Supervised Learning

- Systems duplicate correct analyses from training data
- Hand-annotation of data
  - Time-consuming
  - Expensive
  - Hard to adapt for new purposes (tasks, languages, domains, etc.)
- Corpus availability drives research, not tasks
- Example: Penn Treebank
  - 50K Sentences
  - Hand-parsed over several years

Unsupervised Learning

- Systems take raw data and automatically detect patterns
- Why unsupervised learning?
  - More data than annotation
  - Insights into machine learning, clustering
  - Kids learn some aspects of language entirely without supervision
- Here: unsupervised learning
  - Work purely from the forms of the utterances
  - Neither assume nor exploit prior meaning or grounding [cf. Feldman et al.]

Unsupervised Parsing?

- Start with raw text, learn syntactic structure
- Some have argued that learning syntax from positive data alone is impossible:
  - Gold, 1967: Non-identifiability in the limit
  - Chomsky, 1980: The poverty of the stimulus
- Many others have felt it should be possible:
  - Lari and Young, 1990
  - Carroll and Charniak, 1992
  - Alex Clark, 2001
  - Mark Paskin, 2001
  - … and many more, but it didn’t work well (or at all) until the past few years
- Surprising result: it’s possible to get entirely unsupervised parsing to (reasonably) work well!

Learnability

- Learnability: formal conditions under which a class of languages can be learned in some sense
- Setup:
  - Class of languages is $\mathcal{L}$
  - Learner is some algorithm $H$
  - Learner sees a sequences $X$ of strings $x_1 \ldots x_n$
  - $H$ maps sequences $X$ to languages $L$ in $\mathcal{L}$
- Question: for what classes do learners exist?

Learnability: [Gold 67]

- Criterion: identification in the limit
  - A presentation of $L$ is an infinite sequence of $x$’s from $L$ in which each $x$ occurs at least once
  - A learner $H$ identifies $L$ in the limit if for any presentation of $L$, from some point $n$ onward, $H$ always outputs $L$
  - A class $\mathcal{L}$ is identifiable in the limit if there is some single $H$ which correctly identifies in the limit any $L$ in $\mathcal{L}$
- Example: $L = \{a\}, \{a, b\}$ is learnable in the limit
- Theorem [Gold 67]: Any $\mathcal{L}$ which contains all finite languages and at least one infinite language (i.e. is superfinite) is unlearnable in this sense
Learnability: [Gold 67]

- Proof sketch
  - Assume ∏ is superfinite
  - There exists a chain $L_1 \subset L_2 \subset \ldots \subset L_n$
  - Take any learner H assumed to identify ∏
  - Construct the following misleading sequence
    - Present strings from $L_1$ until it outputs $L_2$
    - Present strings from $L_2$ until it outputs $L_3$
    - …
    - This is a presentation of $L_n$, but H won’t identify $L_n$

Learnability: [Horning 69]

- Proof sketch
  - Assume ∏ is a recursively enumerable set of recursive languages (e.g. the set of PCFGs)
  - Assume an ordering on all strings $x_1 < x_2 < \ldots$
  - Define two sequences A and B agree through n if for all $x < x_n$, $x \in A$ if and only if $x \in B$
  - Define the error set $E(L,n,m)$:
    - All sequences such that the first m elements do not agree with L through n
    - These are the sequences which contain early strings outside of L (can’t happen) or fail to contain all the early strings in L (happens less as m increases)
  - Claim: $|E(L,n,m)|$ goes to 0 as m goes to $\infty$
  - Let $d(n)$ be the smallest m such that $|E(L,n,m)| < 2^n$
  - Let $d(n)$ be the largest $d(n)$ in first n languages
  - Learner after $d(n)$, pick first L, that agrees with evidence through n
  - Can only fail for sequence X if X keeps showing up in $E(L,n,d(n))$, which happens infinitely often with probability zero (we skipped some details)

Learnability: [Horning 69]

- Problem: IIL requires that H succeed on each presentation, even the weird ones
- Another criterion: measure one identification
  - Assume a distribution $P_T(x)$ for each L
  - Assume $P_T(x)$ puts non-zero mass on all and only x in L
  - Assume infinite presentation X drawn i.i.d. from $P_T(x)$
  - H measure-one identifies L if probability of drawing an X from which H identifies L is 1
- [Horning 69]: PCFGs can be identified in this sense
  - Note: there can be misleading sequences, they just have to be (infinitely) unlikely

Learnability

- Gold’s result says little about real learners (requirements of IIL are way too strong)
- Horning’s algorithm is completely impractical (needs astronomical amounts of data)
- Even measure-one identification doesn’t say anything about tree structures (or even density over strings)
- Only talks about learning grammatical sets
- Strong generative vs weak generative capacity

Unsupervised Tagging?

