Apprentissage, réseaux de neurones et modèles graphiques (RCP209)
Structured Prediction

Nicolas Thome
Prenom.Nom@cnam.fr
http://cedric.cnam.fr/vertigo/Cours/ml2/

Département Informatique
Conservatoire National des Arts et Métiers (Cnam)

Outline

1 From Binary to Strutured Prediction

2 Training Formulation

3 Optimization

4 Instanciations

Apprentissage supervisé

Prédiction binaire *vs* prédiction stucturée

- Prédiction binaire : données d'éntrée $x \in \mathcal{X}$, fonction de prédiction $\hat{y}(x, w) = f_w(x) =$ (par exemple) $= \text{sign}\left[\langle w, x\rangle(+b)\right]$
  - Prédiction $\hat{y}(x, w) \in \{-1; +1\}$
- Prédiction strcuturée : données d'éntrée $x \in \mathcal{X}$, fonction de prédiction $\hat{y}(x, w) \in \mathcal{Y}$
  - $\mathcal{Y}$ quelconque (pas scalaire) : vecteur, chaîne, graphe, *etc*

Apprentissage supervisé

Classification Binaire

- $x \in \mathbb{R}^d$, fonction de prédiction : $\hat{y}(x, w) = f_w(x) = \text{sign}\left[\langle w, x \rangle (+b)\right]$.
- Schéma d'apprentissage : optimiser w à partir d'un ensemble d'apprentissage $\{x_i\}_{i=1,..,N}$:

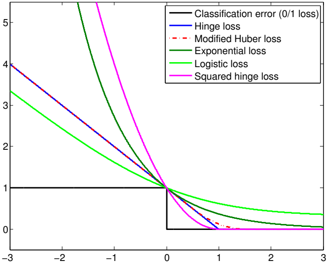
$$\mathcal{P}(w) = R(w) + \frac{C}{N} \sum_{i=1}^{N} \Delta\left[\hat{y}(x_i, w), y_i\right]$$

- Δ : fonction de coût entre y_i et $\hat{y}(x_i, w)$. Classif: "vrai" loss: 0-1 loss : $1_{\hat{y}y_i<0}$
- MAIS 0-1 loss est NP hard à optimiser $\Rightarrow$ surrogate loss

Classification Binaire

0-1 loss surrogate

- Surrogate loss : trouver une borne supérieur convexe $L_i(w)$ tq : $\Delta\left[\hat{y}(x_i, w), y_i\right] \leq L_i(w)$
- Ex: hinge loss : $\left[\langle w, x_i \rangle y_i\right]_+ = \max\left[0, 1 - \langle w, x_i \rangle y_i\right]$



$$\mathcal{P}(w) = R(w) + \frac{C}{N} \sum_{i=1}^{N} \max\left[0, 1 - \langle w, x_i \rangle y_i\right]$$

Classification Binaire

Régularisation

- Régularisation avec norme ℓ_2 : $R(w) = \frac{1}{2}||w||^2$
- Cas du SVM : se prémunir contre le sur-apprentissage

$$\mathcal{P}(w) = \frac{1}{2}||w||^2 + \frac{C}{N}\sum_{i=1}^{N} \max\left[0, 1 - \langle w, x_i \rangle y_i\right]$$

  - Bornes de généralisation
  - Vision "smooth" de la régularisation: Cauchy-Schwarz:

$$\langle w, (x - x') \rangle \leq ||w||^2 ||x - x'||^2$$

  - Une petite variation sur l'entrée x ne doit pas "trop" impacter le score $\langle w, x \rangle$. $||w||^2$ est une borne pour cette variation

Prédiction structurée (Structured output prediction)

Définition

- $\mathcal{X}$ est l'espace d'entrée : arbitraire (pas nécessairement vectorial, *etc*)
- $\mathcal{Y}$ est l'espace de sortie structuré
  - Discret
  - Structuré: vecteur, chaîne, arbre, graphe, *etc*
- **Préliminaire :** définir une fonction décrivant la relation entre l'entrée $\mathbf{x}$ et la sortie $\mathbf{y}$("joint feature map") : $\Psi(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^d$
- **Modèle :** apprendre un modèle de prédiction linéaire qui assigne le score $\langle w, \Psi(\mathbf{x}, \mathbf{y}) \rangle$ à chaque paire $(\mathbf{x}, \mathbf{y})$
  - Extension à des fonctions non linéaire possible (kernelisation)
- Fonction de prédiction :

