Lecture 8: Discriminative structured output learning, Perceptron algorithm

Vojtěch Franc

April 16, 2015

7.A: Definition of structured classification task and its solution via generative and discriminative learning

7.B: Implementation of ERM learning using Perceptron algorithm

7.C: Learning of max-sum classifier

XEP33SML – Structured Model Learning, Summer 2015

7.A: Structured Output Classification

The setting

- $\boldsymbol{x} \in \mathcal{X}$ is input (also observation or measurement)
- $\boldsymbol{y} \in \mathcal{Y}$ is output (also hidden state or label)
- $(\boldsymbol{x}, \boldsymbol{y})$ are assumed to be realizations of random variables (X, Y) with p.d.f. $p(\boldsymbol{x}, \boldsymbol{y})$
- $\ell \colon \mathcal{Y} \times \mathcal{Y} \to [0, \infty)$ loss function

The task is to find the optimal (Bayes) classification strategy $h \colon \mathcal{X} \to \mathcal{Y}$ achieving the minimal expected risk

$$R^* = \inf_{h \colon \mathcal{X} \to \mathcal{Y}} R(h) \qquad \text{where} \qquad R(h) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}) \sim p}[\ell(\boldsymbol{y}, h(\boldsymbol{x})]$$

Taxonomy based on the set $\mathcal{Y}$:

1. (Flat) classification: $\mathcal{Y} = \{1, \ldots, K\}$

2. Structured output classification: $\mathcal{Y}$ is a finite set of structured objects

   E.g. $\boldsymbol{y} = (y_i \in \mathcal{Y}_i \mid i \in \mathcal{V}) \in \mathcal{Y} = \mathcal{Y}_1 \times \cdots \times \mathcal{Y}_{|\mathcal{V}|}$ is a labeling of a set of parts $\mathcal{V}$.

7.A: Learning: using examples to solve the task

- Typically, $p(\boldsymbol{x}, \boldsymbol{y})$ is unknown but we are given training set

$$\mathcal{T} = \{(\boldsymbol{x}^i, \boldsymbol{y}^i) \in \mathcal{X} \times \mathcal{Y} \mid i \in \mathcal{I} = \{1, \ldots, m\}\}$$

  drawn from i.i.d. random variables with p.d.f. $p(\boldsymbol{x}, \boldsymbol{y})$.

- Given training set $\mathcal{T}$, the learning algorithm $A\colon \cup_{m=1}^{\infty} (\mathcal{X} \times \mathcal{Y})^m \to \mathcal{H}$ selects a classifier $h_{\mathcal{T}}$ from a fixed class of strategies $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$.

- The learning algorithm is characterized by the excess error:

$$\underbrace{\left(R(h_{\mathcal{T}}) - R^*\right)}_{\text{excess error}} = \underbrace{\left(R(h_{\mathcal{T}}) - R(h_{\mathcal{H}})\right)}_{\text{estimation error}} + \underbrace{\left(R(h_{\mathcal{H}}) - R^*\right)}_{\text{approximation error}}$$

  where $h_{\mathcal{H}}$ denotes the optimal classifier within the class $\mathcal{H}$, i.e

$$R(h_{\mathcal{H}}) = \inf_{h \in \mathcal{H}} R(h)$$

- Selection of the class of strategies $\mathcal{H}$ controls the trade-off between the estimation error (depends on $\mathcal{H}$ and m) and the approximation error (depends on $\mathcal{H}$).

7.A: Generative learning (lectures 3-7)

1. Use the training set $\mathcal{T} = \{(\boldsymbol{x}^i, \boldsymbol{y}^i) \in \mathcal{X} \times \mathcal{Y} \mid i \in \{1, \dots, m\}\}$ to approximate $p(\boldsymbol{x}, \boldsymbol{y})$ by $\hat{p}(\boldsymbol{x}, \boldsymbol{y})$.

