Part II: Structured Prediction

Alex Schwing & Raquel Urtasun

University of Toronto

December 12, 2015
Roadmap Towards Learning Deep Structured Models

1. Part I: Deep Models

2. Part II: Structured Models

3. Part III: Deep Structured Models
Part II: Structured Prediction
How to predict in structured spaces?
How to learn in structured spaces?
How to predict in structured spaces?
Prediction answers a Yes-No question

\[ x \rightarrow y \in \{\text{Yes, No}\} \]
Binary Classification

Prediction answers a Yes-No question

- Spam filtering (Is this eMail spam?)

Subject: ICCV Tutorial
Text: Hey Raquel, ...

\[ x \rightarrow y \in \{\text{Yes, No}\} \]
Binary Classification

Prediction answers a Yes-No question

- Spam filtering *(Is this eMail spam?)*
- Quality control *(Can I sell this product?)*

\[ x \rightarrow y \in \{\text{Yes}, \text{No}\} \]
Binary Classification

Prediction answers a Yes-No question

- Spam filtering (Is this eMail spam?)
- Quality control (Can I sell this product?)
- Medical testing (Does this lab analysis look normal?)

\[ x \rightarrow y \in \{ \text{Yes, No} \} \]
Binary Classification

Prediction answers a Yes-No question

- Spam filtering (Is this eMail spam?)
- Quality control (Can I sell this product?)
- Medical testing (Does this lab analysis look normal?)
- Driver assistance systems (Is there an obstacle upfront?)

\[ x \rightarrow y \in \{\text{Yes, No}\} \]
Binary Classification

How to find $y \in \{\text{Yes, No}\}$ given input data $x$?
Binary Classification

How to find $y \in \{\text{Yes}, \text{No}\}$ given input data $x$?
Binary Classification

How to find $y \in \{\text{Yes, No}\}$ given input data $x$?
Binary Classification

How to find $y \in \{\text{Yes, No}\}$ given input data $x$?
Binary Classification

How to find $y \in \{\text{Yes, No}\}$ given input data $x$?
Binary Classification

How to find $y \in \{\text{Yes, No}\}$ given input data $x$?
Binary Classification

How to find $y \in \{\text{Yes, No}\}$ given input data $x$?
Binary Classification

How to find $y \in \{\text{Yes, No}\}$ given input data $x$?
Binary Classification

How to find $y \in \{\text{Yes, No}\}$ given input data $x$?
Binary Classification

How to find $y \in \{\text{Yes, No}\}$ given input data $x$?
Linear Discriminant

\[ y \in \{-1, 1\} \]

Equivalent:

\[ y^* = \text{sign}(w^\top x) \]

\[ y^* = \arg \max_{\hat{y}} [w_1 - w_2]^\top [x_\delta(\hat{y} = 1) - x_\delta(\hat{y} = 2)] \]
Linear Discriminant

\[ y \in \{-1, 1\} \]

\[ x_1, x_2, w_1^T x = 0 \]
Linear Discriminant

\[ y \in \{-1, 1\} \]

\[ y^* = \text{sign}(w_1^T x) \]
Linear Discriminant

\[ y \in \{ -1, 1 \} \]

\[ y^* = \text{sign}(w_1^T x) \]

Equivalent:

\[ y \in \{ 1, 2 \} \]

\[ y^* = \arg \max_{\hat{y}} \begin{bmatrix} w_1 \\ -w_1 \end{bmatrix}^T \begin{bmatrix} x_\delta(\hat{y} = 1) \\ x_\delta(\hat{y} = 2) \end{bmatrix} \]
Non-linear Discriminant

\[ y \in \{1, 2\} \]
Non-linear Discriminant

\[ y \in \{1, 2\} \]
Non-linear Discriminant

\[ y \in \{1, 2\} \]

\[ y^* = \arg \max_{\hat{y}} w^T \phi(x, \hat{y}) \]
Non-linear Discriminant

\[ y \in \{1, 2\} \]

\[ y^* = \arg \max_{\hat{y}} w^T \phi(x, \hat{y}) \]

More generally:

\[ y^* = \arg \max_{\hat{y}} F(\hat{y}, x, w) \]
Multiclass Classification

Prediction answers a categorical question

\[ x \rightarrow y \in \{1, \ldots, K\} \]
Multiclass Classification

Prediction answers a categorical question

- Object classification (Which object is in the image?)

\[ x \rightarrow y \in \{1, \ldots, K\} \]
Multiclass Classification

Prediction answers a categorical question

- Object classification (Which object is in the image?)
- Document retrieval (What topic is this document about?)

\[ x \rightarrow y \in \{1, \ldots, K\} \]
Multiclass Classification

Prediction answers a categorical question

- Object classification (Which object is in the image?)
- Document retrieval (What topic is this document about?)
- Medical testing (Which disease fits the symptoms?)

\[ x \rightarrow y \in \{1, \ldots, K\} \]
1 vs all

How to deal with more than two classes?

Use $K-1$ classifiers, each solving a two class problem of separating a point in class $C_k$ from points not in the class.

Known as 1 vs all or 1 vs the rest classifier

Issue: more than one good answer
How to deal with more than two classes?

- Use $K - 1$ classifiers, each solving a two class problem of separating a point in class $C_k$ from points not in the class.
- Known as 1 vs all or 1 vs the rest classifier

Issue: more than one good answer
1 vs 1 classifier

How to deal with more than two classes?

Introduce $K \choose 2$ two-way classifiers, one for each possible pair of classes. Each point is classified according to majority vote amongst the discriminant function. Known as the 1 vs 1 classifier.

Issue: two-way preferences need not be transitive.
1 vs 1 classifier

How to deal with more than two classes?

- Introduce \( K(K - 1)/2 \) two-way classifiers, one for each possible pair of classes
- Each point is classified according to majority vote amongst the discriminant function
- Known as the **1 vs 1 classifier**

Issue: two-way preferences need not be transitive
Score Maximization

\[ y \in \{1, 2, \ldots, K\} \]

Prediction/Inference:

\[ y^* = \arg \max_{\hat{y}} \begin{bmatrix} w_1 \\ \vdots \\ w_K \end{bmatrix} ^\top \begin{bmatrix} x\delta(\hat{y} = 1) \\ \vdots \\ x\delta(\hat{y} = K) \end{bmatrix} \]

More Generally:

\[ y^* = \arg \max_{\hat{y}} w^\top \phi(x, \hat{y}) \]

Even more generally:

\[ y^* = \arg \max_{\hat{y}} F(\hat{y}, x, w) \]

Try all possible classes and return the highest scoring one.
Score Maximization

\[ y \in \{1, 2, \ldots, K\} \]

Prediction/Inference:

\[
y^* = \arg \max_{\hat{y}} \begin{bmatrix} w_1 \vdots w_K \end{bmatrix}^\top \begin{bmatrix} x\delta(\hat{y} = 1) \\ \vdots \\ x\delta(\hat{y} = K) \end{bmatrix}
\]

More Generally:

\[
y^* = \arg \max_{\hat{y}} w^\top \phi(x, \hat{y})
\]

Even more generally:
\[ y \in \{1, 2, \ldots, K\} \]

Prediction/Inference:

\[
y^* = \arg \max_{\hat{y}} \begin{bmatrix} w_1 \\ \vdots \\ w_K \end{bmatrix}^\top \begin{bmatrix} x \delta(\hat{y} = 1) \\ \vdots \\ x \delta(\hat{y} = K) \end{bmatrix}
\]

More Generally:

\[
y^* = \arg \max_{\hat{y}} w^\top \phi(x, \hat{y})
\]

Even more generally:

\[
y^* = \arg \max_{\hat{y}} F(\hat{y}, x, w)
\]
Score Maximization

\[ y \in \{1, 2, \ldots, K\} \]

Prediction/Inference:

\[ y^* = \arg \max_{\hat{y}} \left[ \begin{array}{c} w_1 \\ \vdots \\ w_K \end{array} \right]^\top \left[ \begin{array}{c} x \delta(\hat{y} = 1) \\ \vdots \\ x \delta(\hat{y} = K) \end{array} \right] \]

More Generally:

\[ y^* = \arg \max_{\hat{y}} w^\top \phi(x, \hat{y}) \]

Even more generally:

\[ y^* = \arg \max_{\hat{y}} F(\hat{y}, x, w) \]

Try all possible classes and return the highest scoring one.
Relationship not explicitly taken into account.
Relationship not explicitly taken into account.
Z
Relationship not explicitly taken into account.
Structured Prediction