$$\hat{y}(x, w) = \arg\max_{y \in \mathcal{Y}} \langle w, \Psi(x, y) \rangle$$

Outline

1 From Binary to Strutured Prediction

2 Training Formulation

3 Optimization

4 Instanciations

Prédiction structurée

Formulation de l'apprentissage

- On souhaite minimiser la fonction objectif suivante:

$$\mathcal{P}(w) = \frac{1}{2}||w||^2 \quad \text{s.t.:}$$

  - $\forall x_i,\ i \in \{1, .., N\}\ \forall y \neq yi$: $\langle w, \Psi(x_i, y_i)\rangle \geq \langle w, \Psi(x_i, y)\rangle + 1$
  - Marge de of 1 comme pour les SVM binaires

- Variables "ressort" ξ_i (slack variables) :

$$\mathcal{P}(w) = \frac{1}{2}||w||^2 + \frac{C}{N(|\mathcal{Y}| - 1)} \sum_{i=1}^{N} \sum_{y \neq y_i} \xi_i \quad \text{s.t.:}$$

$\forall x_i,\ \forall y \neq yi$: $\langle w, \Psi(x_i, y_i)\rangle \geq \langle w, \Psi(x_i, y)\rangle + 1 - \xi_i$

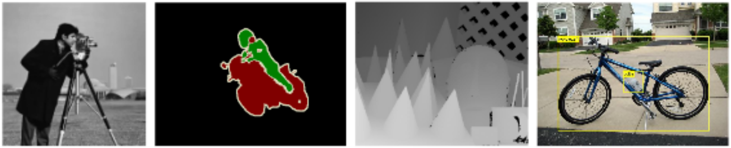
  - Problème d'optimisation convexe

Prédiction structurée

Formulation de l'apprentissage

- $\forall x_i, \; \forall y \neq yi$: $\langle w, \Psi(x_i, y_i) \rangle \geq \langle w, \Psi(x_i, y) \rangle + 1 - \xi_i$
- La marge de 1 est indépendante de la différence entre y and y_i
- Idée : utiliser $\Delta(y, y_i)$ pour refléter la connaissance *a priori* pour mesurer la différence entre y and y_i

Choix de la fonction de coût $\Delta(y, y_i)$: dépend de l'application



Prédiction structurée

Margin rescaling [TJHA05b]

- Plutôt que d'avoir une marge fixée à 1, fixer la marge $\Delta(y, y_i)$: $\forall x_i$, $\forall y \neq yi : \langle w, \Psi(x_i, y_i)\rangle \geq \langle w, \Psi(x_i, y)\rangle + \Delta(y, y_i) - \xi_i$
- On peut montrer que la fonction objectif dans le cas du margin rescaling s'écrit :

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\left[\max_{\mathbf{y}\in\mathcal{Y}} \left[\Delta(\mathbf{y}_i, \mathbf{y}) + \langle \Psi(\mathbf{x}_i, \mathbf{y}), \mathbf{w}\rangle\right] - \langle \Psi(\mathbf{x}_i, \mathbf{y}_i), \mathbf{w}\rangle\right]$$

Prédiction structurée

Margin rescaling [TJHA05b]

- Autre façon de voir : la fonction objectif "idéale" serait :

$$P(w) = \min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\Delta(\mathbf{y}_i, \hat{\mathbf{y}})$$

- MAIS optimiser $\Delta(\mathbf{y}_i, \hat{\mathbf{y}})$ est NP hard $\Rightarrow$ surrogate loss
- Exercice : montrer que $\left[\max_{\mathbf{y}\in\mathcal{Y}} \left[\Delta(\mathbf{y}_i, \mathbf{y}) + \langle\Psi(\mathbf{x}_i, \mathbf{y}), \mathbf{w}\rangle\right] - \langle\Psi(\mathbf{x}_i, \mathbf{y}_i), \mathbf{w}\rangle\right]$ est une borne sup convexe à $\Delta(\mathbf{y}_i, \hat{\mathbf{y}})$

Prédiction structurée

Slack rescaling [TJHA05b]