   For example, use the Maximum-Likelihood method:

   (a) Guess the shape of the distribution, e.g. $\hat{p}_{\boldsymbol{u}}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{Z(\boldsymbol{u})} \exp\langle \boldsymbol{u}, \boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{y}) \rangle$, $\boldsymbol{u} \in \mathcal{U}$

   (b) Find the ML estimate

$$\boldsymbol{u}_{\mathcal{T}} \in \operatorname*{argmax}_{\boldsymbol{u} \in \mathcal{U}} \sum_{i=1}^{m} \log \hat{p}_{\boldsymbol{u}}(\boldsymbol{x}^i, \boldsymbol{y}^i)$$

2. Construct a plug-in classifier

$$h_{\mathcal{T}}(\boldsymbol{x}) \in \operatorname*{argmin}_{h\colon \mathcal{X} \to \mathcal{Y}} \mathbb{E}_{(\boldsymbol{x}, \boldsymbol{y}) \sim \hat{p}}[\ell(\boldsymbol{y}, h(\boldsymbol{x}))]$$

7.A: Discriminative learning

- Replace $R(h)$ by a surrogate risk $R_{\mathcal{T}}(h)$ which can be evaluated on the training set $\mathcal{T} = \{(\boldsymbol{x}^i, \boldsymbol{y}^i) \in \mathcal{X} \times \mathcal{Y} \mid i \in \{1, \dots, m\}\}$ and find the classifier by solving

$$h_{\mathcal{T}} \in \operatorname*{argmin}_{h \in \mathcal{H}} R_{\mathcal{T}}(h)$$

  where $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ is a fixed class of strategies.

- For example, the Empirical Risk Minimization (ERM) learning uses the empirical risk

$$R_{\mathcal{T}}(h) = \frac{1}{m} \sum_{i=1}^{m} \ell(\boldsymbol{y}^i, h(\boldsymbol{x}^i))$$

7.A: Statistical Consistency

Definition 1. *Let $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a class of strategies. Let $A\colon \cup_{m=1}^{\infty} (\mathcal{X} \times \mathcal{Y})^m \to \mathcal{H}$ be a learning algorithm which returns a classifier $h_{\mathcal{T}} \in \mathcal{H}$ for given training set $\mathcal{T} = ((\boldsymbol{x}^i, \boldsymbol{y}^i) \in \mathcal{X} \times \mathcal{Y} \mid i \in \{1, \ldots, m\}\}$ generated from i.i.d. with $p(\boldsymbol{x}, \boldsymbol{y})$. The algorithm A is statistically consistent in $\mathcal{H}$ w.r.t. $p(\boldsymbol{x}, \boldsymbol{y})$ if for all $\varepsilon > 0$ it holds that*

$$\lim_{m \to \infty} \mathbb{P}_{\mathcal{T} \sim p^m} \left(R(h_{\mathcal{T}}) - R(h_{\mathcal{H}}) \geq \varepsilon \right) = 0$$

If A is consistent for all $p(\boldsymbol{x}, \boldsymbol{y})$ then A is so called universally consistent in $\mathcal{H}$.

In words, with the number of examples going to infinity the estimation error converges in probability to zero.

7.A: Consistency of ERM

The estimation error of ERM can be upper bounded as follows:

$$\begin{aligned} R(h_{\mathcal{T}}) - R(h_{\mathcal{H}}) &= \Big(R(h_{\mathcal{T}}) - R_{\mathcal{T}}(h_{\mathcal{T}})\Big) + \underbrace{\Big(R_{\mathcal{T}}(h_{\mathcal{T}}) - R_{\mathcal{T}}(h_{\mathcal{H}})\Big)}_{\text{non-pos. due to ERM}} + \Big(R_{\mathcal{T}}(h_{\mathcal{H}}) - R(h_{\mathcal{H}})\Big) \\ &\leq \Big(R(h_{\mathcal{T}}) - R_{\mathcal{T}}(h_{\mathcal{T}})\Big) + \Big(R_{\mathcal{T}}(h_{\mathcal{H}}) - R(h_{\mathcal{H}})\Big) \\ &\leq 2 \sup_{h \in \mathcal{H}} \Big|R(h) - R_{\mathcal{T}}(h)\Big| \end{aligned}$$

