From Binary to Extreme Classification
### Q&A

<table>
<thead>
<tr>
<th>Q: How do I get into the online section?</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>A:</strong> Sorry! I erroneously claimed we would automatically add you to the online section. Here’s the correct answer:</td>
</tr>
</tbody>
</table>

To join the online section, email Dorothy Holland-Minkley at dfh@andrew.cmu.edu stating that you would like to join the online section.

Why the extra step? We want to make sure you’ve seen the **non-professional video recording** and are okay with the quality.
Q&A

**Q:** Will I get off the waitlist?

**A:** Don’t be on the waitlist. Just email Dorothy to join the online section instead!
### Q&A

<table>
<thead>
<tr>
<th>Q:</th>
<th>Can I move between 10-418 and 10-618?</th>
</tr>
</thead>
<tbody>
<tr>
<td>A:</td>
<td>Yes. Just email Dorothy Holland-Minkley at <a href="mailto:dfh@andrew.cmu.edu">dfh@andrew.cmu.edu</a> to do so.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Q:</th>
<th>When is the last possible moment I can move between 10-418 and 10-618?</th>
</tr>
</thead>
<tbody>
<tr>
<td>A:</td>
<td>I’m not sure. We’ll announce on Piazza once I have an answer.</td>
</tr>
</tbody>
</table>
Q: Why do interactions appear between variables in this example?

A: Consider the test time setting:

- Author writes a new article (vector $x$)
- Infobox is empty
- ML system must populate all fields (vector $y$) at once
- Interactions that were seen (i.e. vector $y$) at training time are unobserved at test time – so we wish to model them
ROADMAP
How do we get from Classification to Structured Prediction?

1. We start with the simplest decompositions (i.e. classification)
2. Then we formulate structured prediction as a search problem (decomposition of into a sequence of decisions)
3. Finally, we formulate structured prediction in the framework of graphical models (decomposition into parts)
A **joint distribution** defines a probability $p(x)$ for each assignment of values $x$ to variables $X$. This gives the **proportion** of samples that will equal $x$. 

Sample 1: $n$, $v$, $p$, $d$, $n$
Sample 2: $n$, $n$, $v$, $d$, $n$
Sample 3: $n$, $v$, $p$, $d$, $n$
Sample 4: $v$, $n$, $p$, $d$, $n$
Sample 5: $v$, $n$, $p$, $d$, $n$
Sample 6: $n$, $v$, $p$, $d$, $n$
A joint distribution defines a probability $p(x)$ for each assignment of values $x$ to variables $X$. This gives the proportion of samples that will equal $x$. 
Sampling from a Joint Distribution

A joint distribution defines a probability $p(x)$ for each assignment of values $x$ to variables $X$. This gives the proportion of samples that will equal $x$. 

Sample 1:
- time
- flies
- like
- an
- arrow

Sample 2:
- time
- flies
- like
- an
- arrow

Sample 3:
- flies
- fly
- with
- their
- wings

Sample 4:
- with
- time
- you
- will
- see
Factors have local opinions (≥ 0)

Each black box looks at some of the tags $X_i$ and words $W_i$

Note: We chose to reuse the same factors at different positions in the sentence.
Factors have local opinions (≥ 0)

Each black box looks at some of the tags $X_i$ and words $W_i$

$$p(n, v, p, d, n, \text{time, flies, like, an, arrow}) = ?$$
Global probability = product of local opinions

Each black box looks at some of the tags $X_i$ and words $W_i$

$$p(n, v, p, d, n, \text{time, flies, like, an, arrow}) = \frac{1}{Z}(4 * 8 * 5 * 3 * \ldots)$$

Uh-oh! The probabilities of the various assignments sum up to $Z > 1$. So divide them all by $Z$. 

 Uh-oh! The probabilities of the various assignments sum up to $Z > 1$. So divide them all by $Z$. 
Markov Random Field (MRF)

Joint distribution over tags $X_i$ and words $W_i$.
The individual factors aren’t necessarily probabilities.

$$p(n, v, p, d, n, \text{time, flies, like, an, arrow}) = \frac{1}{\mathcal{Z}} (4 \times 8 \times 5 \times 3 \times \ldots)$$
Hidden Markov Model

But sometimes we choose to make them probabilities. Constrain each row of a factor to sum to one. Now $Z = 1$.

$$p(n, v, p, d, n, \text{time, flies, like, an, arrow}) = \frac{1}{Z} (0.3 \times 0.8 \times 0.2 \times 0.5 \times \ldots)$$
Markov Random Field (MRF)

Joint distribution over tags $X_i$ and words $W_i$

$$p(n, v, p, d, n, \text{time, flies, like, an, arrow}) = \frac{1}{Z} (4 \times 8 \times 5 \times 3 \times \ldots)$$
Conditional Random Field (CRF)