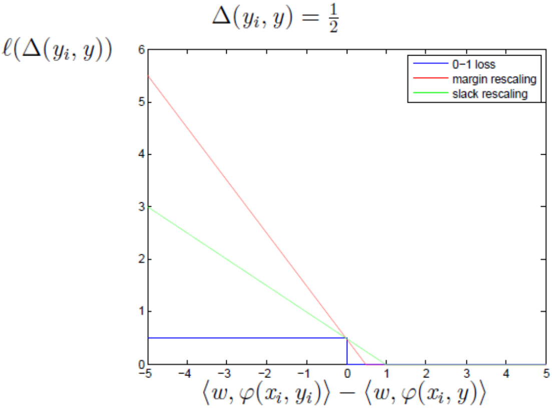
- $\forall x_i$, $\forall y \neq yi$: $\langle w, \Psi(x_i, y_i)\rangle \geq \langle w, \Psi(x_i, y)\rangle - \frac{\xi_i}{\Delta(y, y_i)}$
- La fonction objectif devient

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n} \max_{\mathbf{y}\in\mathcal{Y}} \left[\Delta(\mathbf{y}_i, \mathbf{y})\left(1 + \langle\Psi(\mathbf{x}_i, \mathbf{y}), \mathbf{w}\rangle - \langle\Psi(\mathbf{x}_i, \mathbf{y}_i), \mathbf{w}\rangle\right)\right] \qquad (1)$$

- Comme dans le cas du margin rescaling, on peut montrer que $\max_{\mathbf{y}\in\mathcal{Y}} \left[\Delta(\mathbf{y}_i, \mathbf{y})\left(1 + \langle\Psi(\mathbf{x}_i, \mathbf{y}), \mathbf{w}\rangle - \langle\Psi(\mathbf{x}_i, \mathbf{y}_i), \mathbf{w}\rangle\right)\right]$ est une borne sup convexe à $\Delta(\mathbf{y}_i, \hat{\mathbf{y}})$

Prédiction structurée

Margin *vs* Slack rescaling



- Slack rescaling is scale invariant
- Margin rescaling is the most commonly used

Outline

1 From Binary to Strutured Prediction

2 Training Formulation

3 Optimization
- Structural SVM: Optimisation

4 Instanciations

Structural SVM - Optimisation

Différentes formulations équivalentes :

Formulation 1: n-slack

$$\min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\left[\max_{\mathbf{y}\in\mathcal{Y}} \left[\Delta(\mathbf{y}_i,\mathbf{y}) + \langle\Psi(\mathbf{x}_i,\mathbf{y}),\mathbf{w}\rangle\right] - \langle\Psi(\mathbf{x}_i,\mathbf{y}_i),\mathbf{w}\rangle\right]$$

ou:

$$\min_{\mathbf{w},\xi} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i$$

$$\forall i \in \{1,\dots,n\} \quad \max_{\mathbf{y}\in\mathcal{Y}} \left[\Delta(\mathbf{y}_i,\mathbf{y}) + \langle\Psi(\mathbf{x}_i,\mathbf{y}),\mathbf{w}\rangle\right] - \langle\Psi(\mathbf{x}_i,\mathbf{y}_i),\mathbf{w}\rangle \leq \xi_i$$

- n contraintes non-linéaires (à cause du max) ☹

Optimisation

Formulation 2: n-slack

$$\min_{\mathbf{w},\xi} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i \quad (2)$$

$$\forall i \in \{1,\ldots,n\}, \quad \forall \mathbf{y} \in \mathcal{Y}: \quad \langle \mathbf{w}; \delta\Psi(\mathbf{x}_i,\mathbf{y}_i,\mathbf{y})\rangle \geq \Delta(\mathbf{y}_i,\mathbf{y}) - \xi_i, \quad \xi_i \geq 0 \quad (3)$$

avec $\delta\Psi(\mathbf{x}_i,\mathbf{y}_i,\mathbf{y}) = \Psi(\mathbf{x}_i,\mathbf{y}_i) - \Psi(\mathbf{x}_i,\mathbf{y})$

- $n|\mathcal{Y}|$ contraintes linéaires ☹

Optimisation

Formulation 3: one-slack

$$\min_{\mathbf{w},\xi} \frac{1}{2}\|\mathbf{w}\|^2 + C\xi$$

$$\forall (\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_n) \in \mathcal{Y} \times \ldots \times \mathcal{Y}$$

$$\sum_{i=1}^{n} \left[\Delta(\mathbf{y}_i, \hat{\mathbf{y}}_i) + \langle \Psi(\mathbf{x}_i, \hat{\mathbf{y}}_i), \mathbf{w} \rangle - \langle \Psi(\mathbf{x}_i, \mathbf{y}_i), \mathbf{w} \rangle\right] \leq n\xi \tag{4}$$