Therefore $\varepsilon \leq R(h_{\mathcal{T}}) - R(h_{\mathcal{H}})$ implies $\frac{\varepsilon}{2} \leq \sup_{h \in \mathcal{H}} \Big|R(h) - R_{\mathcal{T}}(h)\Big|$ and thus

$$\mathbb{P}_{\mathcal{T} \sim p^m}\Big(R(h_{\mathcal{T}}) - R(h_{\mathcal{H}}) \geq \varepsilon\Big) \leq \mathbb{P}_{\mathcal{T} \sim p^m}\Big(\sup_{h \in \mathcal{H}} \Big|R(h) - R_{\mathcal{T}}(h)\Big| \geq \frac{\varepsilon}{2}\Big)$$

7.A: Uniform convergence of emprical risk

Definition 2. *The empirical risk converges uniformly to the expected risk in $\mathcal{H} \subseteq \mathcal{Y}^{\mathcal{X}}$ if for all $\varepsilon > 0$ it holds that*

$$\lim_{m \to \infty} \mathbb{P}_{\mathcal{T} \sim p^m} \left(\sup_{h \in \mathcal{H}} \left| R(h) - R_{\mathcal{T}}(h) \right| \geq \varepsilon \right) = 0$$

Corrolary: The uniform convergence of the empirical risk in $\mathcal{H}$ implies the consistency of ERM in $\mathcal{H}$ as we see from previous slide.

7.A: Uniform convergence of emprical risk

Two examples of $\mathcal{H}$ when the empirical risk converges uniformly:

1. $\mathcal{H}$ is a finite set. Then,

$$\mathbb{P}_{\mathcal{T}\sim p^m}\left(\max_{h\in\mathcal{H}}\left|R(h)-R_{\mathcal{T}}(h)\right|\geq\varepsilon\right)\leq 2|\mathcal{H}|\exp\left(\frac{-2m\,\varepsilon^2}{l_{max}}\right)$$

   where $l_{max}=\max_{\boldsymbol{y},\boldsymbol{y}'}\ell(\boldsymbol{y},\boldsymbol{y}')$.

   A similar bound for the ML learning was explained in Lecture 6, section 5.

2. $\ell(\boldsymbol{y},\boldsymbol{y}')=[\![\boldsymbol{y}\neq\boldsymbol{y}']\!]$ and VC-dimension of $\mathcal{H}$ is finite.

   VC-dimension (capacity) of $\mathcal{H}$ is the maximal number of inputs which can be classified by strategies from $\mathcal{H}$ in all possible (that is $|\mathcal{Y}|^d$) ways.

7.B: Learning linear classifier from separable examples

Let us consider $\mathcal{H}$ composed of all linear classifiers

$$h(\boldsymbol{x}; \boldsymbol{w}) = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \langle \boldsymbol{w}, \boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{y}) \rangle$$

where $\boldsymbol{w} \in \mathbb{R}^n$ are the parameters and $\boldsymbol{\Psi} \colon \mathcal{X} \times \mathcal{Y} \to \mathbb{R}^n$ is a fixed function (feature map).

Definition 3. *(Linearly separable examples) The examples* $\{(\boldsymbol{x}^i, \boldsymbol{y}^i) \in \mathcal{X} \times \mathcal{Y} \mid i \in \mathcal{I} = \{1, \ldots, m\}\}$ *are called linearly separable if there exists* $\boldsymbol{w} \in \mathbb{R}^n$ *such that*

$$\langle \boldsymbol{w}, \boldsymbol{\Psi}(\boldsymbol{x}^i, \boldsymbol{y}^i) \rangle > \langle \boldsymbol{w}, \boldsymbol{\Psi}(\boldsymbol{x}^i, \boldsymbol{y}) \rangle \,, \quad \forall i \in \mathcal{I}, \boldsymbol{y} \in \mathcal{Y} \setminus \{\boldsymbol{y}^i\}$$

