Ex 1b: Hidden Markov Model (HMM) based Predictive Text System

Learning Objective:

To implement a predictive text system using Hidden Markov Model (HMM), enabling students to understand contextual probability, POS transitions, and next-word prediction using the Brown Corpus.

Steps:

1. Initialize the environment by installing and importing the **Natural Language Toolkit** (nltk). Download the **Brown Corpus** (for text data) and **the Universal Tagset** (for simplified Part-of-Speech tags).
2. Extract raw sentences from the Brown Corpus using brown.sents().
3. **Tokenization**: Convert all words to lowercase and flatten the nested list into a single list of tokens to verify the total volume of training data.
4. Load the tagged version of the corpus (brown.tagged_sents) which provides the 'Hidden States' (POS tags) linked to the 'Observations' (words).
5. Train the HMM Model. Use the HiddenMarkovModelTrainer to perform Supervised Learning. Calculate the internal parameters of the HMM: Initial State Probabilities, Transition Probabilities, Emission Probabilities
6. Build POS Transition Probabilities
7. Build Emission Probabilities
8. Define Prediction Function
9. Test the Model
10. Verify POS prediction on certain words.

Program:

```
import nltk
from nltk.corpus import brown
from nltk.tag import HiddenMarkovModelTrainer
from collections import defaultdict, Counter

nltk.download('brown')
nltk.download('averaged_perceptron_tagger_eng')

sentences = brown.sents()[:3000]
tagged_sents = [nltk.pos_tag(sent) for sent in sentences]

trainer = HiddenMarkovModelTrainer()
hmm_model = trainer.train(tagged_sents)

emission_counts = defaultdict(Counter)
for sent in tagged_sents:
    for word, tag in sent:
        emission_counts[tag][word.lower()] += 1

emission_prob = {}
for tag, counter in emission_counts.items():
    total = sum(counter.values())
    emission_prob[tag] = {w: c / total for w, c in counter.items()}

def predict_next_word_with_probability(sentence):
    tagged_input = nltk.pos_tag(sentence)
    last_pos = tagged_input[-1][1]

        next_pos = hmm_model._transitions[last_pos].max()
    pos_probability = hmm_model._transitions[last_pos].prob(next_pos)
```

```
    word, word_probability = max(
        emission_prob[next_pos].items(),
        key=lambda x: x[1]
    )


    final_probability = pos_probability * word_probability
    return tagged_input, next_pos, word, final_probability

user_input = input("Enter a sentence: ").lower().split()
pos_tags, next_pos, next_word, prob = predict_next_word_with_probability(user_input)

print("\nPOS Tags:")
for w, t in pos_tags:
    print(w, "→", t)

print("\nPredicted Next POS:", next_pos)
print("Predicted Next Word:", next_word)
print("Prediction Probability:", prob)
```

Output:

```
[nltk_data] Downloading package brown to /root/nltk_data...
[nltk_data]   Unzipping corpora/brown.zip.
[nltk_data] Downloading package universal_tagset to /root/nltk_data...
[nltk_data]   Unzipping taggers/universal_tagset.zip.
```

True

Total sentences: 57340

['The', 'Fulton', 'County', 'Grand', 'Jury', 'said', 'Friday', 'an', 'investigation', 'of', "Atlanta's", 'recent', 'primary', 'election', 'produced', '``', 'no', 'evidence', "''", 'that', 'any', 'irregularities', 'took', 'place', '.']

Total tokens: 1161192

['the', 'fulton', 'county', 'grand', 'jury', 'said', 'friday', 'an', 'investigation', 'of', "atlanta's", 'recent', 'primary', 'election', 'produced', '``', 'no', 'evidence', "''", 'that']

[('The', 'DET'), ('Fulton', 'NOUN'), ('County', 'NOUN'), ('Grand', 'ADJ'), ('Jury', 'NOUN'), ('said', 'VERB'), ('Friday', 'NOUN'), ('an', 'DET'), ('investigation', 'NOUN'), ('of', 'ADP'), ("Atlanta's", 'NOUN'), ('recent', 'ADJ'), ('primary', 'NOUN'), ('election', 'NOUN'), ('produced', 'VERB'), ('``', '.'), ('no', 'DET'), ('evidence', 'NOUN'), ("''", '.'), ('that', 'ADP'), ('any', 'DET'), ('irregularities', 'NOUN'), ('took', 'VERB'), ('place', 'NOUN'), ('.', '.')]

Tagged sentence:

[('in', 'ADP'), ('the', 'DET')]

Top predictions:

, POS=. prob=0.00505
other POS=ADJ prob=0.00487
new POS=ADJ prob=0.00468
. POS=. prob=0.00427
time POS=NOUN prob=0.00363

Tagged sentence:

[('the', 'DET'), ('president', 'NOUN')]

Top predictions:

, POS=. prob=0.11253

. POS=. prob=0.09519

of POS=ADP prob=0.06165

and POS=CONJ prob=0.04529

in POS=ADP prob=0.03533

Learning Outcome:

Upon completion of this experiment, the bigram and trigram probability distributions for next-word prediction was implemented and evaluated using the Brown Corpus.