Q V I Z
Q V I Z
Q V I Z
Structured Prediction

Q V I Z

Q V I Z

Q V I Z

Q U I Z
Relationship not explicitly taken into account.
Structured Prediction

Example: Disparity map estimation

Why not to predict every variable separately:

Image Independent Prediction Structured Prediction
Structured Prediction

Prediction estimates a complex object

\[ x \rightarrow y = (y_1, \ldots, y_D) \]
Structured Prediction

Prediction estimates a complex object

- Image segmentation (estimate a labeling)

\[ x \rightarrow y = (y_1, \ldots, y_D) \]
Structured Prediction

Prediction estimates a complex object

- Image segmentation (estimate a labeling)
- Sentence parsing (estimate a parse tree)

\[ x \rightarrow y = (y_1, \ldots, y_D) \]

I saw the man with the telescope.
Structured Prediction

Prediction estimates a complex object

- Image segmentation (estimate a labeling)
- Sentence parsing (estimate a parse tree)
- Protein folding (estimate a protein structure)

\[ x \rightarrow y = (y_1, \ldots, y_D) \]
Structured Prediction

Prediction estimates a complex object

- Image segmentation (estimate a labeling)
- Sentence parsing (estimate a parse tree)
- Protein folding (estimate a protein structure)
- Stereo vision (estimate a disparity map)

\[ x \rightarrow y = (y_1, \ldots, y_D) \]
“Standard” Prediction: output $y \in \mathcal{Y}$ is a scalar number

$$\mathcal{Y} = \{1, \ldots, K\} \quad \text{or} \quad \mathcal{Y} = \mathbb{R}$$
Structured Prediction

“Standard” Prediction: output $y \in \mathcal{Y}$ is a scalar number

$$\mathcal{Y} = \{1, \ldots, K\} \quad \text{or} \quad \mathcal{Y} = \mathbb{R}$$

“Structured” Prediction: output $y$ is a structured output:
Structured Prediction

Formally:

\[ y = (y_1, \ldots, y_D) \quad y_i = \{1, \ldots, K\} \]
Structured Prediction

Formally:

\[ y = (y_1, \ldots, y_D) \quad y_i = \{1, \ldots, K\} \]

Inference/Prediction:

\[ y^* = \arg \max_{\hat{y}} F(\hat{y}_1, \ldots, \hat{y}_D, x, w) \]

How many possibilities do we have to store and explore?
Structured Prediction

Formally:
\[ y = (y_1, \ldots, y_D) \quad y_i = \{1, \ldots, K\} \]

Inference/Prediction:
\[ y^* = \arg \max_{\hat{y}} F(\hat{y}_1, \ldots, \hat{y}_D, x, w) \]

How many possibilities do we have to store and explore?
\[ K^D \]
Structured Prediction

Formally:

\[ \mathbf{y} = (y_1, \ldots, y_D) \quad y_i = \{1, \ldots, K\} \]

Inference/Prediction:

\[ \mathbf{y}^* = \arg \max_{\mathbf{\hat{y}}} F(\mathbf{\hat{y}}_1, \ldots, \mathbf{\hat{y}}_D, x, w) \]

How many possibilities do we have to store and explore?

\[ K^D \]

That’s a problem. What can we do?
Structured Prediction

Separate prediction:

$$\max_{y} F(y_1, \ldots, y_D, x, w) = \sum_{i=1}^{D} \max_{y_i} f_i(y_i, x, w)$$

Why not predict every variable $y_i$ from $y = (y_1, \ldots, y_D)$ separately?
Separate prediction:

$$\max_{\mathbf{y}} F(y_1, \ldots, y_D, x, w) = \sum_{i=1}^{D} \max_{y_i} f_i(y_i, x, w)$$

Why not predict every variable $y_i$ from $\mathbf{y} = (y_1, \ldots, y_D)$ separately?

Relationship not explicitly taken into account.
Structured Prediction

Discriminant function decomposes:

\[ F(y_1, \ldots, y_D, x, w) = \sum_r f_r(y_r, x, w) \]

Restriction: every \( r \subseteq \{1, \ldots, D\} \)
Structured Prediction

Discriminant function decomposes:

\[ F(y_1, \ldots, y_D, x, w) = \sum_r f_r(y, x, w) \]

Restriction: every \( r \subseteq \{1, \ldots, D\} \)

Discrete domain:

\[ f_{\{1,2\}}(y_{\{1,2\}}) = f_{\{1,2\}}(y_1, y_2) = [f_{\{1,2\}}(1, 1), f_{\{1,2\}}(1, 2), \ldots] \]
Structured Prediction

Discriminant function decomposes:

\[ F(y_1, \ldots, y_D, x, w) = \sum_r f_r(y_r, x, w) \]

Restriction: every \( r \subseteq \{1, \ldots, D\} \)

Discrete domain:

\[ f_{\{1,2\}}(y_{\{1,2\}}) = f_{\{1,2\}}(y_1, y_2) = [f_{\{1,2\}}(1, 1), f_{\{1,2\}}(1, 2), \ldots] \]

\[
\begin{array}{c|cccccc}
 & Q & U & I & Z & V \\
\hline
Q & 0 & 0.8 & 0.2 & 0.1 & 0.1 \\
U & \ddots & \ddots & \ddots & \ddots & \ddots \\
I & : & : & : & : & : \\
Z & \vdots & \vdots & \vdots & \vdots & \vdots \\
V & \vdots & \vdots & \vdots & \vdots & \vdots \\
\end{array}
\]
Example:

\[ F(y_1, \ldots, y_4, x, w) = f_1(y_1, x, w) + f_2(y_2, x, w) + f_3(y_3, x, w) + f_4(y_4, x, w) \\
+ f_{1,2}(y_1, y_2, x, w) + f_{2,3}(y_2, y_3, x, w) + f_{3,4}(y_3, y_4, x, w) \]
Structured Prediction

Example:

\[ F(y_1, \ldots, y_4, x, w) = f_1(y_1, x, w) + f_2(y_2, x, w) + f_3(y_3, x, w) + f_4(y_4, x, w) \]
\[ + f_{1,2}(y_1, y_2, x, w) + f_{2,3}(y_2, y_3, x, w) + f_{3,4}(y_3, y_4, x, w) \]

How many function values need to be stored if \( y_i \in \{1, \ldots, 26\} \ \forall i? \]
Structured Prediction

Example:

\[ F(y_1, \ldots, y_4, x, w) = f_1(y_1, x, w) + f_2(y_2, x, w) + f_3(y_3, x, w) + f_4(y_4, x, w) + f_{1,2}(y_1, y_2, x, w) + f_{2,3}(y_2, y_3, x, w) + f_{3,4}(y_3, y_4, x, w) \]

How many function values need to be stored if \( y_i \in \{1, \ldots, 26\} \ \forall i? \]

\[ 26^4 \quad \text{v.s.} \quad 3 \cdot 26^2 (+4 \cdot 26) \]
Structured Prediction

Visualization of the decomposition:

\[ F(y_1, \ldots, y_4, x, w) = f_1(y_1, x, w) + f_2(y_2, x, w) + f_3(y_3, x, w) + f_4(y_4, x, w) + f_{1,2}(y_1, y_2, x, w) + f_{2,3}(y_2, y_3, x, w) + f_{3,4}(y_3, y_4, x, w) \]
Structured Prediction

Visualization of the decomposition:

\[ F(y_1, \ldots, y_4, x, w) = f_1(y_1, x, w) + f_2(y_2, x, w) + f_3(y_3, x, w) + f_4(y_4, x, w) \]
\[ + f_{1,2}(y_1, y_2, x, w) + f_{2,3}(y_2, y_3, x, w) + f_{3,4}(y_3, y_4, x, w) \]

Edges denote subset relationship
Structured Prediction

Special cases
Predicting every variable separately:

\[
F(y_1, \ldots, y_D, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w)
\]
Structured Prediction

Special cases
Predicting every variable separately:

\[ F(y_1, \ldots, y_D, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w) \]

\[ y_1 \quad y_2 \quad \ldots \]
Special cases
Predicting every variable separately:

\[ F(y_1, \ldots, y_D, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w) \]

Markov random field with only unary variables
Structured Prediction

Special cases
Predicting every variable separately:

\[ F(y_1, \ldots, y_D, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w) \]

Markov random field with only unary variables

Multi-variate prediction:

\[ F(y_1, \ldots, y_D, x, w) = f_{1,...,D}(y_1,\ldots,D, x, w) \]
Structured Prediction

