Part-of-Speech Tagging using Hidden Markov Model
Jiechen Li & Yulei Xia

1. Model Components Generation:
Data Source: The Brown corpus from the **nltk** library was utilized, focusing on the first 10,000 tagged sentences using the universal tagset.
Components:

- **Initial State Distribution (initiate_state function)**: This distribution offers insight into the likelihood of a sentence starting with a specific POS tag, computed by analyzing the tags of the first word across the 10k sentences.
- **Transition Matrix (transition function)**: Represents the likelihood of transitioning from one tag to another, computed by examining consecutive tag pairs in the training sentences.
- **Observation Matrix (emission function)**: Illustrates the likelihood of a word (observation) being generated by a tag (state). It was determined by investigating word-tag pairs in the training sentences.

The model incorporates an "OOV" (Out-of-Vocabulary) observation to manage words not seen during training. Smoothing was also applied across all matrices to ensure non-zero probabilities.

2. Sequence Inference:
The Viterbi algorithm was employed to deduce the most probable sequence of states (tags) for sentences 10150-10152 from the Brown corpus.

3. Comparison:
The inferred sequence was juxtaposed with the actual tags.
Model Prediction: [0, 3, 5, 9, 3, 6, 6, 0, 3, 10, ...]
True Results: [0, 6, 5, 9, 3, 6, 6, 0, 3, 10, ...]
Accuracy: Approximately 93.62%
This quantitative measure provides a clear understanding of the tagger's efficacy.

4. Analysis:
Reasons for the POS tagger's performance:

- **Training Data**: The tagger benefitted from the comprehensive insights provided by a vast dataset of 10k sentences.
- **OOV Handling**: The "OOV" observation ensures that the tagger can effectively handle unfamiliar words.
- **Smoothing**: Smoothing guarantees that the model doesn't assign a zero probability to unseen word-tag pairs or tag transitions, which is crucial for the Viterbi algorithm's optimal functioning.

However, there are instances where the tagger might falter:

- **Ambiguity**: The context is key. Some words can be tied to multiple POS tags based on their context. Without wider context evaluation or more advanced models, these ambiguities can lead to errors.

- **Fixed Probabilities**: The model's fixed probabilities, based on the training data, may not always align with real-world sentence structures or word usage patterns.
- **HMM Limitations**: The HMM model operates on a limited context, considering only the current and previous state. More advanced models, like Conditional Random Fields (CRFs) or neural network-based models, can encompass wider contexts.

In conclusion, the HMM-based POS tagger, with an accuracy of approximately 93.62%, showcases decent performance grounded in its foundational principles and extensive training dataset. However, there's potential for improvement. To further boost accuracy and better handle linguistic nuances, it's worth exploring advanced models and techniques.