Statistical Machine Learning
https://cvml.ist.ac.at/courses/SML_W18

Christoph Lampert

IST Austria
Institute of Science and Technology

Winter Semester 2018/2019
Lecture 9

(lots of material courtesy of S. Nowozin, http://www.nowozin.net)
Standard Regression/Classification:

\[ f : \mathcal{X} \rightarrow \mathbb{R}. \]

- inputs \( \mathcal{X} \) can be any kind of objects
- output \( y \in \mathcal{Y} \) is a number (real or integer)

Structured Prediction:

\[ f : \mathcal{X} \rightarrow \mathcal{Y}. \]

- inputs \( \mathcal{X} \) can be any kind of objects
- outputs \( y \in \mathcal{Y} \) are complex (structured) objects
**What is structured data?**

**Ad hoc definition:** data that consists of several parts, and not only the parts themselves contain information, but also the way in which the parts belong together.
What is structured output prediction?

**Ad hoc definition:** predicting *structured* outputs from input data
(in contrast to predicting just a single number, like in classification or regression)

- **Natural Language Processing:**
  - Automatic Translation (output: sentences)

- **Bioinformatics:**
  - Secondary Structure Prediction (output: bipartite graphs)

- **Speech Processing:**
  - Text-to-Speech (output: audio signal)

- **Robotics:**
  - Planning (output: sequence of actions)

- **Information Retrieval:**
  - Ranking (output: ordered list of documents)

This lecture: mainly examples from Computer Vision
Example: Human Pose Estimation

- Given an image, where is a person and how is it articulated?

\[ f : \mathcal{X} \rightarrow \mathcal{Y} \]

- Image \( x \), but what is \( y \in \mathcal{Y} \) precisely?

Image: Flickr.com user lululemon athletica
Example: Human Pose Estimation

- Body Part: $y_{\text{head}} = (u, v, \theta)$ where $(u, v)$ center, $\theta$ rotation
  - $(u, v) \in \{1, \ldots, M\} \times \{1, \ldots, N\}$, $\theta \in \{0, 45^\circ, 90^\circ, \ldots\}$
Example: Human Pose Estimation

- **Body Part:** $y_{\text{head}} = (u, v, \theta)$ where $(u, v)$ center, $\theta$ rotation
  - $(u, v) \in \{1, \ldots, M\} \times \{1, \ldots, N\}$, $\theta \in \{0, 45^\circ, 90^\circ, \ldots\}$
Example: Human Pose Estimation

- **Body Part:** $y_{head} = (u, v, \theta)$ where $(u, v)$ center, $\theta$ rotation
  - $(u, v) \in \{1, \ldots, M\} \times \{1, \ldots, N\}$, $\theta \in \{0, 45^\circ, 90^\circ, \ldots\}$
Example: Human Pose Estimation

- Body Part: $y_{\text{head}} = (u, v, \theta)$ where $(u, v)$ center, $\theta$ rotation
  - $(u, v) \in \{1, \ldots, M\} \times \{1, \ldots, N\}$, $\theta \in \{0, 45^\circ, 90^\circ, \ldots\}$
Body Part: $y_{\text{head}} = (u, v, \theta)$ where $(u, v)$ center, $\theta$ rotation

- $(u, v) \in \{1, \ldots, M\} \times \{1, \ldots, N\}$, $\theta \in \{0, 45^\circ, 90^\circ, \ldots\}$
Example: Human Pose Estimation

- Body Part: \( y_{\text{head}} = (u, v, \theta) \) where \((u, v)\) center, \(\theta\) rotation
  - \((u, v) \in \{1, \ldots, M\} \times \{1, \ldots, N\}, \theta \in \{0, 45^\circ, 90^\circ, \ldots\}\)
Example: Human Pose Estimation

