Admin

- **Assignment 5:**
  - Due in 1 week.

- **Project:**
  - Due date moved again to **April 26** (so that undergrads can graduate).
  - Graduate students graduating in May must submit by April 21.

- No tutorial Friday (or in subsequent weeks).
- Final help session Monday.
- Thursday class may go long.
Motivation: Automatic Brain Tumor Segmentation

- Task: segmentation tumors and normal tissue in multi-modal MRI data.

- Applications:
  - Radiation therapy target planning, quantifying treatment responses.
  - Mining growth patterns, image-guided surgery.

- Challenges:
  - Variety of tumor appearances, similarity to normal tissue.
  - “You are never going to solve this problem.”
Naïve Approach: Voxel-Level Classifier

• We could treat classifying a voxel as supervised learning:

  \[ x^i = \langle 98, 187, 246 \rangle \quad y^i = "\text{tumor}" \]

• “Learn” model that predicts \( y^i \) given \( x^i \): model can classify new voxels.
• Advantage: we can apply machine learning, and ML is cool.
• Disadvantage: it doesn’t work at all.
Naïve Approach: Voxel-Level Classifier

• Even in “nice” cases, significant overlap between tissues:
  – Mixture of Gaussians and “outlier” class:

• Problems with naïve approach:
  – Intensities not standardized.
  – Location and texture matter.
Improvement 1: Intensity Standardization

• Want $x^i = \langle 98, 187, 246 \rangle$ to mean same thing in different places.
• Pre-processing to normalize intensities:

Within Images:  

Between slices:  

Between people:
Improvement 2: Template Alignment

• Location matters:
  – Seeing $x_i = \langle 98, 187, 246 \rangle$ in one area of head is different than in other areas.

• Alignment to **standard coordinates** system:
Improvement 2: Template Alignment

- Add **spatial features** that take into account location information:

  **Aligned input images:**

  ![Aligned input images](image1)

  **Template images:**

  ![Template images](image2)

  **Bilateral symmetry based on known axis:**

  ![Bilateral symmetry](image3)

  **Priors for normal tissue locations:**

  ![Priors](image4)
Improvement 3: Convolutions

• Use convolutions to incorporate neighborhood information.
  – We used fixed convolutions, now you would try to learn them.
Performance of Final System
Challenges

• Final system used linear classifier, and typically worked well.
• But several ML challenges arose:
  1. **Time**: 14 hours to train logistic regression on 10 images.
     • Lead to quasi-Newton, stochastic gradient, and SAG work.
  2. **Overfitting**: using all features hurt, so we used manual feature selection.
     • Lead to regularization, L1-regularization, and structured sparsity work.
  3. **Relaxation**: post-processing by filtering and `hole-filling of labels.
     • Lead to conditional random fields, shape priors, and structure learning work.
Outline

• Motivation
• Conditional Random Fields Clean Up
• Latent/Deep Graphical Models
Multi-Class Logistic Regression: View 1

- Recall that **multi-class logistic regression** makes decisions using:
  \[ \hat{y} = \underset{y \in \{1, 2, \ldots, k\}}{\arg\max} \ w_y^T f(x) \]

- Here, \( f(x) \) are features and we have a vector \( w_y \) for each class ‘\( y \)’.

- Normally fit \( w_y \) using **regularized maximum likelihood** assuming:
  \[ p(y \mid x, w_1, w_2, \ldots, w_k) \propto \exp(w_y^T f(x)) \]

- This **softmax** function yields a differentiable and convex NLL.
Multi-Class Logistic Regression: View 2

• Recall that multi-class logistic regression makes decisions using:

\[ \hat{\gamma} = \arg\max_{\gamma \in \{1, 2, \ldots, K\}} w_{\gamma}^T f(x) \]

• Claim: can be written using a single ‘w’ and features of ‘x’ and ‘y’.

\[ \hat{\gamma} = \arg\max_{\gamma \in \{1, 2, \ldots, K\}} w^T f(x, y) \]

