CL1 Assignment Report

Nikhita Anjani Ravi

25th March 2025

1 Introduction

POS tagging is the process of assigning grammatical categories (e.g., noun, verb, adjective) to words in a given text. Two commonly-used approaches to POS tagging are HMMs (Hidden Markov Models) and CRFs (Conditional Random Fields).

2 HMMs

HMM is a generative probabilistic model based on the Markov assumption, which states that the probability of a state (tag) depends only on the previous state. It models the joint probability of sequences of words and tags. The basic principles behind HMMs are:
Hidden States: These are the hidden, unobservable factors that drive the system's evolution.
Markov Assumption: The HMM assumes that the current state depends only on the previous state, not on the entire history.
Observations: These are the visible outputs generated by the hidden states (Here, they are words).
Probabilistic Model: They use probabilities to find the relationships between the hidden states and the observations.
HMM consists of:

- **States**: POS tags.
- **Observations**: Words in the sentence.
- **Transition Probabilities**: $P(t_i|t_{i-1})$. The transition probability is the likelihood of a particular sequence to be followed, for example, how likely is that a noun is followed by a verb. This probability is known

as Transition probability. In short, it is the probability of a tag given the previous tag.

- **Emission Probabilities**: $P(w_i|t_i)$. It is the probability of a word given a tag.
- **Initial Probability**: $P(t_1)$. It is the probability of a sentence starting with a particular tag (e.g Proper nouns or common nouns often start a sentence.)

The best sequence of tags is found using the Viterbi algorithm, which efficiently finds the most probable tag sequence given a sentence.
Strengths:

1. **Efficient Inference:** The Viterbi algorithm allows fast and efficient sequence decoding.
2. **Works Well with Small Data:** Performs well even with limited training data due to its probabilistic nature.
3. **Probabilistic Interpretation:** Provides confidence scores for predictions, making it useful in uncertain environments.
4. **Handles Hidden States:** Well-suited for tasks involving hidden/latent structures, such as speech recognition and part-of-speech tagging.
5. **Simple and Computationally Efficient:** Requires fewer parameters compared to CRF, leading to faster training and inference.

Weaknesses:

1. **Strong Independence Assumptions:** Assumes observations depend only on the current state, limiting its ability to capture long-range dependencies.
2. **Label Bias Problem:** Suffers from bias towards states with fewer outgoing transitions.
3. **Limited Feature Representation:** Cannot incorporate rich linguistic features like prefixes, suffixes, or word shape.
4. **Struggles with Complex Dependencies:** Performs poorly in tasks requiring long-range contextual understanding.
5. **Handling of Unseen Words:** Relies on emission probabilities, making it less effective for handling unseen words without smoothing techniques.

3 CRFs

CRF is a discriminative probabilistic model that directly models the conditional probability $P(T|W)$, where T is the sequence of POS tags and W is the sequence of words. Unlike HMMs, which model the joint probability $P(T, W)$ and make independence assumptions, CRFs do not assume that observations (words) are conditionally independent given the tags. This allows them to incorporate more features and account for complexities in sentences, thus being more useful and flexible as compared to HMMs. By considering the entire sequence at once, rather than predicting each tag independently, CRFs avoid the label bias problem found in HMMs and other generative models. However, they are computationally more expensive and require more data for effective training.
The fundamental principles of CRFs are:

- **Feature-based representation** using arbitrary features.
- **Global optimization** over entire sequences.
- **Log-linear model**:

$$P(T|W) = \frac{\exp(\sum_k \lambda_k f_k(T, W))}{Z(W)} \tag{1}$$

  where $f_k(T, W)$ are feature functions, λ_k are learned weights, and $Z(W)$ is the normalization factor.

Inference is performed using the **Viterbi or Forward-Backward algorithm**.

Strengths:

1. **No Independence Assumption:** Unlike HMM, CRF can model long-range dependencies and interactions between features.
2. **Higher Accuracy:** Achieves better performance in structured prediction tasks like POS tagging and named entity recognition.
3. **Richer Feature Representation:** Can incorporate arbitrary overlapping features such as word shape, capitalization, and surrounding words.
4. **Avoids Label Bias Problem:** Normalizes probabilities globally across the entire sequence rather than locally.

5. **More Flexible for Complex Structures:** Suitable for real-world NLP applications where multiple contextual cues contribute to decisions.

Weaknesses:

1. **Computationally Expensive:** Requires iterative optimization techniques like gradient descent, making training slower.
2. **Requires Large Training Data:** Needs more labeled data for effective generalization compared to HMM.
3. **Inference Complexity:** Inference is slower compared to HMM due to the need for belief propagation or gradient-based methods.
4. **Difficult to Implement and Tune:** Requires careful feature selection and regularization to prevent overfitting.