- Body Part: $y_{\text{head}} = (u, v, \theta)$ where $(u, v)$ center, $\theta$ rotation
  - $(u, v) \in \{1, \ldots, M\} \times \{1, \ldots, N\}$, $\theta \in \{0, 45^\circ, 90^\circ, \ldots\}$
- same for torso, left arm, right arm, ...
- Entire Body: $y = (y_{\text{head}}, y_{\text{torso}}, y_{\text{left-lower-arm}}, \ldots) \in \mathcal{Y}$
Example: Human Pose Estimation

- Idea: Have a head detector (CNN, SVM, RF, ...)

\[ f_{\text{head}} : \mathcal{X} \rightarrow \mathbb{R} \]
Example: Human Pose Estimation

- Idea: Have a head detector (CNN, SVM, RF, ...)

\[ f_{\text{head}} : \mathcal{X} \rightarrow \mathbb{R} \]

- Evaluate for every possible location and record score
- Same construction for all other body parts
Example: Human Pose Estimation

Image $x \in \mathcal{X}$

- Put together body from individual parts

$$y^{\text{best}} = (y^{\text{best}}_{\text{head}}, y^{\text{best}}_{\text{torso}}, \ldots)$$
Example: Human Pose Estimation

- Put together body from individual parts

\[ y^{best} = (y^{best}_{head}, y^{best}_{torso}, \ldots) \]

- Each part looks reasonable, but overall makes no sense
Enforce **relations between parts**

- For example, *head* must be connected to *torso*
- Problem:

\[ y^{\text{best}} \neq (y^{\text{best}}_{\text{head}}, y^{\text{best}}_{\text{torso}}, \ldots) \]

independent decisions for each body part are not optimal anymore
Enforce **relations between parts**

- For example, *head* must be connected to *torso*
- Problem:

\[ y^{\text{best}} \neq (y^{\text{best}}_{\text{head}}, y^{\text{best}}_{\text{torso}}, \ldots) \]

independent decisions for each body part are not optimal anymore

- Needs structured output prediction function \( f : \mathcal{X} \rightarrow \mathcal{Y} \)
### Normal prediction function, $\mathcal{X} = \text{anything}, \mathcal{Y} = \mathbb{R}$

Extract feature vector from $x$ and compute a number from it

\[ f(x) = \langle w, \phi(x) \rangle + b \]
The general recipe

**Normal prediction function, \( \mathcal{X} = \text{anything}, \mathcal{Y} = \mathbb{R} \)**

Extract feature vector from \( x \) and compute a number from it

\[
f(x) = \langle w, \phi(x) \rangle + b
\]

**Structured output prediction function, \( \mathcal{X} = \text{anything}, \mathcal{Y} = \text{anything} \)**

1) Define auxiliary function, \( g : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R} \),

\[
ge(x, y) = \prod_i \psi_i(y_i, x) \prod_{i \sim j} \psi_{ij}(y_i, y_j, x)
\]

2) Construct \( f : \mathcal{X} \rightarrow \mathcal{Y} \) from \( g \), e.g.,

\[
f(x) = \arg\max_{y \in \mathcal{Y}} g(x, y)
\]

Challenges:

- how to learn \( g(x, y) \) from training data?
- how to compute \( f(x) \) from \( g(x, y) \)?
Supervised Learning Problem

- Given training examples $(x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}$
- $x \in \mathcal{X}$: input, e.g. image
- $y \in \mathcal{Y}$: structured output, e.g. human pose, sentence

Images: HumanEva dataset

- How to make predictions for new inputs, i.e. learn a function $f : \mathcal{X} \rightarrow \mathcal{Y}$?
Supervised Learning Problem

• Given training examples \((x_1, y_1), \ldots, (x_n, y_n) \in \mathcal{X} \times \mathcal{Y}\)
  \(x \in \mathcal{X}\): input, e.g. image
  \(y \in \mathcal{Y}\): structured output, e.g. human pose, sentence

• How to make predictions for new inputs, i.e. learn \(f : \mathcal{X} \rightarrow \mathcal{Y}\) ?

Approach 1) Discriminative Probabilistic Learning

1) Use training data to obtain an estimate \(p(y|x)\).