- AKA part-of-speech induction

- Task:
  - Raw sentences in
  - Tagged sentences out

- Obvious thing to do:
  - Start with a (mostly) uniform HMM
  - Run EM
  - Inspect results

EM for HMMs: Process

- Alternate between recomputing distributions over hidden variables (the tags) and reestimating parameters
- Crucial step: we want to tally up how many (fractional) counts of each kind of transition and emission we have under current params:

$$\text{count}(w, s) = \sum_{l, w_i = w} P(t_i = s | w)$$

$$\text{count}(s \rightarrow s') = \sum_{i} P(t_i = s, t_{i-1} = s', t_i = s | w)$$

- Same quantities we needed to train a CRF!
Merialdo: Setup

- Some (discouraging) experiments [Merialdo 94]

- Setup:
  - You know the set of allowable tags for each word
  - Learn a supervised model on k training sentences
  - Learn P(t1,t2) on these examples
  - On n > k sentences, re-estimate with EM

- Note: we know allowed tags but not frequencies

Merialdo: Results

<table>
<thead>
<tr>
<th>Number of tagged sentences used for the initial model</th>
<th>0</th>
<th>100</th>
<th>5000</th>
<th>10000</th>
<th>20000</th>
<th>all</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iter</td>
<td>0</td>
<td>100</td>
<td>5000</td>
<td>10000</td>
<td>20000</td>
<td>all</td>
</tr>
<tr>
<td>Correct tags (% words) after MI on 1M words</td>
<td>77.0</td>
<td>95.0</td>
<td>95.4</td>
<td>96.2</td>
<td>96.6</td>
<td>96.9</td>
</tr>
<tr>
<td>1</td>
<td>80.3</td>
<td>95.6</td>
<td>95.8</td>
<td>96.3</td>
<td>96.6</td>
<td>96.7</td>
</tr>
<tr>
<td>2</td>
<td>81.8</td>
<td>95.0</td>
<td>95.7</td>
<td>96.1</td>
<td>96.3</td>
<td>96.4</td>
</tr>
<tr>
<td>3</td>
<td>83.0</td>
<td>95.1</td>
<td>95.6</td>
<td>96.1</td>
<td>96.1</td>
<td>96.2</td>
</tr>
<tr>
<td>4</td>
<td>84.0</td>
<td>95.0</td>
<td>95.2</td>
<td>95.5</td>
<td>95.8</td>
<td>96.0</td>
</tr>
<tr>
<td>5</td>
<td>84.8</td>
<td>95.3</td>
<td>95.4</td>
<td>95.6</td>
<td>95.8</td>
<td>95.8</td>
</tr>
<tr>
<td>6</td>
<td>85.5</td>
<td>95.2</td>
<td>94.9</td>
<td>95.2</td>
<td>95.5</td>
<td>95.6</td>
</tr>
<tr>
<td>7</td>
<td>85.6</td>
<td>95.2</td>
<td>94.7</td>
<td>95.1</td>
<td>95.3</td>
<td>95.5</td>
</tr>
<tr>
<td>8</td>
<td>86.1</td>
<td>95.2</td>
<td>94.6</td>
<td>95.0</td>
<td>95.2</td>
<td>95.4</td>
</tr>
<tr>
<td>9</td>
<td>86.3</td>
<td>95.3</td>
<td>94.9</td>
<td>95.1</td>
<td>95.3</td>
<td>95.5</td>
</tr>
<tr>
<td>10</td>
<td>86.6</td>
<td>95.3</td>
<td>94.8</td>
<td>95.0</td>
<td>95.2</td>
<td>95.2</td>
</tr>
</tbody>
</table>