- Seulement 1 slack variable
- $|\mathcal{Y}|^n$ contraintes linéaires ☹

Optimisation

Différentes formulations : conclusion

- Formulation 1 a seulement n contraintes: peut être résolu directement
- Formulations 2 et 3 conduisent à un nombre de contraintes gigantesque. Exemple: n=100 images binaires de taille 16×16
  - Formulation 1: 100 contraintes
  - Formulation 2: $\sim 10^{79}$ contraintes
  - Formulation 3: $\sim 10^{177}$ contraintes
- MAIS pour Formulations 2 & 3: enforcer des contraintes "actives" est suffisant $\Rightarrow$ working set training

Optimisation

Résolution de la formulation 1

Descente de (sous)-gradient

Algorithm 1 Algorithme d'apprentissage structuré

Entrées : Paires d'apprentissage $\{(\mathrm{x_i}; \mathrm{y_i}) \in \mathcal{X} \times \mathcal{Y}\}_{i \in \{1;N\}}$, joint feature map $\Psi(\mathrm{x}, \mathrm{y}) \in \mathbb{R}^d$, loss $\Delta(\mathrm{y}, \mathrm{y'})$, régulariseur C, nombre d'itérations T, pas de gradient η_t.

Sortie : $\mathbf{w} \Rightarrow$ fonction de prédiction $f_{\mathbf{w}}(x) = \arg\max_{y \in \mathcal{Y}} \langle w, \Psi(x, y) \rangle$.

1: $\mathbf{w} \leftarrow \vec{0}$ // Initialisation
2: **for** $t = 1, \ldots, T$ **do**
3: **for** $i = 1, \ldots, N$ **do**
4: $(x_i; y_i) \leftarrow$ Sélection aléatoire d'une paire d'apprentissage
5: $\hat{y} \leftarrow \arg\max_{y \in \mathcal{Y}} [\Delta(\mathrm{y}, \mathrm{y_i}) + \langle \Psi(x_i, y), w \rangle]$ // "Loss-augmented inference"
6: $g_i = \Psi(x_i, \hat{y}) - \Psi(x_i, y_i)$ // Calcul du gradient
7: **end for**
8: $\mathbf{w} \leftarrow w - \eta_t (w + \frac{C}{N} \sum_{i=1}^{N} g_i)$ // Mise à jour
9: **end for**
10: **return** $\mathbf{w}$

Optimisation

Résolution de la formulation 1

Descente de (sous)-gradient stochastique : en TME !

Algorithm 2 Algorithme d'apprentissage structuré

Entrées : Paires d'apprentissage $\{(\mathbf{x_i}; \mathbf{y_i}) \in \mathcal{X} \times \mathcal{Y}\}_{i \in \{1;N\}}$, joint feature map $\Psi(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^d$, loss $\Delta(\mathbf{y}, \mathbf{y'})$, régulariseur C, nombre d'itérations T, pas de gradient η_t.
Sortie : $\mathbf{w}$ $\Rightarrow$ fonction de prédiction $f_w(x) = \arg\max_{y \in \mathcal{Y}} \langle w, \Psi(x, y) \rangle$.

1: $\mathbf{w} \leftarrow \vec{0}$ // Initialisation
2: **for** $t = 1, \ldots, T$ **do**
3:     **for** $i = 1, \ldots, N$ **do**
4:         $(x_i; y_i) \leftarrow$ Sélection aléatoire d'une paire d'apprentissage
5:         $\hat{y} \leftarrow \arg\max_{y \in \mathcal{Y}} [\Delta(\mathbf{y}, \mathbf{y_i}) + \langle \Psi(x_i, y), w \rangle]$ // "Loss-augmented inference"
6:         $g_i = \Psi(x_i, \hat{y}) - \Psi(x_i, y_i)$ // Calcul du gradient
7:         $\mathbf{w} \leftarrow w - \eta_t (w + C g_i)$ // Mise à jour
8:     **end for**
9: **end for**
10: **return** $\mathbf{w}$

Optimisation

Résolution de la formulation 1 : cutting plane

$$L(w) = \min_{\mathbf{w}} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\left[\max_{\mathbf{y}\in\mathcal{Y}} \left[\Delta(\mathbf{y}_i,\mathbf{y}) + \langle\Psi(\mathbf{x}_i,\mathbf{y}),\mathbf{w}\rangle\right] - \langle\Psi(\mathbf{x}_i,\mathbf{y}_i),\mathbf{w}\rangle\right]$$

- $L(w)$: fonction convexe, non dérivable, avec une contante de Lipschitz bornée (i.e., ne varie pas trop vite).
  - Optimisation de ce genre de fonctions : intensivement étudiée en recherche opérationnelle
- Alternative à la descente de gradient : Bundle Method for Regularized Risk Minimization (BMRM)
  - Cas particulier des bundle method for regularised loss functions, qui sont une version stabilisée du cutting plane.