Special cases
Predicting every variable separately:

\[
F(y_1, \ldots, y_D, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w)
\]

Markov random field with only unary variables

Multi-variate prediction:

\[
F(y_1, \ldots, y_D, x, w) = f_{1,\ldots,D}(y_1,\ldots,y_D, x, w)
\]
Structured Prediction

Example: stereo vision

Markov/Conditional random field:

$$F(y, x, w) = D \sum_{i=1}^{N} f_i(y_i, x, w) + \sum_{i,j} f_{i,j}(y_i, y_j, x, w)$$

- **Unary term:** image evidence
- **Pairwise term:** smoothness prior
Structured Prediction

Example: stereo vision

Markov/Conditional random field:

\[ F(y, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w) + \sum_{i,j} f_{i,j}(y_i, y_j, x, w) \]

- Unary term: image evidence
- Pairwise term: smoothness prior
Structured Prediction

Example: stereo vision

Markov/Conditional random field:

\[ F(y, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w) + \sum_{i,j} f_{i,j}(y_i, y_j, x, w) \]

- Unary term:
- Pairwise term:
Structured Prediction

Example: stereo vision

Markov/Conditional random field:

$$F(y, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w) + \sum_{i,j} f_{i,j}(y_i, y_j, x, w)$$

- Unary term: image evidence
- Pairwise term:
Structured Prediction

Example: stereo vision

Markov/Conditional random field:

\[
F(y, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w) + \sum_{i,j} f_{i,j}(y_i, y_j, x, w)
\]

- Unary term: image evidence
- Pairwise term: smoothness prior
Structured Prediction

Example: semantic segmentation

Markov/Conditional random field:

\[ F(\mathbf{y}, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w) + \sum_{i,j} f_{i,j}(y_i, y_j, x, w) \]
Structured Prediction

Example: semantic segmentation

Markov/Conditional random field:

\[ F(y, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w) + \sum_{i,j} f_{i,j}(y_i, y_j, x, w) \]

- **Unary term:**
- **Pairwise term:**
Structured Prediction

Example: semantic segmentation

Markov/Conditional random field:

\[
F(y, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w) + \sum_{i,j} f_{i,j}(y_i, y_j, x, w)
\]

- Unary term: image evidence
- Pairwise term:
Structured Prediction

Example: semantic segmentation

Markov/Conditional random field:

\[
F(y, x, w) = \sum_{i=1}^{D} f_i(y_i, x, w) + \sum_{i,j} f_{i,j}(y_i, y_j, x, w)
\]

- Unary term: image evidence
- Pairwise term: smoothness prior
Structured Prediction

Inference:

\[ y^* = \arg \max_{\hat{y}} F(\hat{y}, x, w) \]
Structured Prediction

Inference:

\[ \mathbf{y}^* = \arg \max_{\hat{\mathbf{y}}} F(\hat{\mathbf{y}}, x, w) \]

Probability of a configuration \( \mathbf{y} \):

\[ p(\mathbf{y} \mid x, w) = \frac{1}{Z(x, w)} \exp F(\mathbf{y}, x, w) \]
Structured Prediction

Inference:

\[ y^* = \arg \max_{\hat{y}} F(\hat{y}, x, w) \]

Probability of a configuration \( y \):

\[ p(y | x, w) = \frac{1}{Z(x, w)} \exp F(y, x, w) \]

Normalization constant/partition function:

\[ Z(x, w) = \sum_{y} \exp F(y, x, w) \]
Structured Prediction

Inference:
\[ y^* = \arg \max_{\hat{y}} F(\hat{y}, x, w) \]

Probability of a configuration \( y \):
\[ p(y \mid x, w) = \frac{1}{Z(x, w)} \exp F(y, x, w) \]

Normalization constant/partition function:
\[ Z(x, w) = \sum_y \exp F(y, x, w) \]

Inference as probability maximization:
\[ y^* = \arg \max_{\hat{y}} p(\hat{y} \mid x, w) \]
Structured Prediction

Inference:
\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r, x, w) \]

Some inference algorithms:

- Exhaustive search
- Dynamic programming
- Integer linear program
- Linear programming relaxation
- Message passing
- Graph-cut
Structured Prediction

Inference:

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r, x, w) \]

Some inference algorithms:

- Exhaustive search
- Dynamic programming
- Integer linear program
- Linear programming relaxation
- Message passing
- Graph-cut
\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r, x, w) \]
Structured Prediction - Exhaustive Search

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r, x, w) \]

Algorithm:

- try all possible configurations \( \hat{y} \in \mathcal{Y} \)
- keep highest scoring element
Structured Prediction - Exhaustive Search

\[ y^* = \arg\max_{\hat{y}} \sum_{r} f_r(\hat{y}_r, x, w) \]

Algorithm:
- try all possible configurations \( \hat{y} \in \mathcal{Y} \)
- keep highest scoring element

**Advantage:** very simple to implement

**Disadvantage:** very slow for reasonably sized problems
\[
\max_{\hat{\mathbf{y}}} F(\hat{\mathbf{y}}, \mathbf{x}, w) = \max_{\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2} f_2(\hat{\mathbf{y}}_2) + f_1(\hat{\mathbf{y}}_1) + f_{1,2}(\hat{\mathbf{y}}_1, \hat{\mathbf{y}}_2)
\]
\[
\begin{align*}
\max_{\hat{y}} F(\hat{y}, x, w) &= \max_{\hat{y}_1, \hat{y}_2} f_2(\hat{y}_2) + f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2) \\
&= \max_{\hat{y}_2} f_2(\hat{y}_2) + \max_{\hat{y}_1} \{ f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2) \}
\end{align*}
\]
\[
\max F(\hat{y}, x, w) = \max_{\hat{y}_1, \hat{y}_2} f_2(\hat{y}_2) + f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2)
\]

\[
= \max_{\hat{y}_2} f_2(\hat{y}_2) + \max_{\hat{y}_1} \left\{ f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2) \right\}
\]

\[
\mu_{1,2 \rightarrow 2}(\hat{y}_2)
\]
\[
\max_{\hat{y}} F(\hat{y}, x, w) = \max_{\hat{y}_1, \hat{y}_2} f_2(\hat{y}_2) + f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2) \\
= \max_{\hat{y}_2} f_2(\hat{y}_2) + \max_{\hat{y}_1} \left\{ f_1(\hat{y}_1) + f_{1,2}(\hat{y}_1, \hat{y}_2) \right\} \\
= \max_{\hat{y}_2} f_2(\hat{y}_2) + \mu_{1,2\rightarrow 2}(\hat{y}_2)
\]
We can reorganize terms whenever the graph is a **tree**.

What to do for general loopy graphs?
We can reorganize terms whenever the graph is a \textbf{tree}.

What to do for general loopy graphs?

- Dynamic programming extensions (message passing)
- Graph cut algorithms
Structured Prediction - Integer Linear Program

Example:

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r) \]

Integer Linear Program (LP) equivalence: variables \( b_r(y_r) \)

\[
\begin{bmatrix}
    b_1(1) \\
    b_1(2) \\
    b_2(1) \\
    b_2(2) \\
    b_{12}(1, 1) \\
    b_{12}(2, 1) \\
    b_{12}(1, 2) \\
    b_{12}(2, 2)
\end{bmatrix}^T
\begin{bmatrix}
    f_1(1) \\
    f_1(2) \\
    f_2(1) \\
    f_2(2) \\
    f_{12}(1, 1) \\
    f_{12}(2, 1) \\
    f_{12}(1, 2) \\
    f_{12}(2, 2)
\end{bmatrix}
\]

\[
\max_{b_1, b_2, b_{12}}
\]
Structured Prediction - Integer Linear Program

Example:

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r) \]

Integer Linear Program (LP) equivalence: variables \( b_r(y_r) \)

\[
\max_{b_1, b_2, b_{12}} \begin{bmatrix}
    b_1(1) \\
    b_1(2) \\
    b_2(1) \\
    b_2(2) \\
    b_{12}(1, 1) \\
    b_{12}(2, 1) \\
    b_{12}(1, 2) \\
    b_{12}(2, 2)
\end{bmatrix}^\top
\begin{bmatrix}
    f_1(1) \\
    f_1(2) \\
    f_2(1) \\
    f_2(2) \\
    f_{12}(1, 1) \\
    f_{12}(2, 1) \\
    f_{12}(1, 2) \\
    f_{12}(2, 2)
\end{bmatrix}
\]

s.t.

