Unsupervised POS Tagging

- **Goal**: POS-tag an unlabeled corpus given a POS lexicon, subject to the constraints imposed by the lexicon

**Common Approach**

- Train an HMM (i.e., learn its parameters, \( \theta \), which consists of the tag-transition distributions and the output distributions) to maximize the likelihood of the unlabeled corpus using EM
- **Problem**: Tagging accuracy is sensitive to many factors (e.g., parameter initializations)

**Alternative: Goldwater and Griffiths’s (2007) Nonparametric Fully-Bayesian Approach**

- Adopts an HMM as the underlying model as before, but:
  1. integrates over all possible parameter values, rather than committing to a particular \( \theta \)
  2. favours the learning of skewed tag-transition and output distributions via the use of a prior, \( P(\theta | w) \)
- Performs inference using Gibbs sampling
- Still makes the usual (unrealistic) assumption that a perfect POS lexicon is available

Our Goals

1. Relax this unrealistic assumption by learning the lexicon **automatically** from a small set of tagged sentences
2. Propose two extensions to G&G’s approach for tagging for morphologically-rich, resource-scarce languages
   - Use **Bengali** as our representative language

**Extension 1: Induced Suffix Emission (IS)**

**Motivation**: Suffixes are useful indicators of POS tags

A (somewhat naive) way of exploiting suffixes:

1. Generate a list of induced suffixes from an unlabeled corpus (using Keshava and Pitler’s (2006) algorithm)
2. Create a **suffix-based POS lexicon** by replacing each word in the original (i.e., word-based) POS lexicon, \( W \), with its suffix induced in Step 1
3. Have the HMM emit suffixes rather than words, subject to the constraints in the suffix-based POS lexicon

**Potential problem**: Over-generalization

**Our solution**: Adopt a hybrid approach:

- Emit a word if it is in \( W \), otherwise emit its suffix

**Extension 2: Discriminative Prediction (DP)**

**Motivation**: We can learn from the POS-tagged sentences, \( L \), how to exploit **contextual information** to tag a word. How?

- Learn three types of probabilities from \( L \):
  1. \( P(t | w_{i-2}, w_{i-1}) \): probability of tag \( t \) following a word bigram
  2. \( P(t | w_{i-1}) \): probability of tag \( t \) following a word
  3. \( P(w_i | t) \): probability of a word having tag \( t \)

- **Apply** the Discriminative Prediction Algorithm:
  - If \( w_i \) is in \( L \), assign \( t_i \) to \( w_i \) with \( P(t_i | w_i) \)
  - Else if \( (w_{i-2}, w_{i-1}) \) is in \( L \), assign \( t_i \) to \( w_{i-2}, w_{i-1} \) with \( P(t_i | w_{i-2}, w_{i-1}) \)
  - Else if \( w_{i-1} \) is in \( L \), assign \( t_i \) to \( w_{i-1} \) with \( P(t_i | w_{i-1}) \)
  - Else obtain the tag using the Gibbs sampler

**Evaluation**

**Goal**: Evaluate our two extensions to G&G’s tagging model using POS lexicons constructed by three methods

**Corpus**: Bengali dataset from IJCNLP-08 workshop, which comprises a 50K-token training set & a 30K-token test set

**Training set**: for constructing POS lexicons

**Test set**: for evaluating model accuracy

**Tagset**: IIIT Hyderabad’s POS tagset reduced to 15 tags

**Inference**: running 5K iterations of the Gibbs sampler; hyperparameters learned by Metropolis-Hastings

**Lexicon Construction Methods**

- **Lexicon 1**: Includes only the words that appear at least \( d \) times in the test data
- **Lexicon 2**: Includes only the words that appear at least \( d \) times in the training data
- **Lexicon 3**: Includes only the words and their tags that appear in the training data (\( L \))

**Results using Lexicon 3**

**POS tagging models**:
- **BHMM (Baseline)**: G&G’s fully-Bayesian tagging model
- **BHMM+IS**: BHMM with the induced suffix extension
- **BHMM+IS+DP**: BHMM with both extensions

**Learning curves** of the POS tagging models:

**Discussions**

- Results show that both extensions are useful – BHMM+IS and BHMM+IS+DP outperform BHMM by 8–13% and 12–17%, respectively
- **Major sources of errors**: NN vs. NNP (8.4%), NN vs. JJ (6.9%), VM vs. VAUX (5.9%), VM vs. NN (5.1%)
- **Ambiguous token rate** ranges from 57.7% with 5.1 tags/token (50K) to 61.5% with 8.1 tags/token (5K)
- **Unseen word rate** ranges from 25% (50K) to 50% (5K)
- BHMM+IS also outperforms BHMM using Lexicon 1 and Lexicon 2 by 4–9% and 5–10%, respectively