Optimisation

Résolution de la formulation 2 [TJHA05a]

$$\min_{\mathbf{w},\xi} \frac{1}{2}\|\mathbf{w}\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi_i \text{ s.t.}$$

$\forall i \in \{1,\ldots,n\}, \forall \mathbf{y} \in \mathcal{Y}$:

$\langle \mathbf{w}; \delta\Psi(\mathbf{x}_i,\mathbf{y}_i,\mathbf{y})\rangle \geq \Delta(\mathbf{y}_i,\mathbf{y}) - \xi_i$,

$\xi_i \geq 0$

$\delta\Psi(\mathbf{x}_i,\mathbf{y}_i,\mathbf{y}) = \Psi(\mathbf{x}_i,\mathbf{y}_i) - \Psi(\mathbf{x}_i,\mathbf{y})$

Working Set S-SVM Training

input training pairs $\{(x^1,y^1),\ldots,(x^n,y^n)\} \subset \mathcal{X}\times\mathcal{Y}$,
input feature map $\varphi(x,y)$, loss function $\Delta(y,y')$, regularizer C

1: $S \leftarrow \emptyset$
2: **repeat**
3: $(w,\xi) \leftarrow$ *solution to QP only with constraints from* S
4: **for** i=1,. . . ,n **do**
5: $\hat{y} \leftarrow \operatorname{argmax}_{y\in\mathcal{Y}} \Delta(y^i,y) + \langle w, \varphi(x^i,y)\rangle$
6: **if** $\hat{y} \neq y^i$ **then**
7: $S \leftarrow S \cup \{(x^i,\hat{y})\}$
8: **end if**
9: **end for**
10: **until** S doesn't change anymore.

output prediction function $f(x) = \operatorname{argmax}_{y\in\mathcal{Y}}\langle w,\varphi(x,y)\rangle$.

1. Ligne 3 : résolue par (S)GD ou cutting plane
2. Algorithme garanti de converger vers la même solution qu'avec l'ensemble des contraintes, à une précision ϵ
   - Pour une précision ϵ fixée, seulement $O(n/\epsilon^2)$ contraintes requises **(Tsochantaridis *et.al*, JMLR 2005)**

Optimisation

Résolution de la formulation 3 [JFY09]

$$\min_{\mathbf{w},\xi} \frac{1}{2}\|\mathbf{w}\|^2 + C\xi \text{ s.t.}$$

$\forall(\hat{\mathbf{y}}_1, \ldots, \hat{\mathbf{y}}_n) \in \mathcal{Y} \times \ldots \times \mathcal{Y}$:

$\sum_{i=1}^{n} [\Delta(\mathbf{y}_i, \hat{\mathbf{y}}_i) + \langle \delta\Psi(\mathbf{x}_i, \mathbf{y}_i, \hat{\mathbf{y}}), \mathbf{w}\rangle] \leq n\xi$,

$\xi_i \geq 0$,

$\delta\Psi(\mathbf{x}_i, \mathbf{y}_i, \hat{\mathbf{y}}) = \Psi(\mathbf{x}_i, \mathbf{y}_i) - \Psi(\mathbf{x}_i, \hat{\mathbf{y}})$

Working Set One-Slack S-SVM Training

input training pairs $\{(x^1, y^1), \ldots, (x^n, y^n)\} \subset \mathcal{X} \times \mathcal{Y}$,
input feature map $\varphi(x, y)$, loss function $\Delta(y, y')$, regularizer C

1: $S \leftarrow \emptyset$
2: **repeat**
3: $(w, \xi) \leftarrow$ *solution to QP only with constraints from* S
4: **for** i=1,...,n **do**
5: $\hat{y}^i \leftarrow \operatorname{argmax}_{y \in \mathcal{Y}} \Delta(y^i, y) + \langle w, \varphi(x^i, y)\rangle$
6: **end for**
7: $S \leftarrow S \cup \{((x^1, \ldots, x^n), (\hat{y}^1, \ldots, \hat{y}^n))\}$
8: **until** S doesn't change anymore.

output prediction function $f(x) = \operatorname{argmax}_{y \in \mathcal{Y}} \langle w, \varphi(x, y)\rangle$.