Conditional distribution over tags $X_i$ given words $w_i$. The factors and Z are now specific to the sentence $w$.

$$p(n, v, p, d, n \mid \text{time, flies, like, an, arrow}) = \frac{1}{Z} (4 * 8 * 5 * 3 * \ldots)$$
BACKGROUND: BINARY CLASSIFICATION
Linear Models for Classification

- Key idea: Try to learn this hyperplane directly

- Directly modeling the hyperplane would use a decision function:

\[ h(x) = \text{sign}(\theta^T x) \]

for:

\[ y \in \{-1, +1\} \]
(Online) Perceptron Algorithm

**Data:** Inputs are continuous vectors of length $M$. Outputs are discrete.

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots$$

where $x \in \mathbb{R}^M$ and $y \in \{+1, -1\}$

**Prediction:** Output determined by hyperplane.

$$\hat{y} = h_\theta(x) = \text{sign}(\theta^T x)$$

Assume $\theta = [b, w_1, \ldots, w_M]^T$ and $x_0 = 1$

**Learning:** Iterative procedure:

- **initialize** parameters to vector of all zeroes
- **while** not converged
  - **receive** next example $(x^{(i)}, y^{(i)})$
  - **predict** $y' = h(x^{(i)})$
  - **if** positive mistake: **add** $x^{(i)}$ to parameters
  - **if** negative mistake: **subtract** $x^{(i)}$ from parameters
(Binary) Logistic Regression

**Data:** Inputs are continuous vectors of length $M$. Outputs are discrete. \[ D = \{ x^{(i)}, y^{(i)} \} _{i=1}^{N} \text{ where } x \in \mathbb{R}^M \text{ and } y \in \{0, 1\} \]

**Model:** Logistic function applied to dot product of parameters with input vector. \[ p_\theta(y = 1 | x) = \frac{1}{1 + \exp(-\theta^T x)} \]

**Learning:** finds the parameters that minimize some objective function. \[ \theta^* = \arg\min_\theta J(\theta) \]

**Prediction:** Output is the most probable class. \[ \hat{y} = \arg\max_{y \in \{0,1\}} p_\theta(y | x) \]
Support Vector Machines (SVMs)

**Hard-margin SVM (Primal)**

\[
\min_{w, b} \frac{1}{2} \|w\|_2^2 \\
\text{s.t. } y^{(i)}(w^T x^{(i)} + b) \geq 1, \quad \forall i = 1, \ldots, N
\]

**Hard-margin SVM (Lagrangian Dual)**

\[
\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y^{(i)} y^{(j)} x^{(i)} \cdot x^{(j)} \\
\text{s.t. } \alpha_i \geq 0, \quad \forall i = 1, \ldots, N \\
\sum_{i=1}^{N} \alpha_i y^{(i)} = 0
\]

**Soft-margin SVM (Primal)**

\[
\min_{w, b} \frac{1}{2} \|w\|_2^2 + C \left( \sum_{i=1}^{N} e_i \right) \\
\text{s.t. } y^{(i)}(w^T x^{(i)} + b) \geq 1 - e_i, \quad \forall i = 1, \ldots, N \\
e_i \geq 0, \quad \forall i = 1, \ldots, N
\]

**Soft-margin SVM (Lagrangian Dual)**

\[
\max_{\alpha} \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j y^{(i)} y^{(j)} x^{(i)} \cdot x^{(j)} \\
\text{s.t. } 0 \leq \alpha_i \leq C, \quad \forall i = 1, \ldots, N \\
\sum_{i=1}^{N} \alpha_i y^{(i)} = 0
\]
Decision Trees

\[ \{D1, D2, \ldots, D14\} \]
\[ [9+, 5-] \]

Outlook

Sunny \\
Overcast \\
Rain

\{D1, D2, D8, D9, D11\} \\
\{D3, D7, D12, D13\} \\
\{D4, D5, D6, D10, D14\}

\[ [2+, 3-] \] \\
\[ [4+, 0-] \] \\
\[ [3+, 2-] \]

? \\
Yes \\
?