Implementation of ERM: For a loss function $\ell(\boldsymbol{y}, \boldsymbol{y}) = 0$, $\boldsymbol{y} \in \mathcal{Y}$, the linearly separable examples imply that there exists a linear classifier given by $\boldsymbol{w} \in \mathbb{R}^n$ with zero empirical risk, i.e.,

$$h(\boldsymbol{x}^i; \boldsymbol{w}) = \boldsymbol{y}^i \,, \quad \forall i \in \mathcal{I}$$

hence the ERM can be translated to solving a set of $m(|\mathcal{Y}| - 1)$ linear inequalities.

7.B: Perceptron algorithm

Task: given a set of point $\{\boldsymbol{a}^i \in \mathbb{R}^n \mid i \in \mathcal{I}\}$ we want to find $\boldsymbol{w} \in \mathbb{R}^n$ which satisfies the linear inequalities

$$\langle \boldsymbol{w}, \boldsymbol{a}^i \rangle > 0\,, \qquad \forall i \in \mathcal{I}\,.$$

If the task has a solution then the points are linearly separable.

Algorithm 1 Perceptron

1: Set $\boldsymbol{w}^0 = \boldsymbol{0}$, $t \leftarrow 0$

2: $t \leftarrow t + 1$

3: Find index $j \in \mathcal{I}$ of a violating inequality such that

$$\langle \boldsymbol{w}^t, \boldsymbol{a}^j \rangle \leq 0$$

4: If there is no violating inequality return $\boldsymbol{w}^t$. Otherwise update

$$\boldsymbol{w}^{t+1} = \boldsymbol{w}^t + \boldsymbol{a}^j$$

and go to Step 2.

7.B: Convergence of Perceptron

Theorem 1. *For any linearly separable points* $\{\boldsymbol{a}^i \in \mathbb{R}^n \mid i \in \mathcal{I}\}$*, the Perceptron algorithm terminates in*

$$\frac{A^2}{\gamma^2}$$

steps at most where

$$A = \max_{i \in \mathcal{I}} \|\boldsymbol{a}^i\|_2 \qquad \textit{and} \qquad \gamma = \max_{\boldsymbol{w} \in \mathbb{R}^n} \min_{i \in \mathcal{I}} \frac{\langle \boldsymbol{w}, \boldsymbol{a}^i \rangle}{\|\boldsymbol{w}\|_2}$$

Note that the upper bound $\frac{A^2}{\gamma^2}$ does not depend on the number of points $m = |\mathcal{I}|$.

7.B: Perceptron applied for learning linear classifier

Task: Given linearly separable training set $\{(\boldsymbol{x}^i, \boldsymbol{y}^i) \in \mathcal{X} \times \mathcal{Y} \mid i \in \mathcal{I} = \{1, \ldots, m\}\}$ find parameters $\boldsymbol{w} \in \mathbb{R}^n$ of linear classifier

$$h(\boldsymbol{x}; \boldsymbol{w}) = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}} \langle \boldsymbol{w}, \boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{y}) \rangle$$

such that $h(\boldsymbol{x}^i; \boldsymbol{w}) = \boldsymbol{y}^i$, $i \in \mathcal{I}$.

Algorithm 2 Perceptron

1: Set $\boldsymbol{w} \leftarrow \mathbf{0}$

2: Find erroneous example $j \in \mathcal{I}$ such that

$$\boldsymbol{y}^j \neq \hat{\boldsymbol{y}}^j \quad \text{where} \quad \hat{\boldsymbol{y}}^j = h(\boldsymbol{x}^j)$$

3: If there is no erroneous example return $\boldsymbol{w}$. Otherwise update

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \boldsymbol{\Psi}(\boldsymbol{x}^j, \boldsymbol{y}^j) - \boldsymbol{\Psi}(\boldsymbol{x}^j, \hat{\boldsymbol{y}}^j)$$

and go to Step 2.