2) Use \(f(x) = \text{argmin}_{\bar{y} \in \mathcal{Y}} \sum_{y} p(y|x) \Delta(y, \bar{y})\) to make predictions.
   \(\Delta : \mathcal{Y} \rightarrow \mathcal{Y} \rightarrow \mathbb{R}_+\) is a structured loss function (later...)

Approach 2) Loss-minimizing Parameter Estimation

1) Use training data to learn a compatibility function \(g(x, y)\)

2) Use \(f(x) := \text{argmax}_{y \in \mathcal{Y}} g(x, y)\) to make predictions.
Probabilistic Graphical Models
Refresher: Conditional Probability Distributions

Binary Classification

\( \mathcal{X} = \{ \text{anything} \} , \ \mathcal{Y} = \{ \pm 1 \} \)

- \( p(y|x) \): 2 values for each \( x \),
  1 degree of freedom
- learn one function: \( \mathcal{X} \rightarrow \mathbb{R} \)

Multi-class prediction

\( y \in \mathcal{Y} = \{1, \ldots, K\} \)

- \( p(y|x) \): \( K \) values for each \( x \),
- learn \( K - 1 \) functions, or
  \( K \) functions with normalization
Structured objects: predicting $M$ variables jointly

$\mathcal{Y} = \{1, K\} \times \{1, K\} \times \cdots \times \{1, K\}$

For each $x$:

- $K^M$ values, $K^M - 1$ d.o.f.
  $\rightarrow K^M$ functions

Example: pose estimation

$\mathcal{Y}_{\text{part}} = \{1, \ldots, W\} \times \{1, \ldots, H\}$
  $\times \{1, \ldots, 360\}$

$\mathcal{Y} = \mathcal{Y}_{\text{head}} \times \mathcal{Y}_{\text{left-arm}} \times \cdots \times \mathcal{Y}_{\text{right-foot}}$

For each $x$:

- $(360WH)^\#\text{body parts}$ values
  $\rightarrow$ many billions function
Example: image denoising

\[ \mathcal{Y} = \{640 \times 480 \text{ RGB images}\} \]

For each \( x \):
- \((255^3)^{640 \cdot 480}\) values in \(p(y|x)\),
  \rightarrow over 10^{2,000,000} functions

\textbf{too much!}
**Example: image denoising**

\[ \mathcal{Y} = \{640 \times 480 \text{ RGB images}\} \]

For each \( x \):
- \((255^3)^{640 \times 480}\) values in \( p(y|x) \),
- \( \rightarrow \) over \( 10^{2,000,000} \) functions

**too much!**

We cannot consider all possible distributions, we must impose **structure**.
A (probabilistic) graphical model defines

- a family of probability distributions over a set of random variables, by means of a graph.

Popular classes of graphical models:

- Undirected graphical models (Markov random fields),
- Directed graphical models (Bayesian networks),
- Factor graphs,
- Others: chain graphs, influence diagrams, etc.

The graph encodes conditional independence assumptions:

Let $N(i)$ be the neighbors of node $i$ in the graph $(V, E)$. Then $p(y_i | y_{V \setminus \{i\}}) = p(y_i | y_{N(i)})$ with $y_{V \setminus \{i\}} = (y_1, ..., y_{i-1}, y_{i+1}, y_n)$. 

16 / 31
A (probabilistic) graphical model defines

- a family of probability distributions over a set of random variables, by means of a graph.

Popular classes of graphical models,

- Undirected graphical models (Markov random fields),
- Directed graphical models (Bayesian networks),
- Factor graphs,
- Others: chain graphs, influence diagrams, etc.
A (probabilistic) graphical model defines

- a family of probability distributions over a set of random variables, by means of a graph.

Popular classes of graphical models,

- Undirected graphical models (Markov random fields),
- Directed graphical models (Bayesian networks),
- Factor graphs,
- Others: chain graphs, influence diagrams, etc.

