NLP Homework 3

Task 1: Vocabulary Creation

Approach

The vocabulary creation process began with the determination of a frequency threshold for unknown word handling. Words occurring less than the threshold were to be replaced with a special token, **`<unk>`,** to manage the sparsity of the dataset effectively.

The threshold was selected based on a balance between retaining a rich vocabulary and minimizing the noise from rare words. After establishing this threshold, the training data was processed to count the occurrences of each word. This count was used to decide which words would be retained in the vocabulary and which would be replaced by the **`<unk>`** token.

Results

The threshold for unknown words was set at **3** occurrences. Words appearing fewer than three times were considered rare and thus were replaced with **`<unk>`** in the training data.

The processing resulted in a vocabulary comprising **16,920** unique words. Additionally, the special token `<unk>` accounted for a total of **32,536** occurrences. This represents the instances where infrequent words were replaced, adhering to the pre-set threshold.

Conclusion

The selected threshold reflects a strategic decision to ensure that the vocabulary is neither too broad, which could lead to overfitting, nor too narrow, which could result in underfitting. The size of the vocabulary and the number of `<unk>` occurrences suggest a diverse linguistic dataset with a substantial long-tail distribution of word frequencies. The approach balances the trade-off between the granularity of the language model and the manageability of less frequent words.

Task 2: Model Learning

Transition and Emission Probabilities

The calculation of transition and emission probabilities is a central part of training the Hidden Markov Model (HMM). Transition probabilities reflect the likelihood of transitioning from one part-of-speech tag to another, while emission probabilities indicate the likelihood of a particular word being generated from a given tag.

The process included applying Laplace smoothing to ensure that even unobserved transitions and emissions in the training data were assigned a non-zero probability, which is essential for the model to generalize well to unseen data.

Sample Emission Probabilities

To illustrate the emission probabilities, here are a few examples extracted from the model:

- The probability of the tag 'VBP' (non-3rd person singular present verb) emitting the word 'industry' is approximately 0.00003.
- For the tag 'JJ' (adjective), the probability of emitting the word 'closed' is also around 0.00003.
- Similarly, the tag 'NNP' (proper noun, singular) has a probability of emitting the word 'Honda' with a probability of about 0.00003.

These probabilities, while seemingly small, are significantly larger than zero due to smoothing, allowing the model to handle words that may not have been observed with a particular tag in the training set.

Quantitative Analysis

The model consists of transition parameters derived from all possible tag pairs and emission parameters based on the combinations of tags and vocabulary words. With **45** unique tags and **16,920** unique words, the model encompasses **45 * 45** transition probabilities and **45 *16,920** emission probabilities.

The use of Laplace smoothing ensures that each potential tag-word combination has a non-zero probability, allowing the model to account for the possibility of unseen tag-word pairs during training.

Conclusion

The computed transition and emission probabilities form the backbone of the HMM for part-of-speech tagging. They allow the model to make informed predictions about tags based on the context provided by surrounding tags and the words themselves. The inclusion of smoothing techniques in the calculation process is vital for the model's ability to perform well on real-world data, which invariably includes instances beyond the scope of the training dataset.

By establishing these probabilities, we have equipped our model with the foundational parameters necessary for the decoding tasks that follow.

Task 3: Greedy Decoding with HMM

Greedy Decoding Algorithm

The greedy decoding algorithm is a sequential approach to tagging where the best tag for each individual word is chosen locally at each step. For our Hidden Markov Model (HMM), the algorithm iterates through each word in a sentence and calculates the product of transition and emission probabilities for all possible tags. The tag with the highest probability at each position is selected, considering the previously selected tag to maintain context.

When a word is not found in the training data, the algorithm accounts for this by using the **`<unk>`** token probabilities, ensuring that a prediction is made even when encountering unseen words.

Implementation

The implementation involved two key steps:

1. Decoding: For each word in a sentence, the model computes the product of emission and transition probabilities for every tag. It then selects the tag with the highest probability, assigning it as the predicted tag for the given word.

2. Handling Unseen Word: If the word is not in the training vocabulary, the model repeats the process using the probabilities associated with the **`<unk>`** token, which allows for robust predictions in the face of unknown vocabulary.

Results

Upon evaluating the model on the development data using the greedy algorithm, the accuracy was determined to be **91.78%**. This measure reflects the proportion of tags that the model predicted correctly.

Conclusion

The greedy decoding algorithm provides an efficient and straightforward mechanism for part-of-speech tagging with HMM. Its implementation for our model demonstrated a high level of accuracy on the development dataset, indicating a well-trained set of parameters for transition and emission probabilities. While it may not globally optimize the tag sequence, its local decision-making process offers a rapid and effective solution for tagging.

Task 4: Viterbi Decoding with HMM

Viterbi Decoding Algorithm

The Viterbi decoding algorithm offers a more computationally intensive, yet more accurate approach to sequence prediction than greedy decoding. It optimizes the entire sequence of tags for a given sentence rather than making a decision at each step based on local maxima.

The algorithm consists of four main steps:

1. Initialization: It starts by calculating the probabilities of each tag for the first word in the sentence, considering the transition from the start state and the emission probabilities of the word given the tag.

2. Recursion: For each subsequent word, it iterates through the tags and calculates the probabilities by considering the emission probabilities of the word given the tag and the transition probabilities from all possible previous tags. It keeps track of the best previous tag for each current tag.

3. Termination: It identifies the final state with the highest probability after processing the last word in the sentence.

4. Path Backtracking: It then traces back the best path of tags leading to this final state, effectively uncovering the most probable sequence of tags for the entire sentence.

Implementation

The Viterbi decoding process was implemented by following the steps, ensuring to handle unseen words by assigning them the emission probabilities of the `<unk>` token. This approach provides a robust solution for part-of-speech tagging, even when the sentence contains words are not present in the training data.

Results

Upon evaluation of the Viterbi algorithm's performance on the development dataset, the model achieved an accuracy of **92.31%.** This metric underscores the effectiveness of the Viterbi algorithm in identifying the most likely sequence of tags, outperforming the greedy algorithm's accuracy by a small margin.

Conclusion

The Viterbi decoding algorithm proves to be a powerful tool in the context of HMM for part-of-speech tagging. Its ability to consider the sequence allows it to capture the context more effectively than the greedy approach, which is reflected in the higher accuracy achieved on the development data. This global optimization characteristic of the Viterbi algorithm is essential for tasks where context plays a significant role in making accurate predictions.