7.B: Perceptron applied for learning linear classifier

- By Theorem 1 we have a guarantee that for linearly separable training set $\{(\boldsymbol{x}^i, \boldsymbol{y}^i) \in \mathcal{X} \times \mathcal{Y} \mid i \in \mathcal{I}\}$ the Perceptron terminates after at most $\frac{A^2}{\gamma^2}$ iterations where

$$A = \max_{i \in \mathcal{I}, \boldsymbol{y} \in \mathcal{Y} \setminus \{\boldsymbol{y}^i\}} \|\boldsymbol{\Psi}(\boldsymbol{x}^i, \boldsymbol{y}^i) - \boldsymbol{\Psi}(\boldsymbol{x}^i, \boldsymbol{y})\|$$

and

$$\gamma = \max_{\boldsymbol{w} \in \mathbb{R}^n} \min_{i \in \mathcal{I}, \boldsymbol{y} \in \mathcal{Y} \setminus \{\boldsymbol{y}^i\}} \frac{\langle \boldsymbol{w}, \boldsymbol{\Psi}(\boldsymbol{x}^i, \boldsymbol{y}^i) - \boldsymbol{\Psi}(\boldsymbol{x}^i, \boldsymbol{y}) \rangle}{\|\boldsymbol{w}\|_2}$$

- Perceptron requires only the classification oracle $h(\boldsymbol{x}; \boldsymbol{w})$ which does the hard task of searching for the violating inequality among $|\mathcal{Y}|$ options.

7.C: Max-sum classifier

Setting:

- $(\mathcal{V}, \mathcal{E})$ is undirected graph; $\mathcal{V}$ are parts and $\mathcal{E} \subseteq \binom{|\mathcal{V}|}{2}$ pairs of related parts
- each part $v \in \mathcal{V}$ described by observation $x \in \mathcal{X}$ and label $y \in \mathcal{Y}$; $\mathcal{X}$ and $\mathcal{Y}$ are finite
- $q_v \colon \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ quality of label y_v given x_v; $\boldsymbol{q} = (g_v(x, y) \in \mathbb{R} \mid x \in \mathcal{X}, y \in \mathcal{Y}, v \in \mathcal{V})$
- $g_{vv'} \colon \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ quality a label pair $(y_v, y_{v'})$;
  $\boldsymbol{g} = (g_{vv'}(y, y') \in \mathbb{R} \mid (y, y') \in \mathcal{Y}^2, \{v, v'\} \in \mathcal{E})$

Max-sum classifier: Given observations $\boldsymbol{x} = (x_v \in \mathcal{X} \mid v \in \mathcal{V}) \in \mathcal{X}^{\mathcal{V}}$, the max-sum classifier $h \colon \mathcal{X}^{\mathcal{V}} \to \mathcal{Y}^{\mathcal{V}}$ returns labeling $\boldsymbol{y} = (y_v \in \mathcal{Y} \mid v \in \mathcal{V}) \in \mathcal{Y}^{\mathcal{V}}$ with the maximal overall quality

$$h(\boldsymbol{x}; \boldsymbol{q}, \boldsymbol{g}) \in \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}^{\mathcal{V}}} f(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{q}, \boldsymbol{g})$$

where

$$f(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{q}, \boldsymbol{g}) = \sum_{v \in \mathcal{V}} q_v(x_v, y_v) + \sum_{\{v, v'\} \in \mathcal{E}} g_{vv'}(y_v, y_{v'})$$

The max-sum classifier is an instance of the linear classifier since $f(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{q}, \boldsymbol{g}) = \langle \boldsymbol{\Psi}(\boldsymbol{x}, \boldsymbol{y}), \boldsymbol{w} \rangle$ where $\boldsymbol{w} = (\boldsymbol{q}, \boldsymbol{g})$ and $\boldsymbol{\Psi} \colon \mathcal{X}^{\mathcal{Y}} \times \mathcal{Y}^{\mathcal{V}} \to \mathbb{R}^{|\mathcal{Y}||\mathcal{V}| + |\mathcal{E}||\mathcal{Y}|^2}$ is constructed appropriately.