The graph encodes conditional independence assumptions between the variables:

- Let $N(i)$ be the neighbors of node $i$ in the graph $(V, E)$. Then

$$p(y_i | y_{V \backslash \{i\}}) = p(y_i | y_{N(i)})$$

with $y_{V \backslash \{i\}} = (y_1, \ldots, y_{i-1}, y_{i+1}, y_n)$. 
• All parts depend on each other.
  ▶ Knowing where the head is puts constraints on where the feet can be.

• But **conditional independences** as specified by the graph:
  ▶ If we fix where the **left leg** is, the **left foot**’s position does not depend on the **torso** or the **head** position anymore, etc.

\[
p(y_{\text{left-foot}} | y_{\text{top}}, \ldots, y_{\text{torso}}, \ldots, y_{\text{right-foot}}, x) = p(y_{\text{left-foot}} | y_{\text{left-leg}}, x)
\]
Factor Graphs

- Decomposable output $y = (y_1, \ldots, y_{|V|})$

- Graph: $G = (V, \mathcal{F})$,
  - variable nodes $V$,
  - factor nodes $\mathcal{F}$,
  - each factor $F \in \mathcal{F}$ connects a subset of nodes,
  - write $F = \{v_1, \ldots, v_{|F|}\}$ and $y_F = (y_{v_1}, \ldots, y_{v_{|F|}})$
Factor Graphs

- Decomposable output \( y = (y_1, \ldots, y_{|V|}) \)

- Graph: \( G = (V, \mathcal{F}) \),
  - variable nodes \( V \),
  - factor nodes \( \mathcal{F} \),
  - each factor \( F \in \mathcal{F} \) connects a subset of nodes,
  - write \( F = \{v_1, \ldots, v_{|F|}\} \) and \( y_F = (y_{v_1}, \ldots, y_{v_{|F|}}) \)

- Distribution factorizes into potentials \( \psi \) at factors:
  \[
p(y) = \frac{1}{Z} \prod_{F \in \mathcal{F}} \psi_F(y_F)
  \]

- \( Z \) is a normalization constant, called partition function:
  \[
  Z = \sum_{y \in \mathcal{Y}} \prod_{F \in \mathcal{F}} \psi_F(y_F).
  \]
Conditional Distributions

How to model $p(y|x)$?

- Potentials become also functions of (part of) $x$: $\psi_F(y_F; x_F)$ instead of just $\psi_F(y_F)$

$$p(y|x) = \frac{1}{Z(x)} \prod_{F \in \mathcal{F}} \psi_F(y_F; x_F)$$

- Partition function depends on $x_F$

$$Z(x) = \sum_{y \in \mathcal{Y}} \prod_{F \in \mathcal{F}} \psi_F(y_F; x_F).$$

- Note: $x$ is treated just as an argument, not as a random variable.

Conditional random fields (CRFs)
Assume $\psi_F(y_F) > 0$. Then

- instead of potentials, we can use energies:

$$E_F(y_F; x_F) = -\log(\psi_F(y_F; x_F)) \quad \text{for each factor } F.$$  

$$E(y; x) = \sum_{F \in \mathcal{F}} E_F(y_F; x_F) \quad \text{total energy}$$
Assume $\psi_F(y_F) > 0$. Then

- instead of potentials, we can use **energies**:
  
  $$E_F(y_F; x_F) = -\log(\psi_F(y_F; x_F))$$  
  for each factor $F$.

  $$E(y; x) = \sum_{F \in \mathcal{F}} E_F(y_F; x_F)$$  
  total energy

- $p(y|x)$ can be written as **Gibbs distribution**
  
  $$p(y|x) = \frac{1}{Z(x)} \prod_{F \in \mathcal{F}} \psi_F(y_F; x_F)$$

  $$= \frac{1}{Z(x)} \exp(- \sum_{F \in \mathcal{F}} E_F(y_F; x_F)) = \frac{1}{Z(x)} \exp(-E(y; x))$$

In practice, one directly models the energy function
→ the probability distribution is uniquely determined by it.
Example: An Energy Function for Human Pose Estimation

\[
E(y; x) = \sum_{i \in \{\text{head, torso, ...} \}} E_i(y_i; x) + \sum_{(i,j)} E_{ij}(y_i, y_j)
\]

- unary factors (depend on one label): appearance
  - e.g. \( E_{\text{head}}(y; x) \) "Does location \( y \) in image \( x \) look like a head?"