Distributional Clustering

- Three main variants on the same idea:
  - Pairwise similarities and heuristic clustering
    - E.g. [Finch and Chater 92]
    - Produces dendrograms
  - Vector space methods
    - E.g. [Shuetze 93]
    - Models of ambiguity
  - Probabilistic methods
    - Various formulations, e.g. [Lee and Pereira 99]

Distributional Clustering

- [Finch and Chater 92, Shuetze 93, many others]

Nearest Neighbors

Dendrograms
A Probabilistic Version?

\[ P(S, C) = \prod_i P(c_i | w_i) P(w_{i+1} | c_i) \]

- the president said that the downturn was over

Weakly Supervised Learning

- Newly remodeled 2 Bdrms/1 Bath, spacious upper unit, located in Hilltop Mall area. Walking distance to shopping, public transportation, schools and park. Paid water and garbage. No dogs allowed.

<table>
<thead>
<tr>
<th>Prototype Lists</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kitchen, laundry</td>
</tr>
<tr>
<td>Breakfast room</td>
</tr>
<tr>
<td>Tiled, fruit</td>
</tr>
<tr>
<td>Merchants, store</td>
</tr>
<tr>
<td>Large, fast</td>
</tr>
<tr>
<td>Restrict, cat, smoking</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Information Extraction</th>
<th>English POS</th>
</tr>
</thead>
<tbody>
<tr>
<td>From [Haghighi and Klein 06]</td>
<td></td>
</tr>
</tbody>
</table>

Context-Free Grammars

- Looks like a context-free grammar.
- Can model a tree as a collection of context-free rewrites (with probabilities attached).

\[ S \rightarrow NP \quad NP \rightarrow PP \quad NP \rightarrow \text{Shaw Publishing} \quad \text{acquired} \quad \text{30\%} \quad \text{of} \quad \text{American} \quad \text{City} \quad \text{in} \quad \text{March} \]

Early Approaches: Structure Search

- Incremental grammar learning, chunking [Wolff 88, Langley 82, many others]
- Can recover synthetic grammars
- An (extremely good / lucky) result of incremental structure search:

<table>
<thead>
<tr>
<th>N-ary or zero determiner NP</th>
<th>Transitive VPs (complementation)</th>
<th>Intransitive S</th>
</tr>
</thead>
<tbody>
<tr>
<td>sNN → NN</td>
<td>sNN</td>
<td>sNN</td>
</tr>
<tr>
<td>sNN → sNN</td>
<td>sNN</td>
<td>sNN</td>
</tr>
<tr>
<td>sNP → sNP</td>
<td>sNP</td>
<td>sNP</td>
</tr>
<tr>
<td>sNP → PRPS</td>
<td>sNP</td>
<td>sNP</td>
</tr>
<tr>
<td>Proper NP</td>
<td>sNNP</td>
<td>sNNP</td>
</tr>
<tr>
<td>sNNP → sNNP</td>
<td>sNNP</td>
<td>sNNP</td>
</tr>
<tr>
<td>Transitive VPs (adjacency)</td>
<td>sNP</td>
<td>sNP</td>
</tr>
<tr>
<td>Transitive VPs (trans)</td>
<td>sNP</td>
<td>sNP</td>
</tr>
</tbody>
</table>

- Looks good, ... but can't parse in the wild.

Idea: Learn PCFGs with EM

- Classic experiments on learning PCFGs with Expectation-Maximization [Lari and Young, 1990]

\[ \{ X_1, X_2 \ldots X_n \} \]

- Full binary grammar over \( n \) symbols
- Parse uniformly/randomly at first
- Re-estimate rule expectations off of parses
- Repeat

<table>
<thead>
<tr>
<th>Their conclusion:</th>
</tr>
</thead>
<tbody>
<tr>
<td>it doesn't really work.</td>
</tr>
</tbody>
</table>

Problem: Model Symmetries

- How does this relate to trees

\[ X_i \quad X_j \quad X_k \]

\[ S \quad YP \quad NP \quad N \quad V \quad X_i \quad X_k \]

\[ NOUN \quad VERB \quad ADJ \quad NOUN \]

\[ NOUN \quad VERB \quad ADJ \quad NOUN \]
Other Approaches

- Evaluation: fraction of nodes in gold trees correctly posited in proposed trees (unlabeled recall)
- Some recent work in learning constituency:
  - [Adrians, 99] Language grammars aren’t general PCFGs
  - [Clark, 01] Mutual-information filters detect constituents, then an MDL-guided search assembles them
  - [van Zaanen, 00] Finds low edit-distance sentence pairs and extracts their differences

Right-Branching Baseline

- English trees tend to be right-branching, not balanced
  
  they were unwilling to agree to new terms

- A simple (English-specific) baseline is to choose the right chain structure for each sentence

| van Zaanen, 00 | 35.6 |

Idea: Distributional Syntax?