7.C: Relation between Max-sum classifier and Gibbs distribution

- $(\mathcal{V}, \mathcal{E})$ is undirected graph
- $\{(X_v, Y_v) \mid v \in \mathcal{V}\}$ is a field of random variables taking values from $(x_v, y_v) \in X \times \mathcal{Y}, v \in \mathcal{V}$
- the random variables are distributed according to the Gibbs distribution

$$\begin{aligned} p_{\boldsymbol{q},\boldsymbol{g}}(\boldsymbol{x}, \boldsymbol{y}) &= \frac{1}{Z(\boldsymbol{q}, \boldsymbol{g})} \exp\left(\sum_{v \in \mathcal{V}} q_v(x_v, y_v) + \sum_{\{v,v'\} \in \mathcal{E}} g_{vv'}(y_v, y_{v'}) \right) \\ &= \frac{1}{Z(\boldsymbol{q}, \boldsymbol{g})} \exp f(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{q}, \boldsymbol{g}) \end{aligned}$$

- The optimal (Bayes) classifier minimizing the expected risk under the 0/1-loss

$$R(h) = \mathbb{E}_{(\boldsymbol{x},\boldsymbol{y}) \sim p_{\boldsymbol{q},\boldsymbol{g}}} [\![\boldsymbol{y} \neq h(\boldsymbol{x})]\!]$$

is the max-sum classifier

$$h(\boldsymbol{x}; \boldsymbol{q}, \boldsymbol{g}) \in \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}^{\mathcal{V}}} f(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{q}, \boldsymbol{g})$$

7.C: Learning max-sum classifier from linearly separable examples

Task: Given linearly separable training set $\mathcal{T} = \{(\boldsymbol{x}^i, \boldsymbol{y}^i) \in \mathcal{X}^{\mathcal{V}} \times \mathcal{Y}^{\mathcal{V}} \mid i \in \mathcal{I} = \{1, \ldots, m\}\}$ find quality functions $\boldsymbol{q}$, $\boldsymbol{g}$ of the max-sum classifier such that

$$\boldsymbol{y}^i = h(\boldsymbol{x}^i; \boldsymbol{q}, \boldsymbol{g}) = \operatorname*{argmax}_{\boldsymbol{y} \in \mathcal{Y}^{\mathcal{V}}} \left[\sum_{v \in \mathcal{V}} q_v(x^i_v, y_v) + \sum_{\{v,v'\} \in \mathcal{E}} g_{vv'}(y_v, y_{v'}) \right], \qquad i \in \mathcal{I}\,.$$

The max-sum problem $\mathcal{P} = (\mathcal{E}, \mathcal{V}, \boldsymbol{q}, \boldsymbol{g}, \boldsymbol{x})$ associated with the classification $h(\boldsymbol{x}; \boldsymbol{q}, \boldsymbol{g})$ is tractable if:

1. $(\mathcal{V}, \mathcal{E})$ is acyclic graph

2. $\mathcal{Y}$ is fully ordered and $-g_{vv'}$, $\{v, v'\} \in \mathcal{E}$ are submodular w.r.t the ordering: for each $(y_v, y'_v, y_{v'}, y'_{v'}) \in \mathcal{Y}^4$ such that $y_v > y'_v$ and $y_{v'} > y'_{v'}$ it following inequality holds

$$g_{vv'}(y_v, y_{v'}) + g_{vv'}(y'_v, y'_{v'}) \leq g_{vv'}(y_v, y'_{v'}) + g_{vv'}(y'_v, y_{v'})$$

3. $\mathcal{P} = (\mathcal{V}, \mathcal{E}, \boldsymbol{q}, \boldsymbol{g}, \boldsymbol{x})$ have a strictly trivial equivalent, that is, the LP relaxation is tight and the max-sum problem has unique solution