So Far …

• We’ve talked mainly about building models from either annotated data or unannotated data.
• We’ve focused on classes of models that predict different kinds of structure.
• We’ve explored different ways to estimate those models.
• Today, we focus on mixing labeled and unlabeled data.
Word Sense Disambiguation

• Can a word sense disambiguation?
• Homographs
  – park the car vs. walk in the park
  – water the plant vs. work at the plant
  – the $x$ and $y$ axes vs. chopping down trees with axes
  – palm of my hand vs. palm tree
• Assume we know the set of senses for a word type. Can we pick the right one for ambiguous tokens in text?
• Note: the “output variable” ranges over a small, finite set. So machine learning people love WSD.
## One Sense Per Discourse

\[ p(\text{more than one occurrence}) \]

\[ p(\text{most frequent sense} \mid \text{more than one occurrence}) \]

<table>
<thead>
<tr>
<th>Word</th>
<th>Senses</th>
<th>Accuracy</th>
<th>Applicability</th>
</tr>
</thead>
<tbody>
<tr>
<td>plant</td>
<td>living/factory</td>
<td>99.8 %</td>
<td>72.8 %</td>
</tr>
<tr>
<td>tank</td>
<td>vehicle/contnr</td>
<td>99.6 %</td>
<td>50.5 %</td>
</tr>
<tr>
<td>poach</td>
<td>steal/boil</td>
<td>100.0 %</td>
<td>44.4 %</td>
</tr>
<tr>
<td>palm</td>
<td>tree/hand</td>
<td>99.8 %</td>
<td>38.5 %</td>
</tr>
<tr>
<td>axes</td>
<td>grid/tools</td>
<td>100.0 %</td>
<td>35.5 %</td>
</tr>
<tr>
<td>sake</td>
<td>benefit/drink</td>
<td>100.0 %</td>
<td>33.7 %</td>
</tr>
<tr>
<td>bass</td>
<td>fish/music</td>
<td>100.0 %</td>
<td>58.8 %</td>
</tr>
<tr>
<td>space</td>
<td>volume/outer</td>
<td>99.2 %</td>
<td>67.7 %</td>
</tr>
<tr>
<td>motion</td>
<td>legal/physical</td>
<td>99.9 %</td>
<td>49.8 %</td>
</tr>
<tr>
<td>crane</td>
<td>bird/machine</td>
<td>100.0 %</td>
<td>49.1 %</td>
</tr>
<tr>
<td>Average</td>
<td></td>
<td>99.8 %</td>
<td>50.1 %</td>
</tr>
</tbody>
</table>
One Sense Per Discourse

• This is a fancy way of saying that, within a discourse (e.g., document), ambiguous tokens of the same type tend to be correlated.
One Sense Per Collocation

• Certain features of the context are very strong predictors for one sense or another.
  – … power plant …
  – … palm of …
  – … the park …

• This is a fancy way of saying that (some) collocations are excellent features.
The Yarowsky Algorithm

• Given: ambiguous word type \( w \), lots of text

1. Choose a few seed collocations for each sense and label data in those collocations.
A = SENSE-A training example
B = SENSE-B training example
? = currently unclassified training example
\[
\text{Life} = \text{Set of training examples containing the collocation "life".}
\]
The Yarowsky Algorithm

- Given: ambiguous word type $w$, lots of text
  1. Choose a few seed collocations for each sense and label data in those collocations.
  2. Train a supervised classifier on the labeled examples. (Yarowsky used a decision list.)
  3. Label all examples. Keep the labels about which the supervised classifier was highly confident (above threshold).
     - Optionally, exploit one-sense-per-discourse to “spread” a label throughout the discourse.
  4. Go to 2.
Whence Seeds?

• Yarowsky suggests:
  – dictionary definitions
  – single defining collocate (e.g., from WordNet)
  – label extremely common collocations

