस्पतिः, अग्निः, मातरिश्वा , नारदः, वाल्मीकिः , मनुः, सुद्युम्नः, वसिष्ठः, प्रतिष्ठानम्, पुरूरवाः, पृषध्रः, करूषः, कारूष, शम्भुः, कन्दः, गोरक्षः, गोरक्षशतकम् , त्रिनेत्रः, सरस्वती , ब्रह्मा, दक्षः, अश्वी, देवेशः, भरद्वाजः, पुनर्वसुः, हुताशवेशः, चरकः, पातञ्जलमहाभाष्यम्, गणनायकः, शिवः, पिङ्गलः, पिङ्गलनागः, ज्योतिष्टोमः, पाणिनिः, स्वयंभूः, प्रजापतिः,
2. पारिभाषिक शब्द - ब्रह्म, पुरुषार्थ, योग, चित्त, वृत्ति, चित्तवृत्ति, निरोध, प्रमाण, प्रत्यक्ष, अनुमान, उपमान, शब्द , प्रमेय, संशय, प्रयोजन, दृष्टान्त, सिद्धान्त , अवयव, तर्क, निर्णय, वाद, जल्प, वितण्डा, हेत्वाभास, छल, जाति, निग्रहस्थान, पुरुष, आसन, प्राणायाम स्वर, स्पर्श, आदि.
3. गुण, वृद्धि , धातु,म , य, र, त, स, ज, भ, न, ल, ग , यम् , इक्

1. Challenges of Data Collection

1. Fuzzy SVM is used for NER. It is based on Fuzzy logic which is a type of many-valued logic where the truth values of variables can be any real number between 0 and 1. (Joseph et. al.)

Challenges of Sanskrit Language

1. Lack of capitalization
2. Not fixed SOV (Subject-Object-Verb) order (instead SVO in English).
3. Unavailability of gazetteer lists and ambiguity in Indian names
4. No Trained dataset
5. Highly inflectional, agglutinative and morphologically rich in nature
6. Multiple representations are available in text formats

It's highly conscious job to solve all such challenges particularly due to inflectional nature of language. We used method of string parsing, as much as possible to identify NER using (n-1) string length method.

Introduction (Idea)

- Classification is performed using an LSTM-based model. LSTM (Long Short-Term Memory) is a type of recurrent neural network (RNN) that is well-suited for sequence-based classification problems like texts.

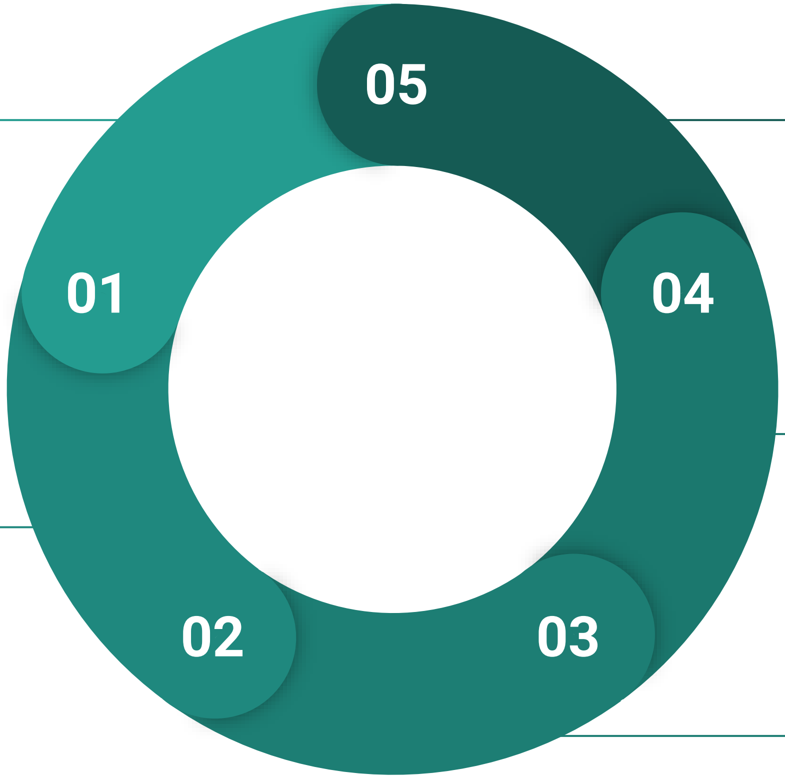
- With careful selection of hyperparameters and tuning of the model (Train-test split ratio, Batch Size, Number of Epochs, Branches in Decision Tree, Number of clusters in Clustering Algorithm, learning rate for training a neural network etc.), it is possible to achieve high accuracy in classification of Sanskrit Shlokas, so we opted real-time parameter for other different models as well to draw up a conclusion.

