EX.NO.: 21
DATE: 28.03.2025

NER, POS AND HMM

To implement Named Entity Recognition (NER) using Hugging Face Transformers and Part-of-Speech (POS) tagging using both the NLTK library and a Hidden Markov Model (HMM) POS tagger trained on the Brown corpus.

PROCEDURE:

1. Install and import libraries
2. Load pre-trained BERT-based NER model using Huggingface transformers
3. Tokenize given sentence and apply NLTK's built-in POS tagging to label each word with POS
4. Load POS tagged sentences from Brown corpus with universal POS tags and do HMM based POS tagger training
5. Predict POS tags using the trained HMM based POS tagger

CODE AND OUTPUT

```
import nltk
from nltk.corpus import brown
from nltk.tag import hmm
from nltk.probability import LidstoneProbDist
from sklearn.model_selection import train_test_split
from transformers import pipeline

# Ensure necessary NLTK datasets are downloaded
nltk.download('punkt')
nltk.download('averaged_perceptron_tagger')
nltk.download('brown')
nltk.download('universal_tagset')
nltk.download('maxent_ne_chunker')
nltk.download('words')

# 1. Named Entity Recognition (NER) using Hugging Face Transformers
def perform_ner(sentence):
    ner_pipeline = pipeline("ner",
model="dbmdz/bert-large-cased-finetuned-conll03-english")
    ner_results = ner_pipeline(sentence)
    print("Named Entities (Hugging Face):")
    for entity in ner_results:
        print(f"{entity['word']} -> {entity['entity']}")

# 2. POS tagging using NLTK
def pos_tag_nltk(sentence):
    words = nltk.word_tokenize(sentence)
    pos_tags = nltk.pos_tag(words)
    print("POS Tags:")
    print(pos_tags)

# 3. HMM-based POS tagger
def train_hmm_pos_tagger():
```

```
    tagged_sentences = brown.tagged_sents(tagset='universal')
    train_data, test_data = train_test_split(tagged_sentences, test_size=0.2,
random_state=42)

    trainer = hmm.HiddenMarkovModelTrainer()
    model = trainer.train(train_data, estimator=lambda fd, bins: LidstoneProbDist(fd,
0.1, bins))

    return model, test_data

def predict_hmm(model, sentence):
    words = nltk.word_tokenize(sentence)
    predicted_tags = model.tag(words)
    print("HMM POS Tags:")
    print(predicted_tags)

if __name__ == "__main__":
    sentence = "Elon Musk founded SpaceX in 2002 and Tesla Motors in 2003."

    # Perform NER
    perform_ner(sentence)

    # Perform POS tagging using NLTK
    pos_tag_nltk(sentence)

    # Train HMM POS Tagger
    hmm_model, test_data = train_hmm_pos_tagger()

    # Predict POS tags using HMM POS Tagger
    predict_hmm(hmm_model, sentence)
```

```
Named Entities (Hugging Face):
El -> I-PER
##on -> I-PER
Mu -> I-PER
##sk -> I-PER
Space -> I-ORG
##X -> I-ORG
Te -> I-ORG
##sla -> I-ORG
Motors -> I-ORG
POS Tags:
[('Elon', 'NNP'), ('Musk', 'NNP'), ('founded', 'VBD'), ('SpaceX', 'NNP'), ('in', 'IN'), ('2002', 'CD'), ('and', 'CC'), ('Tesla', 'NNP'), ('Moto
HMM POS Tags:
[('Elon', 'X'), ('Musk', 'X'), ('founded', 'X'), ('SpaceX', 'X'), ('in', 'ADP'), ('2002', 'NUM'), ('and', 'CONJ'), ('Tesla', 'ADJ'), ('Motors',
```

INFERENCE

The NER model effectively extracts named entities like persons and organizations, while the HMM-based POS tagger provides probabilistic tagging that adapts well to unseen data, making it more robust than rule-based methods.