\[
b_r(y_r) \in \{0, 1\}
\]
Structured Prediction - Integer Linear Program

Example:

$$y^* = \arg \max_\hat{y} \sum_r f_r(\hat{y}_r)$$

Integer Linear Program (LP) equivalence: variables $b_r(y_r)$

$$\max_{b_1, b_2, b_{12}} \begin{bmatrix} b_1(1) \\ b_1(2) \\ b_2(1) \\ b_2(2) \\ b_{12}(1, 1) \\ b_{12}(2, 1) \\ b_{12}(1, 2) \\ b_{12}(2, 2) \end{bmatrix}^\top \begin{bmatrix} f_1(1) \\ f_1(2) \\ f_2(1) \\ f_2(2) \\ f_{12}(1, 1) \\ f_{12}(2, 1) \\ f_{12}(1, 2) \\ f_{12}(2, 2) \end{bmatrix}$$

Subject to:

$$b_r(y_r) \in \{0, 1\}$$

$$\sum_{y_r} b_r(y_r) = 1$$
Structured Prediction - Integer Linear Program

Example:

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r) \]

Integer Linear Program (LP) equivalence: variables \( b_r(y_r) \)

\[
\begin{bmatrix}
    b_1(1) \\
    b_1(2) \\
    b_2(1) \\
    b_2(2) \\
    b_{12}(1, 1) \\
    b_{12}(2, 1) \\
    b_{12}(1, 2) \\
    b_{12}(2, 2)
\end{bmatrix}^T
\begin{bmatrix}
    f_1(1) \\
    f_1(2) \\
    f_2(1) \\
    f_2(2) \\
    f_{12}(1, 1) \\
    f_{12}(2, 1) \\
    f_{12}(1, 2) \\
    f_{12}(2, 2)
\end{bmatrix}
\]

s.t.

\[
\begin{align*}
    b_r(y_r) &\in \{0, 1\} \\
    \sum_{y_r} b_r(y_r) &= 1 \\
    \sum_{y_p \setminus y_r} b_p(y_p) &= b_r(y_r)
\end{align*}
\]
Structured Prediction - Integer Linear Program

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r) \]

Integer linear program:

\[
\begin{bmatrix}
    b_1(1) \\
    b_1(2) \\
    b_2(1) \\
    b_2(2) \\
    b_{12}(1, 1) \\
    b_{12}(2, 1) \\
    b_{12}(1, 2) \\
    b_{12}(2, 2)
\end{bmatrix}^T
\begin{bmatrix}
    f_1(1) \\
    f_1(2) \\
    f_2(1) \\
    f_2(2) \\
    f_{12}(1, 1) \\
    f_{12}(2, 1) \\
    f_{12}(1, 2) \\
    f_{12}(2, 2)
\end{bmatrix}
\]

s.t.

- \( b_r(y_r) \in \{0, 1\} \)
- \( b_r(y_r) \geq 0 \)
- \( \sum_{y_r} b_r(y_r) = 1 \)
- \( \sum_{y_p \setminus y_r} b_p(y_p) = b_r(y_r) \)
Structured Prediction - Integer Linear Program

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r) \]

Integer linear program:

\[
\begin{align*}
\max_{b_r} & \quad \sum_{r, y_r} b_r(y_r) f_r(y_r) \\
\text{s.t.} & \quad b_r(y_r) \in \{0, 1\} \\
& \quad b_r(y_r) \geq 0 \\
& \quad \sum_{y_r} b_r(y_r) = 1 \\
& \quad \sum_{y_p \setminus y_r} b_p(y_p) = b_r(y_r)
\end{align*}
\]
Structured Prediction - Integer Linear Program

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r) \]

Integer linear program:

\[
\max_{b_r} \sum_{r, y_r} b_r(y_r) f_r(y_r) \quad \text{s.t.} \quad \sum_{y_r} b_r(y_r) = 1
\]

\[ b_r(y_r) \in \{0, 1\} \]

\[ b_r(y_r) \geq 0 \]

Marginalization
Structured Prediction - Integer Linear Program

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r) \]

Integer linear program:

max \[ \sum_{r, y_r} b_r(y_r)f_r(y_r) \]

s.t. Local probability \( b_r \)

Marginalization

\[ b_r(y_r) \in \{0, 1\} \]
Structured Prediction - Integer Linear Program

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r) \]

Integer linear program:

\[
\max_{b_r} \sum_{r, y_r} b_r(y_r) f_r(y_r) \quad \text{s.t.} \quad \text{Local probability } b_r \\
\text{Marginalization}
\]

- **Advantage:** very good solvers available
- **Disadvantage:** very slow for larger problems
Structured Prediction - Linear Programming Relaxation

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r) \]

LP relaxation:

\[
\begin{align*}
\text{max} \quad & \sum_{r, y_r} b_r(y_r) f_r(y_r) \\
\text{s.t.} \quad & \text{Local probability } b_r \\
& \text{Marginalization} \\
& \text{s.t. } b \in C
\end{align*}
\]
Structured Prediction - Linear Programming Relaxation

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r) \]

LP relaxation:

\[ \max_{b_r} \sum_{r, y_r} b_r(y_r) f_r(y_r) \quad \text{s.t.} \quad \text{Local probability } b_r \]

\[ b_r(y_r) \in \{0, 1\} \]

Marginalization

- **Advantage:** very good solvers available
- **Disadvantage:** slow for larger problems
Graph structure defined via marginalization constraints
Graph structure defined via marginalization constraints

Message passing solvers:

Advantage: Efficient due to analytically computable sub-problems
Problem: Special care required to find global optimum
For $y_i \in \{1, 2\}$:

- Convert scoring function $F$ into auxiliary graph
For $y_i \in \{1, 2\}$:

- Convert scoring function $F$ into auxiliary graph
- Compute a weighted cut cost corresponding to the labeling score
For $y_i \in \{1, 2\}$:

- Convert scoring function $F$ into auxiliary graph
- Compute a weighted cut cost corresponding to the labeling score

What are the nodes and what are the weights on the edges?
What are the nodes?
What are the nodes?

- Two special nodes called ‘source’ and ‘terminal’
What are the nodes?

- Two special nodes called ‘source’ and ‘terminal’
- Variables $y_i$ as nodes
What are the nodes?

- Two special nodes called ‘source’ and ‘terminal’
- Variables $y_i$ as nodes
Structured Prediction - Graph-cut Solvers

What weights do we assign to edges?
Structured Prediction - Graph-cut Solvers

What weights do we assign to edges?
Recall that local scoring functions are arrays:

\[
\begin{bmatrix}
  f_1(y_1 = 1) & f_1(y_1 = 2)
\end{bmatrix}
\]
Structured Prediction - Graph-cut Solvers

What weights do we assign to edges?
Recall that local scoring functions are arrays:

\[
\begin{bmatrix}
  f_1(y_1 = 1) & f_1(y_1 = 2)
\end{bmatrix}
\]

Graph-cut solvers compute a min-cut:
What weights do we assign to edges?
Recall that local scoring functions are arrays:

\[
\begin{bmatrix}
    f_{12}(1, 1) & f_{12}(1, 2) \\
    f_{12}(2, 1) & f_{12}(2, 2)
\end{bmatrix}
\] =
What weights do we assign to edges?
Recall that local scoring functions are arrays:

\[
\begin{bmatrix}
  f_{12}(1,1) & f_{12}(1,2) \\
  f_{12}(2,1) & f_{12}(2,2)
\end{bmatrix}
= f(1,1) - f(2,1) + f(2,2)
\]

\[
+ \begin{bmatrix}
  0 & 0 \\
  f(2,1) - f(1,1) & f(2,1) - f(1,1)
\end{bmatrix}
\]

\[
+ \begin{bmatrix}
  f(2,1) - f(1,1) & f(2,1) - f(1,1) \\
  f(2,1) - f(2,2) & f(2,1) - f(2,2)
\end{bmatrix}
\]

\[
+ \begin{bmatrix}
  0 & f(1,2) + f(2,1) - f(1,1) - f(2,2) \\
  0 & 0
\end{bmatrix}
\]
What weights do we assign to edges?