Often faster convergence:
We add one strong constraint per iteration instead of n weak ones.

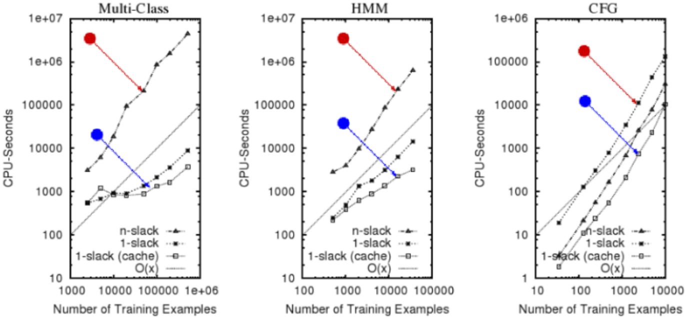
[Joachims, Finley, Yu: "Cutting-Plane Training of Structural SVMs", Machine Learning, 2009]

optimisation avec one-slack généralement plus rapide [JFY09]

Optimisation

Résolution de la formulation 3

Training Times ordinary S-SVM versus one-slack S-SVM



Observation 1: One-slack training is usually faster than n-slack.
Observation 2: Training structured models is nevertheless **slow**.

Figure: [Joachims, Finley, Yu: "Cutting-Plane Training of Structural SVMs", Machine Learning, 2009]

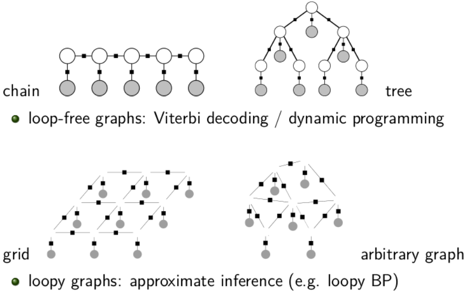
Optimisation

Inference et "loss-augmented" inference

Quelle que soit la formulation :

- Calcul de l'inférence $\hat{y}(x, w) = \underset{y \in \mathcal{Y}}{\arg\max} \langle w, \Psi(x, y) \rangle$ nécessaire en test
- Calcul du "loss-augmented inference" (ou "separation oracle") $\hat{y}(x, w) = \underset{y \in \mathcal{Y}}{\arg\max} (\langle w, \Psi(x, y) \rangle + \Delta(y, y_i))$ nécessaire en apprentissage
- Souvent le goulot d'étranglement
  - Pour des problèmes structurés avec $|\mathcal{Y}|$ grand, recherche exhaustive impossible !

Inference et "loss-augmented" inference : perspectives



- Segmentation:
  - inférence exacte dans des cas particuliers : structure chaîne, arbre, fonctions sub-modulaires pour les énergies
  - Sinon inférence approximée (*e.g.* loopy BP)
- Inference approximée : quelle garantie sur la convergence globale ?? Question ouverte...

Apprentissage structuré

Conclusion

- $\oplus$ Formalisme général pour le problème, élégant, de nombreuses instantiations
- $\oplus$ Problème d'optimisaton convexe
- $\ominus$ Convexifcation (slack/margin rescaling) $\Rightarrow$ modèle (trop) simple
- $\oplus$ Des algorithmes efficaces pour résoudre
- $\ominus$ Relativement lent pour des problèmes à forte strcuture
- $\oplus$ SSVM proche des Conditionnal Random Field (CRF)
- $\ominus$ Entrée sortie observés $\Rightarrow$ pas d'apprentissage de paramètres cachés (représentations)

Outline

1 From Binary to Strutured Prediction

2 Training Formulation

3 Optimization

4 Instanciations

SSVM : Instanciation

Rappel : ingrédient d'un problème d'apprentissage structuré

- Fonction de prédiction nécessite : $\Psi(x, y)$
- Apprentissage : $\Psi(x, y)$ et $\Delta(\mathbf{y}_i, \mathbf{y})$

Exemples classiques d'instantiations SSVM en vision

- Classification binaire
- Classification Multi-classes
- Classification hiérarchique
- Détection d'objets
- Ranking
- Estimation de pose
- Segmentation d'images (segmentation sémantique)
- Prédiction de séquences

SSVM : Instanciation

Classification Binaire

- $\Psi(x, y) = \frac{1}{2} y \cdot x, \; y \in \{-1; 1\}$
- $\Delta(y_i, y) = 1_{y_i \neq y}$
- Exercice: $\hat{y}$ =?, schéma d'apprentissage ?