Which attribute should be tested here?

\[ S_{\text{Sunny}} = \{D1, D2, D8, D9, D11\} \]

\[ \text{Gain} (S_{\text{Sunny}}, \text{Humidity}) = .970 - \frac{3}{5} \cdot 0.0 - \frac{2}{5} \cdot 0.0 = .970 \]

\[ \text{Gain} (S_{\text{Sunny}}, \text{Temperature}) = .970 - \frac{2}{5} \cdot 0.0 - \frac{2}{5} \cdot 1.0 - \frac{1}{5} \cdot 0.0 = .570 \]

\[ \text{Gain} (S_{\text{Sunny}}, \text{Wind}) = .970 - \frac{2}{5} \cdot 1.0 - \frac{3}{5} \cdot 0.918 = .019 \]

Figure from Tom Mitchell
Binary and Multiclass Classification

Supervised Learning:

$$\mathcal{D} = \{x^{(i)}, y^{(i)}\}_{i=1}^{N} \quad x \sim p^*(\cdot) \text{ and } y = c^*(\cdot)$$

Binary Classification:

$$y^{(i)} \in \{+1, -1\}$$

Multiclass Classification:

$$y^{(i)} \in \{1, \ldots, K\}$$
Outline

Reductions
(Binary → Multiclass)
1. one-vs-all (OVA)
2. all-vs-all (AVA)
3. classification tree
4. error correcting output codes (ECOC)

Settings
A. Multiclass Classification
B. Hierarchical Classification
C. Extreme Classification

Why?
– multiclass is the simplest structured prediction setting
– key insights in the simple reductions are analogous to later (less simple) concepts
REDUCTIONS OF MULTICLASS TO BINARY CLASSIFICATION
Reductions to Binary Classification

Whiteboard:

– Setting for multiclass to binary reductions
– Reduction 1: One-vs-All (OVA)
– Reduction 2: All-vs-All (AVA)
– Reduction 3: Classification Tree
HIERARCHICAL CLASSIFICATION
Hierarchical Classification

Setting:
• Given hierarchy over output labels
• Otherwise, the same as multiclass classification
• Each leaf node is a label
### Hierarchical Classification

**Setting:**
- Given hierarchy over output labels
- Otherwise, the same as multiclass classification
- Each leaf node is a label

**Training Data:** pairs of occupation descriptions and their SOC code
- 9560, Rigging up man
- 5900, Mimeographer
- 3040, Doctor of optometry
- 8310, Wool presser
- 8720, Compress machine operator
- 9640, Pretzel packer
- 9260, Hot box spotter
Hierarchical Classification

Setting:
- Given hierarchy over output labels
- Otherwise, the same as multiclass classification
- Each leaf node is a label
Reductions to Binary Classification

*Whiteboard:*

– Hierarchical classification: how to build an appropriate classifier?
– Features of input vector and label
– Reduction 4: Error Correcting Output Codes (ECOC)
EXTREME CLASSIFICATION
Extreme Classification

Example adapted from Paul Miniero’s ICML 2017 talk
Extreme Classification

**Setting:**
- Output label set is *extremely large* (e.g. millions of labels)
- Otherwise, the *same as multiclass* classification

**Example Tasks:**
- Large-scale facial recognition (billions?)
- Predicting Amazon product categories (3 million)
- Recommending Amazon items (100 million products)
- Predicting Wikipedia tags (2 million)
- Predicting Flick image tags
- Language modeling (millions of words)
Logarithmic-time One-Against-Some

Key idea behind this algorithm:

– build a Recall Tree where
  • each leaf node contains a set $S$ of labels where $|S| \leq \log_2(K)$
  • depth of tree is $d \leq \log_2(K)$
– learn one binary classifier per internal node to route an instance (vector $x$) to a leaf node
– learn one multiclass classifier per leaf over the set of labels $S$ which restricts the label set for instances $x$ routed there
– given a new instance, predict one of the $|S|$ labels at the leaf to which the instance was routed

An example Recall Tree:
Logarithmic-time One-Against-Some

Properties:
1. Competes with one-against-all (i.e. standard multiclass classifier) on benchmark datasets
2. Speed: $O(\log K)$ training and prediction
3. Space: $O(K)$, same as one-against-all
4. Online learning!

An example Recall Tree:
Experiments:

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Task</th>
<th>Classes</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>ALOI[10]</td>
<td>Visual Object Recognition</td>
<td>1k</td>
<td>10^5</td>
</tr>
<tr>
<td>Imagenet[19]</td>
<td>Visual Object Recognition</td>
<td>≈ 20k</td>
<td>≈ 10^7</td>
</tr>
<tr>
<td>LTCH[14]</td>
<td>Language Modeling</td>
<td>≈ 80k</td>
<td>≈ 10^8</td>
</tr>
<tr>
<td>ODP[2]</td>
<td>Document Classification</td>
<td>≈ 100k</td>
<td>≈ 10^6</td>
</tr>
</tbody>
</table>

Figure from Daumé III et al., (2017)
Learning Objectives

From Binary to Multiclass Classification

You should be able to...

1. Reduce the multiclass classification problem to a collection of binary classification problems
2. Identify the advantages and deficiencies of different multiclass-to-binary reductions
3. Implement one-vs-all, all-vs-all, classification tree, error correcting output codes
4. Differentiate multiclass, hierarchical, and extreme classification settings