• See Eisner & Karakos (2005) for more about seeds.
## Experimental Results

<table>
<thead>
<tr>
<th>Word</th>
<th>Senses</th>
<th>Samp. Size</th>
<th>% Major Sense</th>
<th>Supvsd Algrtm</th>
<th>Seed Training Options</th>
<th>(7) + OSPD</th>
<th>Schütze Algrtm</th>
</tr>
</thead>
<tbody>
<tr>
<td>plant</td>
<td>living/factory</td>
<td>7538</td>
<td>53.1</td>
<td>97.7</td>
<td>97.1</td>
<td>98.3</td>
<td>98.6</td>
</tr>
<tr>
<td>space</td>
<td>volume/outer</td>
<td>5745</td>
<td>50.7</td>
<td>93.9</td>
<td>89.1</td>
<td>93.5</td>
<td>93.3</td>
</tr>
<tr>
<td>tank</td>
<td>vehicle/container</td>
<td>11420</td>
<td>58.2</td>
<td>97.1</td>
<td>94.2</td>
<td>95.8</td>
<td>96.1</td>
</tr>
<tr>
<td>motion</td>
<td>legal/physical</td>
<td>11968</td>
<td>57.5</td>
<td>98.0</td>
<td>93.5</td>
<td>97.4</td>
<td>97.8</td>
</tr>
<tr>
<td>bass</td>
<td>fish/music</td>
<td>1859</td>
<td>56.1</td>
<td>97.8</td>
<td>96.6</td>
<td>97.7</td>
<td>98.5</td>
</tr>
<tr>
<td>palm</td>
<td>tree/hand</td>
<td>1572</td>
<td>74.9</td>
<td>96.5</td>
<td>93.9</td>
<td>95.8</td>
<td>95.5</td>
</tr>
<tr>
<td>poach</td>
<td>steal/boil</td>
<td>585</td>
<td>84.6</td>
<td>97.1</td>
<td>96.6</td>
<td>97.2</td>
<td>98.4</td>
</tr>
<tr>
<td>axes</td>
<td>grid/tools</td>
<td>1344</td>
<td>71.8</td>
<td>95.5</td>
<td>94.0</td>
<td>94.7</td>
<td>96.8</td>
</tr>
<tr>
<td>duty</td>
<td>tax/obligation</td>
<td>1280</td>
<td>50.0</td>
<td>93.7</td>
<td>90.4</td>
<td>93.2</td>
<td>93.9</td>
</tr>
<tr>
<td>drug</td>
<td>medicine/narcotic</td>
<td>1380</td>
<td>50.0</td>
<td>93.0</td>
<td>90.4</td>
<td>91.4</td>
<td>93.3</td>
</tr>
<tr>
<td>sake</td>
<td>benefit/drink</td>
<td>407</td>
<td>82.8</td>
<td>96.3</td>
<td>59.6</td>
<td>96.1</td>
<td>96.1</td>
</tr>
<tr>
<td>crane</td>
<td>bird/machine</td>
<td>2145</td>
<td>78.0</td>
<td>96.6</td>
<td>92.3</td>
<td>93.6</td>
<td>95.4</td>
</tr>
<tr>
<td>AVG</td>
<td></td>
<td>3936</td>
<td>63.9</td>
<td>96.1</td>
<td>90.6</td>
<td>94.8</td>
<td>95.5</td>
</tr>
</tbody>
</table>
Several Ways to Think About This

• Like **Viterbi EM**, but new features induced on each iteration.
  – Yarowsky didn’t use a probability model in the conventional way; he used a decision list.
• Leveraging several assumptions about the data to help each other
  – One sense per collocation (inside the decision list)
  – One sense per discourse (finding new collocations)
• Meta-learner in which any supervised method can be nested!
Important Note

• Yarowsky’s algorithm is not just for word sense! Similar algorithms have been applied to diverse problems:
  – Named entity recognition
  – Grammatical gender prediction
  – Morphology learning
  – Bilingual lexicon induction
  – Parsing
Cotraining
(Blum and Mitchell, 1998)

• Rather difficult paper, but rather elegant idea.
• Input is $x$; suppose it can be broken into $x_1$ and $x_2$, disjoint “views” of $x$.
• Cotraining iteratively builds two classifiers (one on $x_1$ and one on $x_2$) and uses each to help improve the other.
Cotraining

- Given labeled examples $L$, unlabeled examples $U$
  1. Train $c_1$ on $x_1$ from $L$, and train $c_2$ on $x_2$ from $L$. (B&M used Naïve Bayes.)
  2. Label examples in $U$ using $c_1$; add those it’s most confident about for each class to $L$.
  3. Ditto ($c_2$).
  4. Go to 1.
WebKB-Course Data

- Data: CS department sites from four universities
- Task: Is a given page a course web page or not?
- $X_1$: bag of words in the page
- $X_2$: bag of words in *hyperlinks to the page*

<table>
<thead>
<tr>
<th></th>
<th>Page-based classifier</th>
<th>Hyperlink-based classifier</th>
<th>Combined classifier</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervised training</td>
<td>12.9</td>
<td>12.4</td>
<td>11.1</td>
</tr>
<tr>
<td>Co-training</td>
<td>6.2</td>
<td>11.6</td>
<td>5.0</td>
</tr>
</tbody>
</table>
What’s Different?

• The “view” formulation.
  – Yarowsky has one classifier; B&M have two.
• Yarowsky allows relabeling of unlabeled examples; B&M do not.
• Yarowsky (1995) focused on particular properties of the data and exploited them. No general claims.
• B&M (1998) were seeking a general meta-learner that could leverage unlabeled examples; they actually gave PAC-style learnability results under an assumption that $X_1$ and $X_2$ were conditionally independent given $Y$.
• Unlike EM, neither of these methods maintains posterior distributions over the labels.
Nigam and Ghani (2000)

- Compare EM and cotraining, with the **same** model/features. On the WebKB-Course dataset:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th># Labeled</th>
<th># Unlabeled</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive Bayes</td>
<td>788</td>
<td>0</td>
<td>3.3%</td>
</tr>
<tr>
<td>Co-training</td>
<td>12</td>
<td>776</td>
<td>5.4%</td>
</tr>
<tr>
<td>EM</td>
<td>12</td>
<td>776</td>
<td>4.3%</td>
</tr>
<tr>
<td>Naive Bayes</td>
<td>12</td>
<td>0</td>
<td>13.0%</td>
</tr>
</tbody>
</table>
Nigam and Ghani (2000)