4 Difference between HMMs and CRFs

Feature	**Hidden Markov Model (HMM)**	**Conditional Random Field (CRF)**
Type of Model	Generative Model	Discriminative Model
Probability Modeled	Models the joint probability $P(T, W)$	Models the conditional probability $P(T\|W)$
Assumptions	- Assumes Markov property (current state depends only on the previous state). - Assumes emission independence (observed word depends only on the current state).	- No strict independence assumptions. - Can model long-range dependencies.
Feature Representation	- Uses only emission and transition probabilities. - Cannot incorporate rich linguistic features.	- Can incorporate arbitrary features (e.g., word shape, suffixes, context).
Inference Algorithm	Viterbi algorithm (Dynamic Programming)	Belief Propagation or Gradient-based optimization
Computational Complexity	- Computationally efficient. - Works well with small datasets.	- Computationally expensive. - Requires large datasets.
Suitability for POS Tagging	- Decent accuracy but limited by independence assumptions.	- Higher accuracy due to richer features, but more computationally expensive.

Table 1: Comparison of HMM and CRF for POS Tagging

5 Difference between Probabilistic and Discriminative Models

Feature	**Probabilistic (Generative) Models**	**Discriminative Models**
Definition	Models the **joint probability** $P(X, Y)$, learning the data distribution and label generation process.	Models the **conditional probability** $P(Y \mid X)$, focusing only on decision boundaries.
How Classification Works	Uses **Bayes' Theorem**: $P(Y \mid X) = \frac{P(X\|Y)P(Y)}{P(X)}$ to infer the most probable label.	Directly estimates $P(Y \mid X)$ from training data without needing the full distribution.
Example Models	Naïve Bayes, Hidden Markov Models (HMM), Gaussian Mixture Models (GMM), Latent Dirichlet Allocation (LDA).	Logistic Regression, Support Vector Machines (SVM), Conditional Random Fields (CRF), Neural Networks (e.g., Transformers, BERT).
Feature Dependency Assumption	Often assumes **independence** between features given the label (e.g., Naïve Bayes).	No strict independence assumption, can capture complex feature interactions.
Performance on Small Data	Works well with **small datasets** since it learns the underlying data distribution.	Requires **more training data** to generalize well.
Computational Complexity	Often computationally **efficient** due to simple probability estimations.	Computationally **expensive** due to complex optimizations.
Handling of Missing Data	Can handle **missing data** well by modeling the entire distribution.	Struggles with missing data.
Flexibility	Limited in capturing rich **contextual** and **sequential** dependencies.	More flexible in learning complex decision boundaries.
Application Domains	Speech recognition, Bayesian modeling, anomaly detection.	POS tagging, object recognition, deep learning applications.
Accuracy in Classification	Usually lower due to independence assumptions.	Higher accuracy because it focuses only on classification.

Table 2: Comparison of Probabilistic (Generative) and Discriminative Models

6 Observations and Results

6.1 HMMs

6.1.1 Results

English
Micro-average metrics:
Precision: 0.8500
Recall: 0.8500
F1: 0.8500

Macro-average metrics:
Precision: 0.7267
Recall: 0.8066
F1: 0.7396

Weighted-average metrics:
Precision: 0.8777
Recall: 0.8500
F1: 0.8564

Accuracy: 0.8500

Hindi
Micro-average metrics:
Precision: 0.7523
Recall: 0.7523
F1: 0.7523

Macro-average metrics:
Precision: 0.6697
Recall: 0.7715
F1: 0.6500

Weighted-average metrics:
Precision: 0.8560
Recall: 0.7523
F1: 0.7674

Accuracy: 0.7523

6.1.2 Errors Observed

Hindi

Confusion between **NN (Noun) and NNP (Proper Noun)** were found often. The model misclassified 19 instances of proper nouns (NNP) as common nouns (NN), and considered 31 common nouns to be proper nouns. This likely happens because proper nouns often share morphological features with common nouns. Since HMM relies on emission probabilities, it struggles when a word can belong to both categories. If a proper noun (e.g., "Raj" as a name) appears frequently as a common noun (e.g., Raj meaning "rule" or "kingdom"), the HMM may wrongly classify it as NN instead of NNP. This is because HMM does not inherently differentiate between different meanings of a word in different contexts.

VAUX (Auxiliary Verbs) and VVM (Main Verbs) have 4 and 13 misclassifications, respectively. This suggests that auxiliary verbs might be mistaken for main verbs when used independently. The lack of long-range dependencies in HMM means that context is not always effectively captured.