Recall that local scoring functions are arrays:

\[
\begin{bmatrix}
  f_{12}(1, 1) & f_{12}(1, 2) \\
  f_{12}(2, 1) & f_{12}(2, 2)
\end{bmatrix}
= f(1, 1) - f(2, 1) + f(2, 2)
\]

\[
+ \begin{bmatrix}
  0 & 0 \\
  f(2, 1) - f(1, 1) & f(2, 1) - f(1, 1)
\end{bmatrix}
+ \begin{bmatrix}
  f(2, 1) - f(2, 2) & 0 \\
  f(2, 1) - f(2, 2) & 0
\end{bmatrix}
+ \begin{bmatrix}
  0 & f(1, 2) + f(2, 1) - f(1, 1) - f(2, 2) \\
  0 & 0
\end{bmatrix}
\]

Graph-cut solvers compute a min-cut:
What weights do we assign to edges?
Recall that local scoring functions are arrays:

\[
\begin{bmatrix}
  f_{12}(1,1) & f_{12}(1,2) \\
  f_{12}(2,1) & f_{12}(2,2)
\end{bmatrix} = f(1,1) - f(2,1) + f(2,2)
\]

\[
+ \begin{bmatrix}
  0 & 0 \\
  f(2,1) - f(1,1) & f(2,1) - f(1,1)
\end{bmatrix}
\]

\[
+ \begin{bmatrix}
  f(2,1) - f(2,2) & 0 \\
  f(2,1) - f(2,2) & 0
\end{bmatrix}
\]

\[
+ \begin{bmatrix}
  0 & f(1,2) + f(2,1) - f(1,1) - f(2,2) \\
  0 & 0
\end{bmatrix}
\]

Graph-cut solvers compute a min-cut:
What weights do we assign to edges?
Recall that local scoring functions are arrays:

\[
\begin{bmatrix}
  f_{12}(1, 1) & f_{12}(1, 2) \\
  f_{12}(2, 1) & f_{12}(2, 2)
\end{bmatrix}
= f(1, 1) - f(2, 1) + f(2, 2)
\]

Graph-cut solvers compute a min-cut:
What weights do we assign to edges?
Recall that local scoring functions are arrays:

\[
\begin{bmatrix}
 f_{12}(1, 1) & f_{12}(1, 2) \\
 f_{12}(2, 1) & f_{12}(2, 2)
\end{bmatrix} = f(1, 1) - f(2, 1) + f(2, 2) \\
\]

\[+ \begin{bmatrix}
 0 & 0 \\
 f(2, 1) - f(1, 1) & f(2, 1) - f(1, 1)
\end{bmatrix} + \begin{bmatrix}
 f(2, 1) - f(2, 2) & 0 \\
 f(2, 1) - f(2, 2) & 0
\end{bmatrix} + \begin{bmatrix}
 0 & f(1, 2) + f(2, 1) - f(1, 1) - f(2, 2) \\
 0 & 0
\end{bmatrix}\]

Graph-cut solvers compute a min-cut:
Requirement for optimality:
Structured Prediction - Graph-cut Solvers

Requirement for optimality: Pairwise edge weights are positive

\[ f(1, 1) + f(2, 2) - f(1, 2) - f(2, 1) \geq 0 \] sub-modularity
Structured Prediction - Graph-cut Solvers

\[ f(1, 1) - f(2, 1) \]
\[ f(1, 1) + f(2, 2) - f(1, 2) - f(2, 1) \]
\[ f(2, 2) - f(2, 1) \]

Requirement for optimality: Pairwise edge weights are positive

\[ f(1, 1) + f(2, 2) - f(1, 2) - f(2, 1) \geq 0 \] sub-modularity

For higher order functions?
Structured Prediction - Graph-cut Solvers

Requirement for optimality: Pairwise edge weights are positive

\[ f(1, 1) + f(2, 2) - f(1, 2) - f(2, 1) \geq 0 \quad \text{sub-modularity} \]

For higher order functions? More complicated graph constructions
Structured Prediction - Graph-cut Solvers

**Requirement for optimality:** Pairwise edge weights are positive

\[ f(1, 1) + f(2, 2) - f(1, 2) - f(2, 1) \geq 0 \quad \text{sub-modularity} \]

For higher order functions? More complicated graph constructions
For more than two labels?
Structured Prediction - Graph-cut Solvers

Requirement for optimality: Pairwise edge weights are positive

\[ f(1, 1) + f(2, 2) - f(1, 2) - f(2, 1) \geq 0 \]  sub-modularity

For higher order functions? More complicated graph constructions
For more than two labels? Move making algorithms
Structured Prediction

Inference:

\[ y^* = \arg \max_{\hat{y}} \sum_r f_r(\hat{y}_r, x, w) \]

Efficiency and accuracy of inference algorithms is problem dependent:

- Exhaustive search
- Dynamic programming
- Integer linear program
- Linear programming relaxation
- Message passing
- Graph-cut
How to learn in structured spaces?
Given: dataset

\[ \mathcal{D} = \{(x, y)\}, \quad N = |\mathcal{D}| \]

containing independent and identically drawn pairs
Binary SVM

Intuitively:

Maximize margin $\frac{2}{\|w\|}$:

$$\min_w \frac{1}{2} \|w\|_2^2 \quad \text{s.t.} \quad yw^\top x \geq 1 \quad \forall (x, y) \in \mathcal{D}$$
Binary SVM

Intuitively:

\[ w^T x = 1 \]
\[ w^T x = 0 \]
\[ w^T x = -1 \]

Maximize margin \( \frac{2}{\|w\|} \):

\[
\min_w \frac{1}{2} \|w\|_2^2 \quad \text{s.t.} \quad yw^T x \geq 1 \quad \forall (x, y) \in \mathcal{D}
\]

Issue: what if data not linearly separable?
Binary SVM

Introduce slack variables $\xi$:

$$
\min_{w, \xi} \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{(x, y) \in D} \xi_x \quad \text{s.t.} \quad yw^\top x \geq 1 - \xi_x \quad \forall (x, y) \in D
$$

Intuitively:
\[
\begin{aligned}
\min_{w, \xi_x \geq 0} & \quad \frac{1}{2} \| w \|_2^2 + {\frac{C}{N}} \sum_{(x,y) \in D} \xi_x \\
\text{s.t.} & \quad yw^T x \geq 1 - \xi_x \quad \forall (x, y) \in D
\end{aligned}
\]
Binary SVM

\[
\min_{w, \xi_x \geq 0} \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{(x,y) \in \mathcal{D}} \xi_x \quad \text{s.t.} \quad yw^T x \geq 1 - \xi_x \quad \forall (x, y) \in \mathcal{D}
\]

Equivalent:

\[
\min_w \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{(x,y) \in \mathcal{D}} \max\{0, 1 - yw^T x\}
\]
Binary SVM

\[
\min_{w, \xi_x \geq 0} \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{(x,y) \in D} \xi_x \quad \text{s.t.} \quad yw^T x \geq 1 - \xi_x \quad \forall (x, y) \in D
\]

Equivalent:

\[
\min_w \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{(x,y) \in D} \max\{0, 1 - yw^T x\}
\]

Generally:

\[
\min_w R(w) + \frac{C}{N} \sum_{(x,y) \in D} \bar{\ell}(x, y, w)
\]
Binary SVM

\[
\min_{w, \xi \geq 0} \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x, y) \in D} \xi_x \quad \text{s.t.} \quad yw^T x \geq 1 - \xi_x \quad \forall (x, y) \in D
\]

Equivalent:

\[
\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x, y) \in D} \max\{0, 1 - yw^T x\}
\]

Generally:

\[
\min_w R(w) + \frac{C}{N} \sum_{(x, y) \in D} \bar{\ell}(x, y, w)
\]

Hinge-Loss $\bar{\ell}$:

\[
\max\{0, 1 - t\}
\]
Multiclass SVM

Recap of Inference:

\[ y \in \{1, 2, \ldots, K\} \]

\[
y^* = \arg \max \hat{y} \begin{bmatrix} w_1 & \ldots & w_K \end{bmatrix}^\top \begin{bmatrix} x_{\delta(\hat{y} = 1)} & \ldots & x_{\delta(\hat{y} = K)} \end{bmatrix}
\]

More generally:

\[
y^* = \arg \max \hat{y} w^\top \phi(x, \hat{y})
\]

Even more generally:

\[
y^* = \arg \max \hat{y} F(\hat{y}, x, w)
\]
Recap of Inference:

$$y \in \{1, 2, \ldots, K\}$$

$$y^* = \arg \max_{\hat{y}} \left[ \begin{array}{c} w_1 \\ \vdots \\ w_K \end{array} \right] ^\top \left[ \begin{array}{c} x\delta(\hat{y} = 1) \\ \vdots \\ x\delta(\hat{y} = K) \end{array} \right]$$

