AIM-829

Assignment 1: Part-of-Speech Tagging Using Hidden Markov Model (HMM)

Objective

In this assignment, you will implement a Part-of-Speech (PoS) tagging system using a Hidden Markov Model (HMM). You will use the provided training and testing datasets to train and evaluate your model. The datasets contain sentences where each word is paired with its corresponding PoS tag. The goal is to correctly tag words in sentences by leveraging the Viterbi algorithm.

Dataset Details

- **Training Set**: Contains sentences where each word is paired with its corresponding POS tag in the format: **`(word-tag)`**.
- **Test Set**: Contains sentences, also in the same format **`(word-tag)`**. Use this dataset to evaluate your model's performance.

Both datasets are provided in CSV format and attached to this assignment.

Steps to Complete the Assignment

Step 1: Understand the Dataset

1. Load the training and testing sets using Python libraries.
2. Explore the structure of both datasets. Each row represents a word and its corresponding tag, grouped by sentences.
3. Split the data into words and their respective tags for processing.

Step 2: Preprocessing

1. Extract unique words and tags from the training dataset to build your vocabulary.

Step 3: Estimate HMM Parameters using the training set

1. **Emission Probabilities**:
   - Compute $P(word|tag)$: the probability of a word being generated by a tag.
   - Formula: $P(word|tag) = \frac{\text{Count(word, tag)}}{\text{Count(tag)}}$.
2. **Transition Probabilities**:
   - Compute $P(tag_{t+1}|tag_t)$: the probability of transitioning from one tag to another.
   - Formula: $P(tag_{t+1}|tag_t) = \frac{\text{Count(tag}_{t+1} \text{ given tag}_t)}{\text{Count(tag}_t)}$.
3. **Initial Probabilities**:
   - Compute $P(tag_{start})$: the probability of a tag starting a sentence.
   - Formula: $P(tag_{start}) = \frac{\text{Count(tag}_{start})}{\text{Total sentences}}$.

Step 4: Implement the Viterbi Algorithm

1. **Goal**: Use the Viterbi algorithm to determine the most probable sequence of PoS tags for a given sentence in the testing set
2. **Steps**:
   - Initialize the Viterbi matrix and backpointer matrix for the sentence.
   - Populate the Viterbi matrix for each word in the sentence using the formula:
     $$V_t(tag) = \max_{prev\_tag} [V_{t-1}(prev\_tag) \cdot P(tag|prev\_tag) \cdot P(word|tag)]$$
   - Track the most probable previous tag for each current tag using the backpointer matrix.
   - At the end of the sentence, use the backpointer matrix to trace back and extract the best sequence of tags.

Summary: Train and Validate Your Model

1. Use the training data to compute the HMM parameters (emission, transition, and initial probabilities).
2. Use your Viterbi implementation to predict POS tags for each sentence in the testing dataset.
3. Compare your predicted tags with the actual tags in the testing set.

Step 6: Evaluate Your Model

1. Calculate accuracy using testing set.

2. Optionally, generate a confusion matrix to analyze the model's performance for different tags.

Rules and Guidelines

1. All submitted work must be your own. Plagiarism will result in a score of zero for the assignment
2. Code generated by AI tools is strictly prohibited.
3. Submissions that are identified as AI-generated will receive a score of zero**.**

Deliverables (each group) in a zip folder (named as [Group No.].zip)

4. A Python script or Jupyter Notebook implementing the above steps.
5. Accuracy and evaluation results on the validation set.
6. Visualizations (e.g., confusion matrix) if applicable.
7. Brief documentation explaining your implementation, results and contribution of each member.

Files Provided

1. **Training Set**: `train_data.csv`
2. **Validation Set**: `validation_data.csv`

Submission Deadline

14/02/2025, EoD to be submitted in LMS.

Note: Late submissions will not be accepted. If you face any issues, contact TAs before the deadline.