

Practical No:10

Title:

Implementation of LSTM-based Part-of-Speech (POS) Tagging using Keras.

Aim:

To design, train, and evaluate a Long Short-Term Memory (LSTM) based Recurrent Neural Network (RNN) for Part-of-Speech (POS) tagging using a dataset of tokenized sentences.

Prerequisites:

- Basic knowledge of Natural Language Processing (NLP) concepts such as tokens, tags, and embeddings.
- Understanding of Recurrent Neural Networks (RNNs) and Long Short-Term Memory (LSTM) units.
- Familiarity with Python, NumPy, TensorFlow/Keras, and scikit-learn.
- A preprocessed dataset containing words and their corresponding POS tags (e.g. Universal Dependencies or NLTK Treebank).

Theory:

1. Introduction to POS Tagging:

Part-of-Speech (POS) tagging is the process of assigning grammatical tags (such as noun, verb, adjective, etc.) to each

word in a sentence.

Example:

“The cat sat on the mat.”

→ [DET, NOUN, VERB, ADP, DET, NOUN]

POS tagging is a fundamental NLP task that supports downstream applications such as text parsing, named entity recognition, and sentiment analysis.

2. LSTM for Sequential Learning:

An LSTM (Long Short-Term Memory) network is an advanced type of Recurrent Neural Network (RNN) capable of learning long-term dependencies in sequential data.

Unlike vanilla RNNs, LSTMs use cell states and gating mechanisms (input, forget, and output gates) to control the flow of information and prevent vanishing gradients.

3. Why LSTM for POS Tagging:

Language is inherently sequential – the meaning and tag of a word often depend on its preceding and succeeding words.

LSTMs are ideal because they:

- Preserve context over long sequences.
- Handle variable-length sentences (through padding and masking).
- Work effectively with word embeddings like Word2Vec or GloVe.

4. Model Overview:

A typical architecture for LSTM-based POS tagging includes:

- Embedding Layer: Converts word indices into dense vectors.
- Bidirectional LSTM Layer: Captures context from both directions.
- Dense (Softmax) Layer: Outputs probability distribution across POS tags.

5. Sample Model Flow (Keras):

```
model = Sequential()
model.add(Embedding(input_dim=vocab_size, output_dim=128, input_length=max_len))
model.add(Bidirectional(LSTM(64, return_sequences=True)))
model.add(TimeDistributed(Dense(num_tags, activation='softmax'))))
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
```

6. Training and Evaluation:

The model is trained on sequences of word indices with corresponding tag sequences.

Evaluation metrics: Accuracy or F1-Score computed on the validation dataset.

Conclusion:

LSTM-based POS Tagging demonstrates the power of deep learning in understanding linguistic structures.

The model effectively captures both syntactic and contextual relationships between words, outperforming traditional statistical taggers like HMMs or CRFs.

This experiment reinforces the application of Recurrent Neural Networks in NLP, emphasizing the significance of long-term dependency modeling in natural language understanding tasks.