SSVM : instantiation

Classification multi-classes

- $\mathcal{Y} = \{1, 2, \ldots, K\}, \quad \Delta(y, y') = \begin{cases} 1 & \text{for } y \neq y' \\ 0 & \text{otherwise.} \end{cases}$
- $\varphi(x, y) = \Big([\![y = 1]\!]\Phi(x),\ [\![y = 2]\!]\Phi(x),\ \ldots,\ [\![y = K]\!]\Phi(x) \Big)$
  $= \Phi(x) e_y^\top \qquad$ with e_y=y-th unit vector

Solve:

$$\min_{w,\xi} \frac{1}{2}\|w\|^2 + \frac{C}{n}\sum_{i=1}^{n} \xi^i$$

subject to, for $i = 1, \ldots, n$,

$$\langle w, \varphi(x^i, y^i)\rangle - \langle w, \varphi(x^i, y)\rangle \geq 1 - \xi^i \quad \text{for all } y \in \mathcal{Y} \setminus \{y^i\}.$$

Classification: MAP $\quad f(x) = \operatorname*{argmax}_{y \in \mathcal{Y}} \ \langle w, \varphi(x, y)\rangle$

Crammer-Singer Multiclass SVM

SSVM : instantiation classification multi-classes

Connections

- Cas particulier de cette instanciation avec 2 classes ?
- Lien avec la classification binaire ?

Inference et "loss-augmented" inference

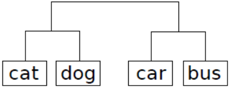
- Classification multi-classes et hiérarchique
  - $|\mathcal{Y}| = K$ où K est le nombre de classes
  - Dans ce cas, il est envisageable de calculer l'inférence où le loss-augmented inference exhaustivement

SSVM : instantiation

Classification hiérarchique

Hierarchical Multiclass Classification

Loss function can reflect hierarchy:



$$\Delta(y, y') := \frac{1}{2}(\text{distance in tree})$$

$$\Delta(\text{cat}, \text{cat}) = 0, \quad \Delta(\text{cat}, \text{dog}) = 1, \quad \Delta(\text{cat}, \text{bus}) = 2, \quad \textit{etc.}$$

Solve:

$$\min_{w,\xi} \frac{1}{2}\|w\|^2 + \frac{C}{n}\sum_{i=1}^{n}\xi^i$$

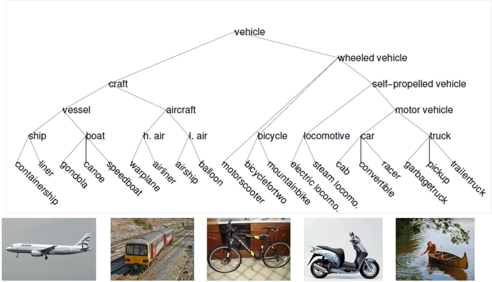
subject to, for $i = 1, \dots, n,$

$$\langle w, \varphi(x^i, y^i)\rangle - \langle w, \varphi(x^i, y)\rangle \geq \Delta(y^i, y) - \xi^i \quad \text{for all } y \in \mathcal{Y} \setminus \{y^i\}.$$

SSVM : instantiation

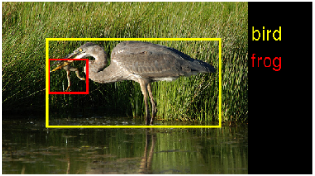
Classification hiérarchique

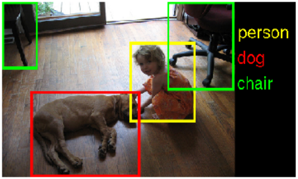
- Comment définir $\Delta(\mathbf{y}_i, \mathbf{y})$?
- Wordnet $\Rightarrow$ distance sémantique entre classes

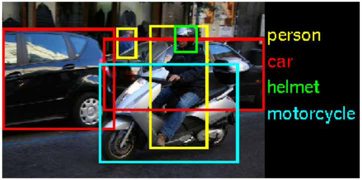


SSVM : instantiation

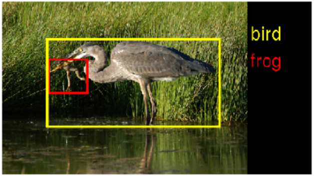
Détection d'objet : fonction de prédiction $\Rightarrow \Psi(x, y) = ?$

