Paper Name	Existing Work	Future Work	Dataset and Tagset
Fine-grained part-of-speech tagging in Nepali text (2021)	**1.** Experiment with 3 DL models (***BiLSTM, BiGRU, and BiLSTM-CRF***) for fine-grain POS tagging for the Nepali language. (*Result showed that DL models could capture fine-grained morphological features like gender, person, number, and honorifics that are encoded within words in highly inflectional languages like Nepali with a large enough dataset.* *BiLSTM-CRF with the Bare embedding performed the best and achieved a new state-of-the-art F1 score of 98.51% for fine-grained Nepali POS tagging*) **2.** Experiment with randomly initialized Bare embeddings and compared its results to the new mBERT embeddings for Nepali POS tagging. (*Result showed that training a much smaller randomly initialized embedding can be more useful for* ***fine*** *and* ***coarse-grained*** *tagging in Nepali*)	**1.** While our models achieved great results on fine-grained tags, there are a few tags that the models fail to learn accurately. *So, future work will be to study these errors and employ techniques to mitigate* **2.** The models performed better for the randomly initialized and trained Bare embedding than for the pre-trained mBERT embedding. While mBERT embedding has been proven to be great for cross-lingual generalization, our results show for a more downstream task like POS tagging for Nepali, training a randomly initialized Bare embedding gives better results. This indeed gives a strong motivation to further *explore the mBERT vocabulary and embedding specific to Nepali language.*	It used the Nepali National Corpus (NNC) for all of its experiments. NNC is the largest Nepali POS-tagged corpus with over 17 million manually and semi-manually words tagged with 112 NELRALEC tags.
Nepali POS Tagging using Deep Learning Approaches (2019)	Implementing and comparing different deep learning based POS tagger for Nepali. (*The accuracy obtained for simple* ***RNN, LSTM, GRU and Bidirectional LSTM*** *was 96.84%, 96.48%, 96.86% and 97.27% respectively. Therefore, Bi-directional LSTM outperformed all other three variants of RNN*)	**1.** The vocabulary size and tag set can be increased to increase the efficiency. **2.** Similarly, reinforcement learning can be added for efficient training.	Data were collected from Madan Puraskar Pustakalaya. It consists of Nepali English parallel corpus annotated with 43 POS tag developed and contains nearly 88000 words. The design of this Nepali POS Tag-set was inspired by the PENN Treebank POS Tag-set.

A Deep Learning Approach for Part-of-Speech Tagging in Nepali Language (2018)	A deep learning based POS tagger for Nepali text is proposed which is built using RNN, LSTM, GRU and their bidirectional variants. (*Bi-directional versions of RNN, LSTM and GRU achieved the maximum performance scores with binary cross entropy as the loss function*. *The accuracy of the system also increases with the increase in the size of word embedding vector*.)		The dataset used for this research is POS Tagged Nepali Corpus, 100720 English words and 4325 English sentences were translated from PENN Treebank corpus, which is available through Linguistic Data Consortium (LDC). The dataset consists of 43 tags.
Deep Learning based Tamil Parts of Speech (POS) Tagger (2021)	Train and evaluate using various deep learning approaches such as RNN, LSTM, Bi-LSTM and GRU. All the models were trained with 4, 16, 32 and 64 hidden states, and the number of epochs taken was 10. The learning rate was fixed as 0.01. The loss function used was cross-entropy, the optimizer was Adam, the activation function was softmax and batch size was chosen as 128. *(Bi-LSTM with 64 hidden states yielded the best accuracy (94%) and F1-Score at the word level out of all the models.)*	**1.** Corpus size could be increased, and tags can be further morphologically analyzed **2.** Attention-based transformer architecture could be implemented in future	tag set of 32 tags and 225 000 tagged Tamil words was utilized for training
Deep Neural Network Architecture for Part-of-Speech Tagging for Turkish Language (2018)	In this study, we trained and evaluated the Recurrent Neural Network (RNN) and Long-Short Term Memory (LSTM). (*The experiment indicates that LSTM outperforms RNN with 88.7% F1-score. In addition, while LSTM has 89.0% accuracy score, RNN has 79.7%.*)	In the future, other NNLM models such as CNN, GRU, etc. can be taken into the evaluation for POS tagging for Turkish.	The IMST Universal Dependencies (IMST-UD) Treebank, there are 48000 words and 14 POS tags.
Improving part-of-speech tagging in Amharic language using deep neural network (2023)	Performed a comparison between CRF and DNN (LSTM, BiLSTM and CNN- BiLSTM-CRF) models. (*best performance obtained by an end-to-end deep neural network model, CNN + BiLSTM + CRF, is 97.23% accuracy*)	Use a transfer learning approach to reduce errors in automated tagging.	Dataset contained 321 K tokens and manually tagged with 31 POS tags.

Parts-of-Speech tagging for Malayalam using deep learning techniques (2020)	Experimented with different deep learning models such as LSTM, GRU and Bi-LSTM. The experiment were conducted with 4, 16, 32 and 64 hidden layers, started with the number of iterations (epochs) as 30 and later increased the same to 50 and the last set of experiments used 100 epochs. For the models specified here, They have used the hidden layer size as 32 and the activation function is set as 'tanh'. A dropout parameter is also used after many trial and error mechanisms to improve the training and the network used a learning parameter of 0.01. (*It was claimed that the Bi-LSTM model with 64 hidden layers achieved an f-measure of 98%.*)	As future work, the authors would like to explore the possibility of creating more tagged datasets for further research in Malayalam computing areas.	dataset with a total of 287588 tagged words and the tag set of 36 tags from the Bureau of Indian Standard (BIS)

Section2: Coding wise implementation

Corpus Pre-processing
(*Bring all the annotated corpus in right format*)

⇩

Word Embedding
(*Converting text documents into numerical vectors using Word2Vec*)

⇩

Splitting Dataset into training and testing

⇩

Build and Use RNN, LSTM and BiLSTM. And mBERT
(*With activation function = sigmoid / ReLU, hidden size= 32/64 and drop out = 0.3*)

⇩

Fit the model to the data (learning patterns)

⇩

Making prediction with the model (using patterns)

⇩

Evaluate the model predictions

Hyper Parameters

Hyper-Parameters	Multiclass Classification
Input layer shape(in_feature)	Same as number of features (eg, 4 for age, sex, height, weight)
Hidden layers	Problem specific, minimum = 1, maximum = unlimited
Neurons per hidden layer	Problem specific, generally 10 to 512
Output layer shape (out_features)	1 per class (eg 3 for food, person, dog)
Hidden layer activation	Usually ReLU (rectified linear unit) but can be many others
Output activation	Softmax (torch.softmax in Pytorch)
Loss function	Cross entropy (torch.nn.CrossEntropyLoss in Pytorch)
Optimizer	SGD (stochastic gradient descent) or Adam etc.
Epoch (loop)	50
Batch size (number of neurons)	128
Dropout	0.3