• To do this, we can use the construction:

\[
\begin{bmatrix}
  w_1 \\
  w_2 \\
  \vdots \\
  w_K
\end{bmatrix}
\]

\[ f(x, 1) = \begin{bmatrix}
  f(x) \\
  0 \\
  \vdots \\
  0
\end{bmatrix} \quad f(x, 2) = \begin{bmatrix}
  0 \\
  f(x) \\
  \vdots \\
  0
\end{bmatrix} \quad \text{Implies that} \quad w^T f(x, y) = w_y^T f(x) \]
Multi-Class Logistic Regression: View 2

• So multi-class logistic regression with new notation uses:
  \[ \hat{y} = \arg \max_{y \in \{1, 2, ..., k\}} w^T f(x; y) \]

• And softmax probabilities gives:
  \[ p(y | x, w) = \frac{\exp(w^T f(x; y))}{\sum_{y'} \exp(w^T f(x; y'))} \propto \exp(w^T f(x; y)) \]

• View 2 gives extra flexibility in defining features:
  – For example, we might have different features for class 1 than 2:
  \[
  \begin{align*}
  f(x; 1) &= \begin{bmatrix} f(x) \\ 0 \\ \vdots \\ 0 \end{bmatrix} \\
  f(x; 2) &= \begin{bmatrix} 0 \\ g(x) \\ \vdots \\ 0 \end{bmatrix} \\
  \text{We can even do crazy stuff like;} \\
  f(x; 3) &= \begin{bmatrix} f(x) \\ g(x) \\ \vdots \\ 0 \end{bmatrix}
  \end{align*}
  \]
Multi-Class Logistic Regression for Segmentation

• In brain tumor example, each $x^i$ is the features for one voxel:
  
  Softmax model gives $p(y^i | x^i, w)$ for any label $y^i$ of voxel $i$.

• But we want to label the whole image:

![Diagram showing independent models for each voxel]

• Probability of segmentation $Y$ given image $X$ with independent model:

$$p(Y | X, w) = \prod_{i=1}^{n} p(y^i | x^i, w).$$
Conditional Random Fields

• Unfortunately, independent model gives silly results:

  ![Images of brain scans with and without labels]

• This model of $p(Y|X,w)$ misses the “guilt by association”:
  – Neighbouring voxels are likely to receive the same values.

• The key ingredients of conditional random fields (CRFs):
  – Define features of entire image and labelling $F(X,Y)$:
  – We can model dependencies using features that depend on multiple $y_i$. 
Conditional Random Fields

• Interpretation of independent model as CRF:

\[
p(Y | X, w) = \prod_{i=1}^{n} p(y_i | x_i, w) \propto \prod_{i=1}^{n} \exp(w^T f(x_i, y_i))
\]

\[
= \exp(\sum_{i=1}^{n} w^T f(x_i, y_i))
\]

\[
= \exp(W^T F(X, Y))
\]

(Using same ‘w’ for all ‘i’ is called parameter tying)

\[
W = \begin{bmatrix}
w \\
w \\
w \\
\vdots \\
w
\end{bmatrix}
\]

\[
F(X, Y) = \begin{bmatrix}
f(x_1, y_1) \\
f(x_2, y_2) \\
f(x_3, y_3) \\
\vdots \\
f(x_n, y_n)
\end{bmatrix}
\]
Conditional Random Fields

Example of modeling dependencies between neighbours as a CRF:

\[ p(Y | X, W) \propto \exp(W^T f(X, Y)) \]

We always condition on \( X \), so \( y_i \) features can depend on any part of \( X \).

Usually we'll use different parameters for dependency features.
Conditional Random Fields for Segmentation

• Recall the performance with the independent classifier:
  – Features of the form $f(X,y^i)$.

![Image of brain scan and corresponding output]

• Consider a CRF that also has pairwise features:
  – Features $f(X,y^i,y^j)$ for all $(i,j)$ corresponding to adjacent voxels.
  – Model “guilt by association”: 

![Image of brain scan and corresponding output]
Conditional Random Fields as Graphical Models

• Seems great: we can now model dependencies in the labels.
  – Why not model three-way interactions with $F(X, y^i, y^j, y^k)$?
  – How about adding things like shape priors $F(X, Y_r)$ for some region ‘r’?