- pairwise factors (depend on two labels): geometry
  - e.g. \( E_{\text{head-torso}}(y_{\text{head}}, y_{\text{torso}}) \) "Is location \( y_{\text{head}} \) above location \( y_{\text{torso}} \)?"
Example: An Energy Function for Image Segmentation

Object segmentation: e.g. horse

$\mathbf{x}$: 

$\mathbf{y}$:

Energy function components ("Ising" model):

- $E_i(y_i = 1, x_i) = \begin{cases} 
  \text{low} & \text{if } x_i \text{ is the right color, e.g. brown} \\
  \text{high} & \text{otherwise}
\end{cases}$

- $E_i(y_i = 0, x_i) = -E_i(y_i = 1, x_i)$

- $E_i(y_i, y_j) = \begin{cases} 
  \text{low} & \text{if } y_i = y_j \\
  \text{high} & \text{otherwise}
\end{cases}$

prefer that neighbors have the same label $\rightarrow$ smooth labelings
What to do with Structured Prediction Models?

Case 1) $p(y|x)$ is known

MAP Prediction

Predict $f : \mathcal{X} \rightarrow \mathcal{Y}$ by optimization

$$y^* = \arg\max_{y \in \mathcal{Y}} p(y|x) = \arg\min_{y \in \mathcal{Y}} E(y, x)$$

Probabilistic Inference

Compute marginal probabilities

$$p(y_F|x)$$

for any factor $F$, in particular, $p(y_i|x)$ for all $i \in V$. 
What to do with Structured Prediction Models?

- MAP makes a single (structured) prediction
  - best overall pose

- Marginal probabilities $p(y_i|x)$ give us
  - potential positions
  - uncertainty of the individual body parts.

Images: Buffy Stickmen dataset (Ferrari et al.)
Case 2) $p(y|x)$ is unknown, but we have training data

**Structure Learning**
Learn graph structure from training data.

**Variable Learning**
Learn, whether to use additional (latent) variables, and which ones. (input and output variables are fixed by the task we try to solve).

**Parameter Learning**
Assume a fixed factor graph, learn parameters of the energy.
 Conditional Random Fields

\[ \max_w p(y|x; w) \]
Conditional Random Field Learning

Goal: learn a conditional distribution

\[
p(y|x) = \frac{1}{Z(x)} e^{-\sum_{F \in \mathcal{F}} E_F(y_F;x)}
\]

with \( \mathcal{F} = \{ \text{all factors} \} \): all unary, pairwise, potentially higher order, \ldots

- parameterize each \( E_F(y_F;x) = \langle w_F, \phi_F(x, y_F) \rangle \).
- fixed feature functions \((\phi_1(x, y_1), \ldots, \phi_{|\mathcal{F}|}(x, y_{|\mathcal{F}|})) \equiv \phi(x, y)\)
- weight vectors \((w_1, \ldots, w_{|\mathcal{F}|}) \equiv w\)

Result: log-linear model with parameter vector \( w \)

\[
p(y|x; w) = \frac{1}{Z(x; w)} e^{-\langle w, \phi(y,x) \rangle}
\]

with \( Z(x; w) = \sum_{\bar{y} \in \mathcal{Y}} e^{-\langle w, \phi(\bar{y},x) \rangle} \) ("partition function")

New goal: find best parameter vector \( w \in \mathbb{R}^D \).
Probabilistic Learning

Maximize conditional likelihood, \( p(D_y|D_x; w) \), or maximum posterior, \( p(w|D) \). Equivalently, minimize

\[
\mathcal{L}(w) = \frac{\lambda}{2} \|w\|^2 - \sum_{n=1}^{N} \log p(y^n|x^n; w)
\]

\[
= \frac{\lambda}{2} \|w\|^2 + \sum_{n=1}^{N} \left[ \langle w, \phi(x^n, y^n) \rangle + \log \sum_{y \in Y} e^{-\langle w, \phi(x^n, y) \rangle} \right]
\]

(\( \lambda = 0 \) makes it unregularized)

Same optimization problem as for multi-class **logistic regression**.