Introduction (Idea)

- LSTM can remember important information over a long period of time and is useful in overcoming the vanishing gradient problem associated with traditional RNNs.

- The implementation of a text classification model using LSTM and Keras involves:
    - Generating a vocabulary and top percentile list
    - One hot encoding classes
    - Building a model using Sequential, Embedding, SpatialDropout1D, LSTM, and Dense layers
    - Training the model using fit() function
    - Evaluating the model using evaluate() function.

Methodology



Data Collection

Collection of large amount of shlokas as data to train and test/validation compilations

Text Preprocessing

Removal of unwanted characters or symbol, input handling such as tokenization, one hot encoding of target.

Embedding

After tokenisation, text data is represented in a numerical format using word embeddings. Word embeddings are vector representations of words that capture the meaning and context of words in a language. Pre-trained word embeddings like Word2Vec is done using Kera’s tokeniser.

LSTM based Model

A LSTM-based model is trained on the preprocessed and embedded data to classify the Sanskrit Shlokas into different categories. The model was trained using a suitable optimizer and loss function.

Evaluation

The final step is to evaluate the performance of the model on the validation set.

Methodology

- It first reads a CSV file containing training data, preprocesses the text data, and builds a neural network model using the Keras Sequential API.

- It then trains the model and evaluates it on a separate test dataset.

- Some of the main libraries used in this code include pandas, numpy, wordcloud, matplotlib, sklearn, keras, and tensorflow.

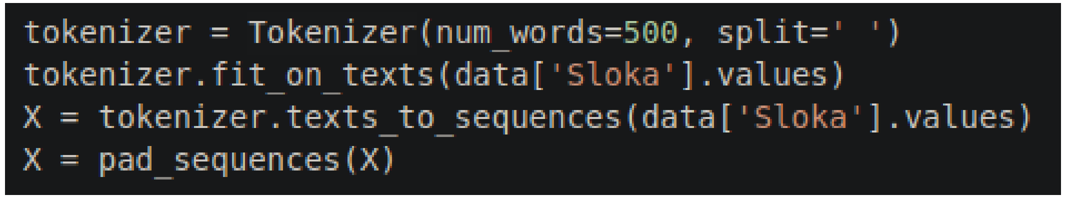
- The text data is first tokenized and converted to sequences using the Token.

Pre-processing

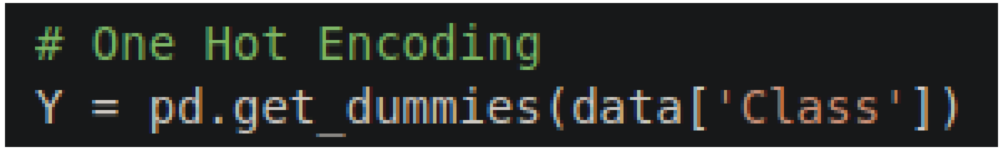
- The first step in text classification is preprocessing the data.

- In this project, we tokenize the text using the Keras tokenizer and pad the sequences to a fixed length.

- We also performed one hot encoding on the target variable.



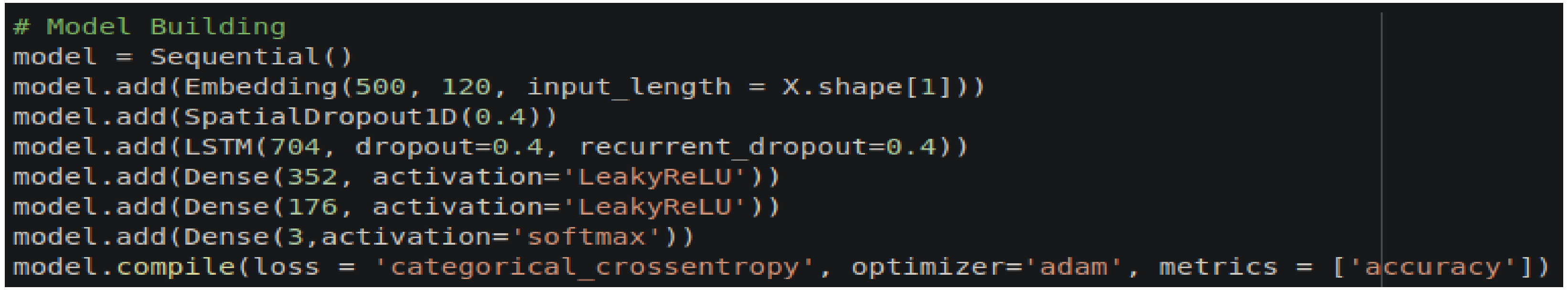
```
tokenizer = Tokenizer(num_words=500, split=' ')
tokenizer.fit_on_texts(data['Sloka'].values)
X = tokenizer.texts_to_sequences(data['Sloka'].values)
X = pad_sequences(X)
```



```
# One Hot Encoding
Y = pd.get_dummies(data['Class'])
```