11 instances of **adjectives (JJ) were misclassified as nouns (NN).** This happens because many adjectives in Hindi can be used as nouns depending on context.

HMM does not incorporate complex syntactic structures, leading to these misclassifications.

Words in categories like **RB (Adverb), RP (Particle), and RDPUNC (Punctuation)** had few misclassifications, indicating that the model performs better on frequent categories.

English

Looking at the English HMM confusion matrix shows that the model struggles with some common linguistic ambiguities. One of the biggest issues is the confusion between **NN (common noun)** and **NNP (proper noun)**, where several proper nouns get mistaken for common ones. Another tricky area is **JJ (adjective)** vs. **NNP (proper noun)**, which suggests that adjectives derived from names or capitalized words were sometimes misclassified. Apart from nouns, the model has trouble with **RB (adverb)** and other categories, possibly because some adverbs can function similarly to adjectives or verbs, depending on sentence structure. HMMs only look at limited context due to the Markov assumption. Without deeper sentence-level understanding, they rely heavily on word probabilities, which isn't

always enough to disambiguate categories that depend on broader linguistic cues.

6.1.3 Ways to improve HMM model

1. Instead of using a first-order HMM (which conditions only on the previous tag), a second-order or even third-order HMM can be employed. This allows the model to consider a larger context, reducing ambiguities in tag sequences.

2. Instead of assigning unknown words a uniform probability, we can use morphological analysis (e.g., suffix-based tagging for inflected words) or open-class probability estimations to infer likely POS tags based on observed patterns.

3. Using Semi-Supervised training.

4. Some errors stem from confusion between high-frequency and low-frequency tags (e.g., common vs. proper nouns). Adjusting the transition probabilities dynamically based on corpus frequency distributions can mitigate these issues.

5. Using a larger dataset for training.

6.2 CRFs

6.2.1 Results

Hindi
Micro-average metrics:
Precision: 0.8386
Recall: 0.8386
F1: 0.8386

Macro-average metrics:
Precision: 0.7403
Recall: 0.8571
F1: 0.7411

Weighted-average metrics:
Precision: 0.8922
Recall: 0.8386

F1: 0.8533

Accuracy: 0.8386

English
Micro-average metrics:
Precision: 0.9024
Recall: 0.9024
F1: 0.9024

Macro-average metrics:
Precision: 0.8358
Recall: 0.8665
F1: 0.8255

Weighted-average metrics:
Precision: 0.9243
Recall: 0.9024
F1: 0.9064

Accuracy: 0.9024

6.2.2 Errors Observed

Hindi
The model struggles in distinguishing between common nouns (NN) and proper nouns (NNP), as seen in the 19 instances of misclassification. This confusion likely arises due to the morphological similarities between proper and common nouns in Hindi, where proper nouns often follow the same inflectional patterns as common nouns. Additionally, adjectives (JJ) are sometimes misclassified as nouns (NN), which can be attributed to Hindi's frequent use of noun-adjective compounding, making it difficult for the model to differentiate based solely on context. The misclassification of postpositions (PSP) and punctuation (RDPUNC) suggests that the model may struggle with syntactic boundary recognition, possibly due to the relatively flexible word order in Hindi. While the CRF model generally performs well, these errors indicate the need for better feature engineering, particularly in handling morphologically ambiguous words and context-dependent grammatical roles.

English
The English CRF confusion matrix shows strong performance, but some errors reveal areas for improvement. A notable issue is the confusion between adjectives (JJ) and nouns (NN), likely due to the overlap where adjectives are frequently used as modifiers and sometimes even as standalone nouns. Proper nouns (NNP) and common nouns (NN) also exhibit misclassification. Despite these challenges, the CRF model outperforms HMM by better capturing contextual dependencies.

6.2.3 Ways to Improve CRF model

- Incorporate richer linguistic features such as word suffixes, prefixes, word shapes (e.g., capitalization patterns), and character n-grams to better capture morphological variations.
- Training on larger and more diverse corpora improves generalization and reduces misclassification for low-resource categories. (The English model was trained on 6 datasets of similar length while it was 4 for Hindi, showing the difference)
- Implement rule-based corrections for frequent errors, such as proper noun capitalization or verb-noun disambiguation.

7 Conclusion

Overall, the CRF model has outperformed the HMM model due to the various reasons listed. CRF effectively captures complex linguistic patterns that HMM struggles with. The ability to condition on the entire observation sequence enables CRF to make more informed predictions. However, HMM is not too far behind due to being trained on a large dataset, tested on a relatively smaller one. Both datasets are balanced and effectively capture the frequencies of words and common patterns in English and Hindi.