More generally:

$$y^* = \arg \max_{\hat{y}} F(\hat{y}, x, w)$$
Recap of Inference:

\[ y \in \{1, 2, \ldots, K\} \]

\[ y^* = \arg \max_{\hat{y}} \begin{bmatrix} w_1^\top \\ \vdots \\ w_K^\top \end{bmatrix} \begin{bmatrix} x_\delta(\hat{y} = 1) \\ \vdots \\ x_\delta(\hat{y} = K) \end{bmatrix} \]

More generally:

\[ y^* = \arg \max_{\hat{y}} w^\top \phi(x, \hat{y}) \]

Even more generally:
Recap of Inference:

\[ y \in \{1, 2, \ldots, K\} \]

\[ y^* = \arg \max_{\hat{y}} \begin{bmatrix} w_1 \\ \vdots \\ w_K \end{bmatrix}^\top \begin{bmatrix} x\delta(\hat{y} = 1) \\ \vdots \\ x\delta(\hat{y} = K) \end{bmatrix} \]

More generally:

\[ y^* = \arg \max_{\hat{y}} w^\top \phi(x, \hat{y}) \]

Even more generally:

\[ y^* = \arg \max_{\hat{y}} F(\hat{y}, x, w) \]
Multiclass SVM

What do we want?
What do we want? Groundtruth $y$ scores higher than any other $\hat{y}$
What do we want? Groundtruth $y$ scores higher than any other $\hat{y}$

$$w^T \phi(x, y) \geq w^T \phi(x, \hat{y}) \quad \forall (x, y) \in D, \hat{y} \in \{1, \ldots, K\}$$
What do we want? Groundtruth $y$ scores higher than any other $\hat{y}$

$$w^T \phi(x, y) \geq w^T \phi(x, \hat{y}) \quad \forall (x, y) \in \mathcal{D}, \hat{y} \in \{1, \ldots, K\}$$

Let’s employ margin and slack:

$$\min_{w, \xi_x \geq 0} \quad \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{(x, y)} \xi_x$$

$$\text{s.t.} \quad w^T (\phi(x, y) - \phi(x, \hat{y})) \geq 1 - \xi_x \quad \forall (x, y), \hat{y}$$
Multiclass SVM

\[
\begin{align*}
\min_{w, \xi_x \geq 0} & \quad \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \xi_x \\
\text{s.t.} & \quad w^\top (\phi(x, y) - \phi(x, \hat{y})) \geq 1 - \xi_x \quad \forall (x, y), \hat{y}
\end{align*}
\]

Properties:
Multiclass SVM

\[
\min_{w, \xi_x \geq 0} \quad \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{(x,y)} \xi_x \\
\text{s.t.} \quad w^\top (\phi(x, y) - \phi(x, \hat{y})) \geq 1 - \xi_x \quad \forall (x, y), \hat{y}
\]

Properties:
- Quadratic cost function
- Linear constraints
- Convex program

Algorithms:

Primal algorithms (Sub-gradient, cutting plane)

Dual algorithms

How does this fit into regularized surrogate loss minimization?

\[
\min_w R(w) + \frac{C}{N} \sum_{(x,y)} \bar{\ell}(x, y, w)
\]
Multiclass SVM

\[
\min_{w, \xi_x \geq 0} \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \xi_x \\
\text{s.t.} \quad w^\top (\phi(x, y) - \phi(x, \hat{y})) \geq 1 - \xi_x \quad \forall (x, y), \hat{y}
\]

Properties:
- Quadratic cost function
- Linear constraints
- Convex program

Algorithms:
- Primal algorithms (Sub-gradient, cutting plane)
- Dual algorithms
Multiclass SVM

\[
\min_{w, \xi_x \geq 0} \quad \frac{1}{2} \| w \|^2 + \frac{C}{N} \sum_{(x,y)} \xi_x \\
\text{s.t.} \quad w^\top (\phi(x, y) - \phi(x, \hat{y})) \geq 1 - \xi_x \quad \forall (x, y), \hat{y}
\]

Properties:
- Quadratic cost function
- Linear constraints
- Convex program

Algorithms:
- Primal algorithms (Sub-gradient, cutting plane)
- Dual algorithms

How does this fit into regularized surrogate loss minimization?

\[
\min_w R(w) + \frac{C}{N} \sum_{(x,y)} \bar{\ell}(x, y, w)
\]
Multiclass SVM

\[
\begin{align*}
\min_{w, \xi x \geq 0} & \quad \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \xi x \\
\text{s.t.} & \quad w^\top (\phi(x, y) - \phi(x, \hat{y})) \geq 1 - \xi x \quad \forall (x, y), \hat{y}
\end{align*}
\]

Let’s get rid of the slack variable:
Multiclass SVM

\[
\begin{align*}
\min_{w, \xi_x \geq 0} & \quad \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x,y)} \xi_x \\
\text{s.t.} & \quad w^\top (\phi(x, y) - \phi(x, \hat{y})) \geq 1 - \xi_x \quad \forall (x, y), \hat{y}
\end{align*}
\]

Let’s get rid of the slack variable:

\[
\begin{align*}
\min_w & \quad \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x,y)} \max \left\{ 0, \max_{\hat{y}} \left( 1 - w^\top (\phi(x, y) - \phi(x, \hat{y})) \right) \right\}
\end{align*}
\]
Multiclass SVM

\[
\begin{align*}
\min_{w, \xi_x \geq 0} & \quad \frac{1}{2} \| w \|^2 + \frac{C}{N} \sum_{(x, y)} \xi_x \\
\text{s.t.} & \quad w^\top (\phi(x, y) - \phi(x, \hat{y})) \geq 1 - \xi_x \quad \forall (x, y), \hat{y}
\end{align*}
\]

Let's get rid of the slack variable:

\[
\begin{align*}
\min_w \frac{1}{2} \| w \|^2 + \frac{C}{N} \sum_{(x, y)} \max \left\{ 0, \max_{\hat{y}} \left( 1 - w^\top (\phi(x, y) - \phi(x, \hat{y})) \right) \right\} \\
\geq 0
\end{align*}
\]
Multiclass SVM

\[
\begin{align*}
\min_{w, \xi_x \geq 0} & \quad \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \xi_x \\
\text{s.t.} & \quad w^\top (\phi(x, y) - \phi(x, \hat{y})) \geq 1 - \xi_x \quad \forall (x, y), \hat{y}
\end{align*}
\]

Let’s get rid of the slack variable:

\[
\begin{align*}
\min_{w} & \quad \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \max \left\{ 0, \max_{\hat{y}} \left( 1 - w^\top (\phi(x, y) - \phi(x, \hat{y})) \right) \right\} \\
& \quad \geq 0
\end{align*}
\]

\[
\begin{align*}
\min_{w} & \quad \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \max_{\hat{y}} \left( 1 + w^\top \phi(x, \hat{y}) \right) - w^\top \phi(x, y)
\end{align*}
\]
Multiclass SVM

\[
\min_{w, \xi_x \geq 0} \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x, y)} \xi_x \\
\text{s.t.} \quad w^\top (\phi(x, y) - \phi(x, \hat{y})) \geq 1 - \xi_x \quad \forall (x, y), \hat{y}
\]

Let’s get rid of the slack variable:

\[
\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x, y)} \max \left\{ 0, \max_{\hat{y}} \left( 1 - w^\top (\phi(x, y) - \phi(x, \hat{y})) \right) \right\} \geq 0
\]

\[
\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x, y)} \max_{\hat{y}} \left( 1 + w^\top \phi(x, \hat{y}) \right) - w^\top \phi(x, y)
\]

Loss-augmented inference
Multiclass SVM

\[
\begin{align*}
\min_w & \quad \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \max_{\hat{y}} \left(1 + w^\top \phi(x, \hat{y})\right) - w^\top \phi(x, y) \\
& \quad \text{(Loss-augmented inference)}
\end{align*}
\]

Fits into the general formulation:
Multiclass SVM

\[
\min_w \frac{1}{2} \|w\|^2_2 + \frac{C}{N} \sum_{(x,y)} \max_{\hat{y}} \left(1 + w^\top \phi(x, \hat{y})\right) - w^\top \phi(x, y)
\]

Loss-augmented inference

Fits into the general formulation:

\[
\min_w R(w) + \frac{C}{N} \sum_{(x,y)} \bar{\ell}(x, y, w)
\]
Multiclass SVM

How to optimize this?