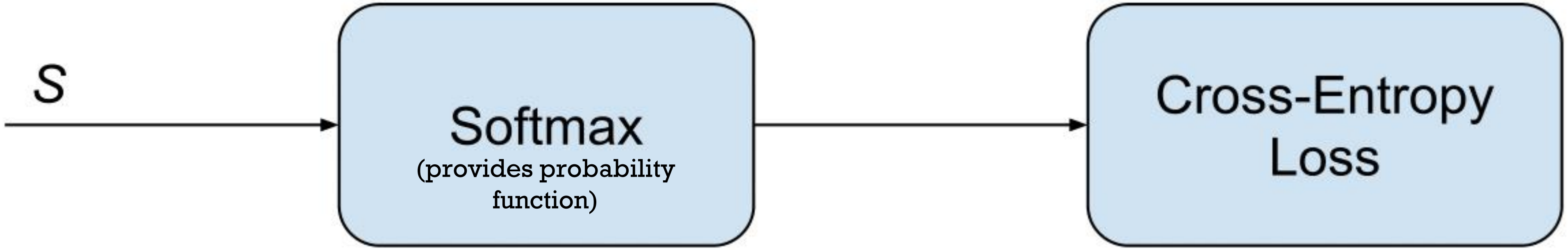
Modelling

- The LSTM model includes 4 layers: Embedding, SpatialDropout1D, LSTM, and Dense.
  - The Embedding layer maps the words in the text to a dense vector space
  - The SpatialDropout1D layer randomly drops out input units to prevent overfitting
  - The LSTM layer processes the sequence data and generates a fixed-length output
  - The Dense layers are used for classification



```
# Model Building
model = Sequential()
model.add(Embedding(500, 120, input_length = X.shape[1]))
model.add(SpatialDropout1D(0.4))
model.add(LSTM(704, dropout=0.4, recurrent_dropout=0.4))
model.add(Dense(352, activation='LeakyReLU'))
model.add(Dense(176, activation='LeakyReLU'))
model.add(Dense(3,activation='softmax'))
model.compile(loss = 'categorical_crossentropy', optimizer='adam', metrics = ['accuracy'])
```

Methodology



s

Softmax
(provides probability function)

Cross-Entropy Loss

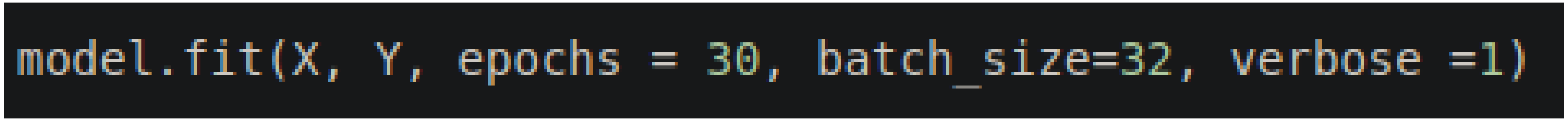
$$f(s)_i = \frac{e^{s_i}}{\sum_j^C e^{s_j}} \qquad CE = -\sum_i^C t_i log(f(s)_i)$$

Training

- The model is trained on preprocessed data for 30 epochs with a batch size of 32.

- The loss function used is categorical_crossentropy and the optimizer used is Adam.

- The model achieves an accuracy of 78.77% on the test data of “anya category; however it has shown gradual decrease in accuracy when other type of scriptures were included…



```
model.fit(X, Y, epochs = 30, batch_size=32, verbose =1)
```

Conclusion

- The presentation discusses the implementation of a shloka-text classification model using LSTM and Keras. (Some of the main libraries used in this code include pandas, numpy, wordcloud, matplotlib, sklearn, keras, and tensorflow.)

- The model achieves good accuracy on the test data; however accuracy does not show effective result with increase in categories.

- The model we developed can be used for various NLP applications like sentiment analysis of shlokas, topic based classification as per supervised label, etc.