- Can we use distributional clustering for learning syntax? [Harris, 51]

<table>
<thead>
<tr>
<th>Span</th>
<th>Context</th>
</tr>
</thead>
<tbody>
<tr>
<td>fell in september</td>
<td>payrolls __ •</td>
</tr>
<tr>
<td>payrolls fell in</td>
<td>factory __ sept</td>
</tr>
</tbody>
</table>

Problem: Identifying Constituents

- Distributional classes are easy to find…

... but figuring out which are constituents is hard.

A Nested Distributional Model

- We’d like a model that:
  - Ties spans to linear contexts (like distributional clustering)
  - Considers only proper tree structures (like a PCFG model)
  - Has no symmetries to break (like a dependency model)

Constituent-Context Model (CCM)

\[
P(S|T) = \prod_{(i,j,c)} P(\text{factory payrolls fell in september})
\]

\[
= \prod_{(i,j,c)} P(f \in p) P(f \in s | +) P(s | +) P(f \in p) P(f \in s | -) P(s | -)
\]

\[
P(\text{fell} | +)
\]
Results: Constituency

<table>
<thead>
<tr>
<th>Constituency</th>
<th>Right-Branch</th>
</tr>
</thead>
<tbody>
<tr>
<td>Treebank Parse</td>
<td>CCM Parse</td>
</tr>
<tr>
<td>[Image of parse trees]</td>
<td>70.0</td>
</tr>
</tbody>
</table>

Spectrum of Systematic Errors

- CCM analysis better
- Treebank analysis better

<table>
<thead>
<tr>
<th>Analysis</th>
<th>Inside NPs</th>
<th>Possesives</th>
<th>Verb groups</th>
</tr>
</thead>
<tbody>
<tr>
<td>CCM</td>
<td>the [lazy cat]</td>
<td>John [’s cat]</td>
<td>[will be] there</td>
</tr>
<tr>
<td>Treebank</td>
<td>the lazy cat</td>
<td>[John ]’s cat</td>
<td>will [be there]</td>
</tr>
<tr>
<td>CCM Right?</td>
<td>Yes</td>
<td>Maybe</td>
<td>No</td>
</tr>
</tbody>
</table>

But the worst errors are the non-systematic ones (~25%)

Syntactic Parsing

- Parsing assigns structures to sentences.
  - Shaw Publishing acquired 30 % of American City in March

- Dependency structure gives attachments.
  - WHEN
  - Shaw Publishing acquired 30 % of American City in March

Idea: Lexical Affinity Models

- Words select other words on syntactic grounds
  - congress narrowly passed the amended bill
  - Link up pairs with high mutual information
    - [Yuret, 1998]: Greedy linkage
    - [Paskin, 2001]: Iterative re-estimation with EM
  - Evaluation: compare linked pairs to a gold standard

<table>
<thead>
<tr>
<th>Method</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Paskin, 2001</td>
<td>39.7</td>
</tr>
</tbody>
</table>

Problem: Non-Syntactic Affinity

- Mutual information between words does not necessarily indicate syntactic selection.
  - congress narrowly passed the amended bill
  - expect brushbacks but no beanballs
  - a new year begins in new york

Idea: Word Classes

- Individual words like congress are entwined with semantic facts about the world.
- Syntactic classes, like NOUN and ADVERB are bleached of word-specific semantics.
- Automatic word classes more likely to look like DAYS-OF-WEEK or PERSON-NAME.
- We could build dependency models over word classes. [cf. Carroll and Charniak, 1992]
**Problems: Word Class Models**

<table>
<thead>
<tr>
<th></th>
<th>CCM+DMV</th>
<th>DMV</th>
<th>CCM+DMV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>41.7</td>
<td>44.7</td>
<td></td>
</tr>
</tbody>
</table>