$$\min_w \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{(x,y)} \max_{\hat{y}} \left( 1 + w^T \phi(x, \hat{y}) \right) - w^T \phi(x, y)$$

E.g., with gradient descent:

Iterate:

- Loss-augmented inference:

$$\arg \max \hat{y}_i \left( 1 + w^T \phi(x, \hat{y}_i) \right)$$

- Perform gradient step:

$$w \leftarrow w - \alpha \nabla_w L$$
Multiclass SVM

How to optimize this?

\[ \min_w \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{(x,y)} \max_{\hat{y}} (1 + w^T \phi(x, \hat{y})) - w^T \phi(x, y) \]

E.g., with gradient descent:

Iterate:

1. Loss-augmented inference:
   \[ \arg \max_{\hat{y}_i} (1 + w^T \phi(x, \hat{y}_i)) \]
Multiclass SVM

How to optimize this?

\[
\min_{\mathbf{w}} \frac{1}{2} \| \mathbf{w} \|_2^2 + \frac{C}{N} \sum_{(x,y)} \max_{\hat{y}} \left( 1 + \mathbf{w}^T \phi(x, \hat{y}) \right) - \mathbf{w}^T \phi(x, y)
\]

Loss-augmented inference

E.g., with gradient descent:

Iterate:

1. Loss-augmented inference:
   \[
   \arg \max_{\hat{y}_i} \left( 1 + \mathbf{w}^T \phi(x, \hat{y}_i) \right)
   \]

2. Perform gradient step:
   \[
   \mathbf{w} \leftarrow \mathbf{w} - \alpha \nabla_{\mathbf{w}} L
   \]
Multiclass SVM

How to optimize this?

\[
\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \max_{\hat{y}} (1 + w^T \phi(x, \hat{y})) - w^T \phi(x, y)
\]

E.g., with gradient descent:

Iterate:

1. Loss-augmented inference:
   \[
   \arg \max_{\hat{y}_i} (1 + w^T \phi(x, \hat{y}_i))
   \]

2. Perform gradient step:
   \[
   w \leftarrow w - \alpha \nabla_w L
   \]

How complicated is this?

Solve one multi-class inference task per sample per iteration.
Structured SVM

\[ \begin{align*}
\text{min} & \quad \|w\|_2^2 + C \sum_{(x,y)} \max \left( \hat{y} \left( \Delta(y, \hat{y}) + w^\top \phi(x, \hat{y}) \right) - w^\top \phi(x, y) \right) \\
\end{align*} \]
Structured SVM

From Multiclass SVM:

$$\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \max \left(1 + w^T \phi(x, \hat{y})\right) - w^T \phi(x, y)$$
Structured SVM

From Multiclass SVM:

$$\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \max (1 + w^T \phi(x, \hat{y})) - w^T \phi(x, y)$$

via general losses:

$$\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \max (\Delta(y, \hat{y}) + w^T \phi(x, \hat{y})) - w^T \phi(x, y)$$
Structured SVM

From Multiclass SVM:

\[
\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \max_{\hat{y}} (1 + w^T \phi(x, \hat{y})) - w^T \phi(x, y)
\]

via general losses:

\[
\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \max_{\hat{y}} (\Delta(y, \hat{y}) + w^T \phi(x, \hat{y})) - w^T \phi(x, y)
\]

to general functions and structures:

\[
\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y)} \max_{\hat{y}} (\Delta(y, \hat{y}) + F(\hat{y}, x, w)) - F(y, x, w)
\]

Loss-augmented inference
Structured SVM

How to optimize this?

\[
\min_w \frac{1}{2} \|w\|^2_2 + \frac{C}{N} \sum_{(x, y) \in \mathcal{D}} \max_{\hat{y}} \left( \Delta(y, \hat{y}) + F(\hat{y}, x, w) \right) - F(y, x, w)
\]

Loss-augmented inference

E.g., with gradient descent:

Iterate:

\[
\text{Perform gradient step: } w \leftarrow w - \alpha \nabla_w L
\]
Structured SVM

How to optimize this?

\[
\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x,y) \in D} \max_{\hat{y}} \left( \Delta(y, \hat{y}) + F(\hat{y}, x, w) \right) - F(y, x, w)
\]

E.g., with gradient descent:

Iterate:

1. Loss-augmented inference:

\[
\arg\max_{\hat{y}} \left( \Delta(y, \hat{y}) + F(\hat{y}, x, w) \right)
\]
Structured SVM

How to optimize this?

\[
\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x, y) \in D} \max_{\hat{y}} \left( \Delta(y, \hat{y}) + F(\hat{y}, x, w) \right) - F(y, x, w)
\]

E.g., with gradient descent:

Iterate:

1. **Loss-augmented inference:**
   \[
   \arg \max_{\hat{y}} \left( \Delta(y, \hat{y}) + F(\hat{y}, x, w) \right)
   \]

2. **Perform gradient step:**
   \[
   w \leftarrow w - \alpha \nabla_w L
   \]
Structured SVM

How to optimize this?

\[
\min_w \frac{1}{2} \|w\|^2_2 + \frac{C}{N} \sum_{(x,y) \in D} \max_{\hat{y}} \left( \Delta(y, \hat{y}) + F(\hat{y}, x, w) \right) - F(y, x, w)
\]

E.g., with gradient descent:

Iterate:

1. Loss-augmented inference:
   \[
   \text{arg max}_{\hat{y}} \left( \Delta(y, \hat{y}) + F(\hat{y}, x, w) \right)
   \]

2. Perform gradient step:
   \[
   w \leftarrow w - \alpha \nabla_w L
   \]

How complicated is this?

Solve one structured prediction task per sample per iteration.
Structured Learning

Soft-max extension:

$$\epsilon \ln \sum_i \exp x_i \to 0 \to \max_i x_i$$
Structured Learning

Soft-max extension:

$$\epsilon \ln \sum_i \exp \frac{x_i}{\epsilon} \quad \xrightarrow{\epsilon \to 0} \quad \max_i x_i$$
Structured Learning

Soft-max extension:

\[ \epsilon \ln \sum_i \exp \frac{x_i}{\epsilon} \xrightarrow{\epsilon \to 0} \max_i x_i \]

Let’s generalize

\[
\min_w \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{(x, y) \in D} \max_{\hat{y}} \left( \Delta(y, \hat{y}) + F(\hat{y}, x, w) \right) - F(y, x, w)
\]

even further
Structured Learning

Soft-max extension:

\[ \epsilon \ln \sum_i \exp \frac{x_i}{\epsilon} \xrightarrow{\epsilon \to 0} \max_i x_i \]

Let's generalize

\[
\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y) \in D} \max_{\hat{y}} \left( \Delta(y, \hat{y}) + F(\hat{y}, x, w) \right) - F(y, x, w)
\]

even further

\[
\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} - F(y, x, w)
\]
A pretty general formulation:

$$\min_w \frac{1}{2} \|w\|^2 + \frac{C}{N} \sum_{(x,y) \in \mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} - F(y, x, w)$$
Structured Learning

A pretty general formulation:

$$\min_w \frac{1}{2} \|w\|^2_2 + \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} - F(y, x, w)$$

Generalizes:
- any form of SVM
- conditional random fields
- maximum-likelihood
- logistic regression
- convolutional neural networks
Structured Learning

A pretty general formulation:

$$\min_w \frac{1}{2} \left\| w \right\|^2_2 + \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} - F(y, x, w)$$

Generalizes:

- any form of SVM
- conditional random fields
- maximum-likelihood
- logistic regression
- convolutional neural networks

How does this correspond to a binary SVM?
A pretty general formulation:

\[
\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} - F(y, x, w)
\]

Generalizes:
- any form of SVM
- conditional random fields
- maximum-likelihood
- logistic regression
- convolutional neural networks

How does this correspond to a binary SVM? Reverse slides.
A pretty general formulation:

$$\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y) \in \mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \left( \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} \right) - F(y, x, w)$$

Generalizes:

- any form of SVM
- conditional random fields
- maximum-likelihood
- logistic regression
- convolutional neural networks

How does this correspond to a binary SVM? Reverse slides. How does this correspond to logistic regression?
Structured Learning

A pretty general formulation:

$$\min_w \frac{1}{2} \|w\|_2^2 + \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} - F(y, x, w)$$

Generalizes:
- any form of SVM
- conditional random fields
- maximum-likelihood
- logistic regression
- convolutional neural networks

