Building Hindi NER for a simple Weather Bot

For building NER 2 question arises: - Which framework to use? and Which type of embedding?

Good Framework should have :-

- Output should be fast
- Good in recognizing entities
- Need less training data

Comparison between some of the framework: -

NLTK :-

Traditional NLP approaches

text is tokenized > the tokens are passed through a Part Of Speech (POS) tagger > a parser chunks the tokens based on their POS tags to find named entities

Stanford NLP :-

This one uses Conditional Random Field (CRF) on top of NLTK approach

This was State of the art approach prior to deep learning.

Spacy :-

Spacy's NER model is a simple classifier (e.g. a shallow feedforward neural network with a single hidden layer) that is made powerful using some clever feature engineering. Feature engineering with embeddings gives it unique representation for each distinct context it is in.

Recently SpaCy BERT is available but in developing phase.

- Fast, industrial strength , better for less complex NER

Polygot :-

it uses huge unlabeled datasets (like Wikipedia) with automatically inferred entity labels (via features such as hyperlinks).

- Fast , no need for human annotated data, robust, multilingual

- cannot be customized

Deep Pavlov :-

It uses bi-directional LSTM on top of word and character-level embedding layers. However, it combines an additional Conditional Random Fields (CRF) layer to the output of the model.

Basically Hybrid Bi-LSTM-CRF model. – for chatbots enabled with SOAT models

AllenNLP:-

fine-grained NER model uses a bi-LSTM-CRF model (like DeepPavlov) but also includes a pre-trained bi-lstm network – Elmo for better word embedding layer.

- Best for more complex NER
- slow

Embeddings :-

Embeddings is the essential part of the of any NLP task. As same words in different context have different meaning and this can change whole meaning of the sentence.

There are many recent advancements in the Embeddings how to better represent. We will go through some of them.

Word2Vec and GloVe :-

Word2vec and Glove word embeddings are context independent- these models output just one vector (embedding) for each word, combining all the different senses of the word into one vector.

FastText :-

FastText is an extension of word2vec model, treats each word as composed of character ngrams. So the vector for a word is made of the sum of this character n grams. This allow it to capture the meaning of shorter words and allows the embeddings to understand suffixes and prefixes.

ELMo:-

Elmo is a character based model using character convolutions and can handle out of vocabulary words. It is context dependent, take word order into account and need fine tuning for better results.

BERT :-

BERT represents input as subwords and learns embeddings for subwords.

Representing input as subwords as opposed to words because it strikes a balance between character based and word based representations - the most important benefit being avoidance of OOV (out of vocabulary) cases which the other two models *(Glove, Word2vec).*

I have use SpaCy and FastText embeddings for Hindi NER keeping fast response and small dataset and low complexity and also as I was already familiar with the SpaCy framework.

Refer Github link for code and model testing.