• Challenge is that inference and decoding can become hard.

• We can view CRFs as undirected graphical models:

  \[ p(Y \mid X, w) \propto \prod_{c \in C} \phi_c(Y_c) \quad \text{where we have a potential } \phi_c(Y_c) \]

  \[ \text{if } Y_c \text{ appear together in one or more features in } F(X, Y_c) \]

• If the graph is too complicated (and we don’t have special ‘F’):
  – Intractable since we need inference (computing $Z$/marginals) for training.
Overview of Exact Methods for Graphical Models

- We can do **exact decoding/inference/sampling** for:
  - Small number of variables (enumeration).
  - Chains (Viterbi, forward-backward, forward-filter backward-sample).
  - Trees (belief propagation).
  - Low treewidth graphs (junction tree).

- Other cases where **exact** computation is possible:
  - Semi-Markov chains (allow dependence on time you spend in a state).
  - Context-free grammars (allows potentials on recursively-nested parts of sequence).
  - Binary ‘k’ and “attractive” potentials (exact decoding via graph cuts).
  - Sum-product networks (restrict potentials to allow exact computation).
Overview of Approximate Methods for Graphical Models

• Approximate **decoding with local search:**
  – Coordinate descent is called iterated conditional mode (ICM).

• Approximate **sampling with MCMC:**
  – We saw Gibbs sampling last week.

• Approximate **inference with variational methods:**
  – Mean field, loopy belief propagation, tree-reweighted belief propagation.

• Approximate decoding with **convex relaxations:**
  – Linear programming approximation.

• **Block versions** of all of the above:
  – Variant is **alpha-expansions:** block moves involving classes.
Overview of Methods for Fitting Graphical Models

• If inference is intractable, there are some alternatives for learning:
  – Variational inference to approximate Z and marginals.
  – Structured SVMs: generalization of SVMs that only requires decoding.
  – Younes: alternate between Gibbs sampling and parameter update.
    • Also known as “persistent contrastive divergence”.

• For more details on graphical models, see:
  – MLRG PGM crash course: http://www.cs.ubc.ca/labs/lci/mlrg
Independent Logistic Regression

\[
\begin{align*}
\text{Labels} & \quad (w^T h_3(X)) \\
\text{Fixed features} & \quad (\text{convolutions: } w^T X) \\
\text{Input} & \quad w^T X
\end{align*}
\]
Independent Logistic Regression

The $y_i$ are treated as random variables, the $x_i$ and $h_i$ are deterministic.

We are linear combinations of linear combinations so the $h$ don't increase expressive power of model.

But since $h$ are convolutions they might be doing some compression of input.
Conditional Random Field (CRF)

UGM on random variables $y^i$ allows it to model dependencies.

Looks weird to have "directed" parents, but observed parents don't introduce dependencies.
Conditional Random Field (CRF)

The edges can depend on any/all $h^i$ but we'll ignore these edges in our diagrams.
Neural Network

Alternative to fixed features is to learn the features.
- More general than any fixed set, but non-convex.
- The $h^i$ are not observed, but we treat them as deterministic transformations and not random variables.
- To avoid "linear combination of linear combination", apply non-linear transform of $h^i$ (sigmoid, tanh, ReLU).
Deep Learning

\[ h_1 \rightarrow g_1 \rightarrow y_1 \]
\[ h_2 \rightarrow g_2 \rightarrow y_2 \]
\[ h_3 \rightarrow g_3 \rightarrow y_3 \]
\[ h_4 \rightarrow g_4 \rightarrow y_4 \]
\[ h_5 \rightarrow g_5 \rightarrow y_5 \]

\{ We're not explicitly modeling dependence between \( y^1 \) in output. \}

\{ Adding more layers increases expressive power. \}

Do we have to choose between deep learning and CRFs?
Conditional Neural Field (CNF)

CNFs use deep learning features fed into CRF.
- Not convex but can be trained jointly.
- Because $g_i^i$ and $h_i^i$ are deterministic transforms, does not increase complexity of inference.
1. Forward propagation to get $y_i$.  
2. Forward-backward to $\nabla \log(\mathcal{L})$.  
3. Backpropagation to get gradient.
Outline

• Motivation
• Conditional Random Fields Clean Up
• Latent/Deep Graphical Models
Motivation: Gesture Recognition

• Want to recognize gestures from video:

• A gesture is composed of a sequence of parts:
  – Some parts appear in different gestures.