How does this correspond to a binary SVM? Reverse slides.
How does this correspond to logistic regression? Up next...
\[
\min_w \frac{1}{2} \| w \|^2_2 + \frac{C}{N} \sum_{(x, y) \in D} \varepsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\varepsilon} - F(y, x, w)
\]
Structured Learning

\[
\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} - F(y, x, w)
\]

Equivalent formulation:

\[
\min_w \frac{1}{2} \| w \|_2^2 - \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \frac{\exp \frac{\Delta(y,y) + F(y, x, w)}{\epsilon}}{\sum_{\hat{y}} \exp \frac{\Delta(y,\hat{y}) + F(\hat{y}, x, w)}{\epsilon}}
\]
Structured Learning

$$\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} - F(y, x, w)$$

Equivalent formulation:

$$\min_w \frac{1}{2} \| w \|_2^2 - \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \frac{\exp \frac{\Delta(y,y)+F(y,x,w)}{\epsilon}}{\sum_{\hat{y}} \exp \frac{\Delta(y,\hat{y})+F(\hat{y},x,w)}{\epsilon}}$$

Annealed loss-augmented probability:

$$p^{\Delta}_\epsilon(y \mid x, w) = \left( \frac{\exp \frac{\Delta(y,y)+F(y,x,w)}{\epsilon}}{\sum_{\hat{y}} \exp \frac{\Delta(y,\hat{y})+F(\hat{y},x,w)}{\epsilon}} \right)^\epsilon$$
\[
\min_w \frac{1}{2}\|w\|_2^2 + \frac{C}{N} \sum_{(x,y) \in \mathcal{D}} \epsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} - F(y, x, w)
\]

Equivalent formulation:

\[
\min_w \frac{1}{2}\|w\|_2^2 - \frac{C}{N} \sum_{(x,y) \in \mathcal{D}} \epsilon \ln \frac{\exp \frac{\Delta(y,y) + F(y,x,w)}{\epsilon}}{\sum_{\hat{y}} \exp \frac{\Delta(y,\hat{y}) + F(\hat{y},x,w)}{\epsilon}}
\]

Annealed loss-augmented probability:

\[
p_{\epsilon}^{\Delta}(y \mid x, w) = \left( \frac{\exp \frac{\Delta(y,y) + F(y,x,w)}{\epsilon}}{\sum_{\hat{y}} \exp \frac{\Delta(y,\hat{y}) + F(\hat{y},x,w)}{\epsilon}} \right)^\epsilon
\]

Equivalent maximum-likelihood formulation:

\[
\min_w \frac{1}{2}\|w\|_2^2 - \frac{C}{N} \ln \prod_{(x,y) \in \mathcal{D}} p_{\epsilon}^{\Delta}(y \mid x, w)
\]
\[
\min_w \frac{1}{2} \| w \|_2^2 - \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \frac{\exp \frac{\Delta(y,y) + F(y,x,w)}{\epsilon}}{\sum_{\hat{y}} \exp \frac{\Delta(y,\hat{y}) + F(\hat{y},x,w)}{\epsilon}}
\]

Simplifications:
\[
\min_w \frac{1}{2} \|w\|^2 - \frac{C}{N} \sum_{(x,y) \in \mathcal{D}} \varepsilon \ln \frac{\exp \left( \frac{\Delta(y,y) + F(y,x,w)}{\varepsilon} \right)}{\sum_{\hat{y}} \exp \left( \frac{\Delta(y,\hat{y}) + F(\hat{y},x,w)}{\varepsilon} \right)}
\]

Simplifications:

- **Binary classification**: \( y \in \{-1, 1\} \)
- **Simple feature-function**: \( F(y, x, w) = yw^\top x \)
- **Epsilon**: \( \varepsilon = 1 \)
\[
\min_w \frac{1}{2} \|w\|^2_2 - \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \frac{\exp \frac{\Delta(y,y) + F(y,x,w)}{\epsilon}}{\sum_{\hat{y}} \exp \frac{\Delta(y,\hat{y}) + F(\hat{y},x,w)}{\epsilon}}
\]

Simplifications:

- Binary classification: \( y \in \{-1, 1\} \)
- Simple feature-function: \( F(y, x, w) = yw^\top x \)
- Epsilon: \( \epsilon = 1 \)

\[
\min_w \frac{1}{2} \|w\|^2_2 - \frac{C}{N} \ln \prod_{(x,y)} \frac{\exp yw^\top x}{\exp(w^\top x) + \exp(-w^\top x)}
\]
\[
\min_w \frac{1}{2} \|w\|_2^2 - \frac{C}{N} \sum_{(x,y) \in \mathcal{D}} \varepsilon \ln \frac{\exp \frac{\Delta(y,y) + F(y,x,w)}{\varepsilon}}{\sum_{\hat{y}} \exp \frac{\Delta(y,\hat{y}) + F(\hat{y},x,w)}{\varepsilon}}
\]

Simplifications:

- Binary classification: \(y \in \{-1, 1\}\)
- Simple feature-function: \(F(y, x, w) = yw^\top x\)
- Epsilon: \(\varepsilon = 1\)

\[
\min_w \frac{1}{2} \|w\|_2^2 - \frac{C}{N} \ln \prod_{(x,y)} \frac{\exp yw^\top x}{\exp(w^\top x) + \exp(-w^\top x)}
\]

\[
\min_w \frac{1}{2} \|w\|_2^2 - \frac{C}{N} \ln \prod_{(x,y)} \frac{1}{1 + \exp (-2yw^\top x)}
\]
\[
\min_w \frac{1}{2} \|w\|_2^2 - \frac{C}{N} \sum_{(x,y) \in \mathcal{D}} \epsilon \ln \frac{\exp \frac{\Delta(y,y) + F(y,x,w)}{\epsilon}}{\sum_{\hat{y}} \exp \frac{\Delta(y,\hat{y}) + F(\hat{y},x,w)}{\epsilon}}
\]

Simplifications:
- Binary classification: \( y \in \{-1, 1\} \)
- Simple feature-function: \( F(y, x, w) = y w^\top x \)
- Epsilon: \( \epsilon = 1 \)

\[
\min_w \frac{1}{2} \|w\|_2^2 - \frac{C}{N} \ln \prod_{(x,y)} \frac{\exp y w^\top x}{\exp(w^\top x) + \exp(-w^\top x)}
\]

\[
\min_w \frac{1}{2} \|w\|_2^2 - \frac{C}{N} \ln \prod_{(x,y)} \frac{1}{1 + \exp(-2y w^\top x)}
\]

\[
\min_w \frac{1}{2} \|w\|_2^2 - \frac{C}{N} \sum_{(x,y)} \ln \sigma \left(2y w^\top x\right) \quad \text{with} \quad \sigma(z) = \frac{1}{1 + \exp(-z)}
\]
We started with hinge loss:

$$\min_w \frac{1}{2} \|w\|^2_2 + \frac{C}{N} \sum_{(x,y)}^n \max \{0, 1 - yw^\top x\}$$
We started with hinge loss:

\[
\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x,y)}^n \max\{0, 1 - yw^\top x\}
\]

Derived the general structured prediction framework:

\[
\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x,y) \in D} \epsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} - F(y, x, w)
\]
We started with hinge loss:

$$\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x,y)}^n \max \{ 0, 1 - yw^T x \}$$

Derived the general structured prediction framework:

$$\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x,y) \in D} \varepsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon} - F(y, x, w)$$

And showed how to simplify it to logistic regression:

$$\min_w \frac{1}{2} \| w \|_2^2 - \frac{C}{N} \sum_{(x,y)} \ln \sigma \left( 2yw^T x \right)$$
We started with hinge loss:

\[
\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x,y)}^n \max\{0, 1 - yw^T x\}
\]

Derived the general structured prediction framework:

\[
\min_w \frac{1}{2} \| w \|_2^2 + \frac{C}{N} \sum_{(x,y) \in D} \frac{\epsilon \ln \sum_{\hat{y}} \exp \frac{\Delta(y, \hat{y}) + F(\hat{y}, x, w)}{\epsilon}}{-F(y, x, w)}
\]

And showed how to simplify it to logistic regression:

\[
\min_w \frac{1}{2} \| w \|_2^2 - \frac{C}{N} \sum_{(x,y)} \ln \sigma \left( 2yw^T x \right)
\]
How does this framework simplify to neural networks?

- Multiclass setting: \( y \in \{1, \ldots, K\} \)
- Composite feature function:
  \[
  F(y, x, w) = f_1(y, w_1, f_2(w_2, f_3(\ldots)))
  \]
- Epsilon: \( \epsilon = 1 \)

Stay tuned for part III