- Ceiling effects?
- Are the content/hyperlink views really independent? (Probably not.) Semi-synthetic experiment:

<table>
<thead>
<tr>
<th>Algorithm</th>
<th># Labeled</th>
<th># Unlabeled</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Naive Bayes</td>
<td>1006</td>
<td>–0–</td>
<td>3.9%</td>
</tr>
<tr>
<td>Co-training</td>
<td>6</td>
<td>1000</td>
<td>3.7%</td>
</tr>
<tr>
<td>EM</td>
<td>6</td>
<td>1000</td>
<td>8.9%</td>
</tr>
<tr>
<td>Naive Bayes</td>
<td>6</td>
<td>–0–</td>
<td>34.0%</td>
</tr>
</tbody>
</table>

- EM > Cotraining
Hybrids

- **EM:**
  \[ ||: A \text{ softly labels data; } A \text{ trains } :|| \]

- **Co-EM:**
  \[ ||: A \text{ softly labels data; } B \text{ trains; } B \text{ softly labels data; } A \text{ trains } :|| \]

- **Co-training:**
  \[ ||: A, B \text{ label a few examples; } A, B \text{ train } :|| \]

- **Self-training:**
  \[ ||: A \text{ labels a few examples; } A \text{ trains } :|| \]
## Results (Synthetic Data)

<table>
<thead>
<tr>
<th>Method</th>
<th>Uses Feature Split?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>Incremental</td>
<td>3.7%</td>
</tr>
<tr>
<td>Iterative</td>
<td>3.3%</td>
</tr>
</tbody>
</table>

- **cotraining**
- **self-training**
- **co-EM**
- **EM**
More Results

- If no natural feature split is available, can split features **randomly**.
- On synthetic data, that actually worked better than the smart split!
- On real data, best results came from self-training (!?!)?
  - Hard to draw any firm conclusions.
  - Possibly has to do with the supervised learner (why not use something more powerful than Naïve Bayes?).
  - Ng and Cardie (2003): more mixed results, but come out in favor of “single-view” algorithms.
  - Critical comment: go back to the objective function!
Abney (2004)

- “Understanding the Yarowsky Algorithm”
- Entirely under-appreciated paper!
- Demonstrates that certain variants of the Yarowsky algorithm are actually optimizing likelihood. Others are optimizing a bound on likelihood.
- Likelihood under what model?
Understanding the Abney Understanding of the Yarowsky Algorithm

• Modifications:
  – Once an originally-unlabeled example is labeled, it stays labeled.
  – Fix threshold at 1/(# classes).

• Assumption: base learner improves KL divergence between empirical distribution and the base model.
  – Either on labeled examples only,
  – or overall (assuming unlabeled examples have uniform empirical)

• Yarowsky’s base learner doesn’t do this; Abney gives variants that do.
  – The “DL-EM” base learners he describes essentially amount to a single step of the EM algorithm.

• The proofs are involved; the insight (I believe) is that the algorithm starts to look more like (Viterbi) EM with some labels fixed so they can’t change.
Cotraining for Parsing?

- Steedman et al. (2003) cotrained two parsers.

<table>
<thead>
<tr>
<th>Collins-CFG</th>
<th>LTAG</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bi-lexical dependencies are between lexicalized nonterminals</td>
<td>Bi-lexical dependencies are between elementary trees</td>
</tr>
<tr>
<td>Can produce novel elementary trees for the LTAG parser</td>
<td>Can produce novel bi-lexical dependencies for Collins-CFG</td>
</tr>
<tr>
<td>When using small amounts of seed data, abstains less often than LTAG</td>
<td>When using small amounts of seed data, abstains more often than Collins-CFG</td>
</tr>
</tbody>
</table>
Parser Self-training
Parser Cotraining

Co-training versus self-training

Co-training rounds

Co-training rounds
Steedman et al., 2003

• Also showed cross-domain improvement (WSJ and Brown corpus).

• If you start with “enough” labeled data, cotraining doesn’t help.
Semisupervised Learning: Hot

• Adaptation to new domains
  – Or languages! Hwa et al., 2002; Wicentowski et al., 2001; Smith and Smith, 2004, …

• Ando and Zhang (2005): use multiple tasks to leverage unlabeled data

• Lessen the cost of annotation projects (annotate fewer examples)

• Interesting theoretical topic (many papers lately)

• So much unlabeled data, how could we not want to learn from it!
Two Important Lessons

- There usually is no unqualified “best” method. All kinds of things affect this. More subtle questions than, “does A beat B”:
  - What conditions lead to better performance for A vs. B?
  - What kinds of errors is A more susceptible to than B?
- Nifty ideas can often be shown (sometimes years later) to have solid mathematical underpinnings.