Plan

- Choose a topic. I will give you options, but you are free to choose anything, subject to my approval.
- Work in groups of 2-3 (if you want to work alone, please send me an email stating why this is necessary).
- Email by next Thursday (4/26) with your group members and topic.
- Interim report and meeting with me last week of class.
- Final report due 5/22. Write-up should be between 4-8 pages.

Corpus

- Orwell's Nineteen Eighty Four (~100k words)
- Slavic: Bulgarian, Czech, Serbian, Slovene
- Uralic: Hungarian, Estonian
- Romance: Romanian
- Germanic: English

Tagset: Full morpho-syntactic tagging (14 coarse POS tags, many many fine grained tags)
Potential Projects

- Supervised learning of ...
- Unsupervised learning of ...
- Projection from one language to another...
  ~ David Yarowsky and Grace Ngai. Inducing multilingual POS taggers and NP bracketers via robust projection across aligned corpora.
- Morphology, POS, both?
- Extract characters and social relations?

Question 4: Linguistics

We have discussed various levels of linguistic structure, including orthography, phonetics, morphology, syntax, and pragmatics. Each sentence below violates the standard rules of English on one of these levels. Label each sentence accordingly.

- I like many cats.
- I like cats many.
- Q: Would you mind passing the cat? A: No, I wouldn’t. (nothing happens)
**Question 2: Trigram HMMs**

In class we discussed the bigram HMM model for part-of-speech tagging. Assume that we have a vocabulary of \( V \) words and a set of \( K \) part-of-speech tags. We define the joint probability of a sentence and its tags as follows:

\[
P(\mathbf{w}, \mathbf{\lambda}) = P(\text{STOP})l_0 \cdot \prod_{i=1}^{L} P(l_i | l_{i-1}) P(w_i | l_i)
\]

where \( \mathbf{w} \) and \( \mathbf{\lambda} \) are word and tag sequences, respectively, both of length \( L \). Given an HMM and a sentence \( \mathbf{w} \), we would like to efficiently compute \( P(\mathbf{w}|\mathbf{\lambda}) \).

**Question (a):** How many possible tag sequences exist for this sentence? Is it feasible to separately compute the probability of each?

**Question (b):** Fortunately, the Viterbi algorithm allows us to efficiently find \( \text{argmax}_\lambda P(\mathbf{w} | \mathbf{\lambda}) \). How does this tag sequence relate to the desired tag sequence \( \text{argmax}_\lambda P(\mathbf{w} | \mathbf{\lambda}) \) as your answer.

**Question (c):** The Viterbi algorithm is an example of a dynamic program. In one short sentence, describe the intuition behind Viterbi: how it allows us to find the highest probability sequence from among an exponential number of possibilities in polynomial time.

---

**Question (d):** Now let’s consider a trigram HMM:

\[
P(\mathbf{w}, \mathbf{\lambda}) = P(\text{STOP})l_0 \cdot \prod_{i=1}^{L} P(l_i | l_{i-1}, l_{i-2}) P(w_i | l_i)
\]

There are two ways to modify our set-up account for this. One way involves artificially increasing the tag-set, leaving the Viterbi algorithm unchanged. The other way involves keeping the tag-set unchanged, but altering Viterbi. Describe these two methods, using pseudo-code or recursive equations where appropriate. What are the new time and space complexities of these two methods?
**Viterbi Equations**

- **max prob of being in state** $q_0$
  - at time $t$
  - $V[0, q_0] = 1$
  - **recursion** transition prob
  - $V[t, q_j] = \max_{q_i} V[t - 1, q_i] \times a_{i,j} \times b_{j,w_t}$
- **goal** transition to end state
  - $\max_{q_i} V[n, q_i] \times a_{i,f}$

**Viterbi Pseudocode**

- $V[0, q_0] \leftarrow 0$
- $V[0, q_0] \leftarrow 1$
- for $t = 1 \ldots n$
  - for $j = 1 \ldots |Q|$
    - for $i = 1 \ldots |Q|$
      - $V[t, q_j] \leftarrow \max\{ V[t, q_j], V[t - 1, q_i] \times a_{i,j} \times b_{j,w_t} \}$
  - for $i = 1 \ldots |Q|$
    - $V[n + 1, q] \leftarrow \max\{ V[t + 1, q], V[n, q] \times a_{i,f} \}$
- return $V[n + 1, q_f]$

**Viterbi**

- Bill directed plays about English kings.

```
q0 1
q1
q2
q3
q4
...n
q_f
```
Bill directed plays about English kings.

\[ V[t, q_i] = \max_q V[t-1, q] \times a_{i,j} \times b_{j,w_t} \]

Probabilistic CKY

<table>
<thead>
<tr>
<th>span symbol</th>
<th>production</th>
<th>prob</th>
<th>Recursion</th>
</tr>
</thead>
<tbody>
<tr>
<td>( C[i-1, i, x_i] ) = 1</td>
<td>( p(A \rightarrow x_i) )</td>
<td>( C[i, k, Y] = \max_{x_j \in N, Y \in N} p(V \rightarrow Y Z) \times C[i, j, Y] \times C[j, k, Z] )</td>
<td></td>
</tr>
<tr>
<td>( p(\gamma^*</td>
<td>{x_1, ..., x_n}) = C[0, n, S] )</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Probabilistic CKY Pseudocode

- Input: \( x, G \)
- Output: \( C \)
  - for \( i = 1 \ldots n \)
    - \( C[A, i-1, i] = p(A \rightarrow x) \)
  - for \( \ell = 2 \ldots n \)
    - for \( i = 0 \ldots n - \ell \)
      - \( k = i + \ell \)
      - for \( j = i + 1 \ldots k - 1 \)
        - \( C[X, i, k] = \max(C[X, i, k], p(X \rightarrow Y Z) \times C[Y, i, j] \times C[Z, j, k]) \)
  - return \( C[S, 0, n] \)
Problems with PCFGs

- What’s different between basic PCFG scores here?
- What (lexical) correlations need to be scored?

Grammar Refinement

- Lexicalization [Collins ‘99, Charniak ‘00]
- Structure Annotation [Johnson ‘98, Klein&Manning ‘03]