• We have gesture (sequence) labels:
  – But no part labels.
  – We don’t know what the parts should be.
Given a particular gesture, we can model video using an HMM:

Discrete latent \( h \) is the "part" at time \( i \):
- we learn \( p(x^i|h^i) \) and \( p(h^i|h^{i-1}) \) for the gesture.

\( x^i \) is image at time \( i \)!
Generative HMM Classifier

We can use the HMM with a generative classifier:

\[ p(y|x) = p(y)p(x|y) = p(y) \sum_{h_1} \sum_{h_2} \cdots \sum_{h_n} p(x, h_1, h_2, \ldots, h_n) \]

- Correctly models that \( y \) is defined by sequence of parts.
- Inference is fast (treewidth=2)
- But we assumed video frames are independent given part, and even with this modeling \( p(x_i|h_i) \) is hard.
Conditional Random Field (CRF)

Let's use a CRF instead:
- Treat $X$ as fixed so we don't need to model it.
- But CRFs are supervised and we don't see the $h_i$. 
Hidden Conditional Random Field (HCRF)

A UGM that includes:
- temporal dependence between parts;
- dependence of gesture \( y \) on sequence of parts.

Called "hidden" CRF because we observe \( y \) but not the \( h_i \) during training.

Tree width = 2 \( \Rightarrow \) Inference is easy.
Graphical Models with Hidden Variables

• As before we deal with hidden variables by marginalizing:

\[ p(y|X) = \sum_{h_1} \sum_{h_2} \ldots \sum_{h_n} p(y, h|X) \]

• If we assume a UGM over \{Y, H\} given X we get:

\[
p(y|X) = \sum_{H} \frac{\prod_{c \in c} \psi_c(y, h)}{Z(Y)} = \frac{\sum_{H} \prod_{c \in c} \psi_c(y, h)}{Z} = \frac{Z(Y)}{Z}
\]

• Consider usual choice of log-linear phi:
  
  – NLL = -log(Z(Y)) + log(Z).
Motivation: Gesture Recognition

• What if we want to label video with multiple potential gestures?
  – Assume we have labeled video sequences.

Latent-Dynamic Conditional Random Field (LDCRF)

$y^1$ $y^2$ $y^3$ $y^4$ $y^5$

$h^1$ $h^2$ $h^3$ $h^4$ $h^5$

$x^1$ $x^2$ $x^3$ $x^4$ $x^5$

\[ \begin{align*} x_1 & = h_1 \\ h_1 & = 1_{\{x_1 \leq C\}} \\ \epsilon_1 & = 1_{\{x_1 \leq h_1\}} \\ & \vdots \end{align*} \]

\[ \begin{align*} y^i & \text{ Label } y^i \text{ for each time:} \\ h^i & \text{ Capture "latent dynamics" - usually each } y^i \text{ associated with possible hidden states.} \]
Latent-Dynamic Conditional Neural Field (LDCNF)
Summary

- **Conditional random fields** generalize logistic regression:
  - Allows dependencies between labels.
  - Requires inference in graphical model.
- **Conditional neural fields** combine CRFs with deep learning.
  - Could also replace CRF with conditional density estimators (e.g., DAGs).
- **UGMs with hidden variables** have nice form: ratio of normalizers.
  - Can do inference with same methods.
- **Latent dynamic** conditional random/neural fields:
  - Allow dependencies between hidden variables.

- Next time: Boltzmann machines, LSTMs, and beyond CPSC 540.