- Issues:
  - Too simple a model—doesn't work much better supervised
  - No representation of valence (number of arguments)

**Local Representations**

```
           head
          /   \
           \   /
            - inequality
          /     \
          arg  distance
```

<table>
<thead>
<tr>
<th>Classes?</th>
<th>Distance</th>
<th>Local Factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pascal 01</td>
<td>❌ ❌</td>
<td>P(a</td>
</tr>
</tbody>
</table>

**Common Errors: Dependency**

<table>
<thead>
<tr>
<th>Overproposed Dependencies</th>
<th>Underproposed Dependencies</th>
</tr>
</thead>
<tbody>
<tr>
<td>DET → N</td>
<td>DET → N</td>
</tr>
<tr>
<td>N-PROP ↔ N-PROP</td>
<td>N-PROP → N-PROP</td>
</tr>
<tr>
<td>NUM → NUM</td>
<td>PREP → N</td>
</tr>
<tr>
<td>PREP ↔ DET</td>
<td>N → V-PRES</td>
</tr>
<tr>
<td>DET ↔ N-PL</td>
<td>DET → N-PL</td>
</tr>
<tr>
<td>DET → PREP</td>
<td>N ↔ PREP</td>
</tr>
<tr>
<td>DET → V-PAST</td>
<td>NUM → DET</td>
</tr>
<tr>
<td>DET → V-PRES</td>
<td>N → V-PAST</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DET → N</th>
<th>DET → N</th>
</tr>
</thead>
<tbody>
<tr>
<td>N-PROP ↔ N-PROP</td>
<td>N-PROP → N-PROP</td>
</tr>
<tr>
<td>NUM → NUM</td>
<td>PREP → N</td>
</tr>
<tr>
<td>PREP ↔ DET</td>
<td>N → V-PRES</td>
</tr>
<tr>
<td>DET ↔ N-PL</td>
<td>DET → N-PL</td>
</tr>
<tr>
<td>DET → PREP</td>
<td>N ↔ PREP</td>
</tr>
<tr>
<td>DET → V-PAST</td>
<td>NUM → DET</td>
</tr>
<tr>
<td>DET → V-PRES</td>
<td>N → V-PAST</td>
</tr>
</tbody>
</table>

**Results: Dependencies**

- Situation so far:
  - Task: unstructured text in, word pairs out
  - Previous results were below baseline
  - We modeled word classes [cf. Carroll & Chamiak 92]
  - We added a model of distance [cf. Collins 99]
  - Resulting model is substantially over baseline
  - ... but we can do much better

**Results: Combined Models**

**Dependency Evaluation (Undir. Dep. Acc.)**

<table>
<thead>
<tr>
<th></th>
<th>CCM+DMV</th>
<th>DMV</th>
<th>CCM+DMV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>45.6</td>
<td>62.7</td>
<td>64.7</td>
</tr>
<tr>
<td>CCM+DMV</td>
<td>64.7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Constituency Evaluation (Unlabeled Recall)**

<table>
<thead>
<tr>
<th></th>
<th>CCM+DMV</th>
<th>DMV</th>
<th>CCM+DMV</th>
</tr>
</thead>
<tbody>
<tr>
<td>Random</td>
<td>39.4</td>
<td>46.7</td>
<td>49.6</td>
</tr>
<tr>
<td>CCM+DMV</td>
<td>64.7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Supervised PCFG constituency recall is at 92.8
- Qualitative improvements
  - Subject-verb groups gone, modifier placement improved

**How General is This?**

**Constituency Evaluation**

<table>
<thead>
<tr>
<th></th>
<th>CCM+DMV</th>
<th>DMV</th>
<th>CCM+DMV</th>
</tr>
</thead>
<tbody>
<tr>
<td>English (7422 sentences)</td>
<td>39.4</td>
<td>46.7</td>
<td>49.6</td>
</tr>
<tr>
<td>CCM+DMV</td>
<td>64.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>German (2175 sentences)</td>
<td>49.6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCM+DMV</td>
<td>69.7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Chinese (2473 sentences)</td>
<td>35.5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCM+DMV</td>
<td>54.2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CCM+DMV</td>
<td>60.0</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Dependency Evaluation**