- unconstrained
- smooth
- convex
Task: Compute $v = \nabla_w \mathcal{L}(w_{\text{cur}})$ and evaluate $\mathcal{L}(w_{\text{cur}} + \eta v)$:

$$\mathcal{L}(w) = \frac{\lambda}{2} \| w \|^2 + \sum_{n=1}^{N} \left[ \langle w, \phi(x^n, y^n) \rangle + \log \sum_{y \in \mathcal{Y}} e^{-\langle w, \phi(x^n, y) \rangle} \right]$$

$$\nabla_w \mathcal{L}(w) = \lambda w + \sum_{n=1}^{N} \left[ \phi(x^n, y^n) - \sum_{y \in \mathcal{Y}} p(y|x^n; w) \phi(x^n, y) \right]$$
Solving the Training Optimization Problem in Practice

**Task:** Compute \( v = \nabla_w L(w_{cur}) \) and evaluate \( L(w_{cur} + \eta v) \):

\[
L(w) = \frac{\lambda}{2} \|w\|^2 + \sum_{n=1}^{N} \left[ \langle w, \phi(x^n, y^n) \rangle + \log \sum_{y \in Y} e^{-\langle w, \phi(x^n, y) \rangle} \right]
\]

\[
\nabla_w L(w) = \lambda w + \sum_{n=1}^{N} \left[ \phi(x^n, y^n) - \sum_{y \in Y} p(y|x^n; w) \phi(x^n, y) \right]
\]

**Problem:** \( Y \) typically is very (exponentially) large:
- binary image segmentation: \(|Y| = 2^{640 \times 480} \approx 10^{92475}\)
- ranking \( N \) images: \(|Y| = N!\), e.g. \( N = 1000\): \(|Y| \approx 10^{2568}\).

We must use the **structure** in \( Y \), otherwise we’re lost.
∇_w \mathcal{L}(w) = \lambda w + \sum_{n=1}^{N} \left[ \phi(x^n, y^n) - \mathbb{E}_{y \sim p(y|x^n; w)} \phi(x^n, y) \right]

Computing the Gradient (naive): \( O(K^M ND) \)

\[ \mathcal{L}(w) = \frac{\lambda}{2} \|w\|^2 + \sum_{n=1}^{N} \left[ \langle w, \phi(x^n, y^n) \rangle + \log Z(x^n; w) \right] \]

Line Search (naive): \( O(K^M ND) \) per evaluation of \( \mathcal{L} \)

- \( N \): number of samples
- \( D \): dimension of feature space
- \( M \): number of output variables \( \approx 10s \) to \( 1,000,000s \)
- \( K \): number of possible labels of each output variables \( \approx 2 \) to \( 1000s \)
In a graphical model with factors $\mathcal{F}$, the features decompose:

$$\phi(x, y) = \left( \phi_F(x, y_F) \right)_{F \in \mathcal{F}}$$

$$\mathbb{E}_{y \sim p(y|x; w)} \phi(x, y) = \left( \mathbb{E}_{y \sim p(y|x; w)} \phi_F(x, y_F) \right)_{F \in \mathcal{F}}$$

$$= \left( \mathbb{E}_{y_F \sim p(y_F|x; w)} \phi_F(x, y_F) \right)_{F \in \mathcal{F}}$$

$$\mathbb{E}_{y_F \sim p(y_F|x; w)} \phi_F(x, y_F) = \sum_{y_F \in \mathcal{Y}_F} p(y_F|x; w) \phi_F(x, y_F)$$

Factor marginals $\mu_F = p(y_F|x; w)$

- are much smaller than complete joint distribution $p(y|x; w)$,
- compute/approximate them by probabilistic inference