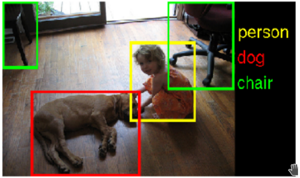






Détection d'objet : apprentissage $\Rightarrow \Delta(\mathbf{y}_i, \mathbf{y}) = ?$

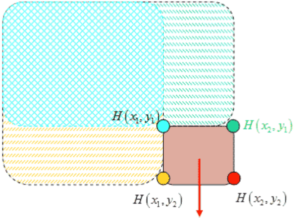




SSVM : instantiation pour la Détection d'objet

Inference et "loss-augmented" inference

- Détection d'objet : $|\mathcal{Y}|$: nombre de régions possibles
  - Approche classique : fenêtre glissante $\Rightarrow$ très grand nb de régions candidates
  - Car on parcours à différentes positions, échelles, ratio (voire rotation)
- Pour accélérer le calcul : "histogrammes intégrales"
  - Permet de calculer l'hostogramme (BoW) en tps contsant



- Mais nécessite tjs l'évaluation de tous les produits scalaires
- autres méthodes : couper l'espace de recherche
  - stratégie branch and bound [BL08]

SSVM : instantiation

Ranking : fonction de prédiction $\Rightarrow \Psi(x,y) = ?$

- Entrée $\mathcal{X}$: liste d'éléments, sortie $\mathcal{Y}$ ordonnacement de ces éléments
  - $\mathbf{x} = (o_1, \dots o_n)$
  - Ordonnacement peut être représenté par une matrice $\mathbf{Y}$ tq :

$$y_{ij} = \begin{cases} +1 & \text{si } o_i \prec_y o_j \ (o_i \text{ est classé avant } o_j \text{ dans la liste ordonnée}) \\ -1 & \text{si } o_i \succ_y o_j \ (o_i \text{ est classé après } o_j) \end{cases}$$

  - Ranking feature map:

$$\Psi(x,y) = \sum_{i \in \oplus} \sum_{j \in \ominus} y_{ij} \left(\phi(o_i) - \phi(o_j) \right)$$

    - On apprend $\mathbf{w}$ pour que les elts soient ordonnées comme le vrai ranking
    - $\phi(o_i)$: description de o_i, *e.g.* $\phi(o_i) = b_i \in \mathbb{R}^d$ BoW ou feature deep (image)
- Question : comment déterminer $\hat{y}(x,w) = \arg\max_{y \in \mathcal{Y}} \langle w, \Psi(x,y) \rangle$?

SSVM : instantiation

Ranking : fonction de prédiction $\Rightarrow \Psi(x,y) = ?$

- Entrée $\mathcal{X} = \{q\}$: ensemble des requêtes q, sortie $\mathcal{Y}$: ordonnacement des éléments du corpus par rapport à chaque requête q.
  - Ranking feature map [YFRJ07] :

$$\langle \mathbf{w}; \Psi(x,y) \rangle = \sum_{i \in \mathcal{X}_{\boldsymbol{q}}^+} \sum_{j \in \mathcal{X}_{\boldsymbol{q}}^-} y_{ij} \langle \phi(q, o_i) - \phi(q, o_j); \mathbf{w} \rangle$$

  - Ordonnacement peut être représenté par une matrice $\mathbf{Y}$ tq :

$$y_{ij} = \begin{cases} +1 & \text{si } i <_y j \text{ (} i \text{ est classé avant } j \text{ dans la liste ordonnée / } q\text{)} \\ -1 & \text{si } i >_y j \text{ (} i \text{ est classé après } j\text{)} \end{cases}$$

    - On apprend $\mathbf{w}$ pour que les elts soient ordonnées comme le vrai ranking
    - $\phi(q, o_i)$: traduit la relation entre la reqêute et le document i, *e.g.* $\phi(q, o_i) = b_{\boldsymbol{q}} - b_{\boldsymbol{i}}$, où $b_{\boldsymbol{q}} \in \mathbb{R}^{\boldsymbol{d}}$ est la représentation de q (resp. $b_{\boldsymbol{i}}$ pour i)

SSVM : instantiation

Ranking et apprentissage de distance

$$\langle \mathbf{w}; \Psi(x,y)\rangle = \sum_{i \in \mathcal{X}_{\boldsymbol{q}}^+} \sum_{j \in \mathcal{X}_{\boldsymbol{q}}^-} y_{ij} \langle \phi(q,o_i) - \phi(q,o_j); \mathbf{w}\rangle$$

- Si on pose $\phi(q,o_i) = (b_q - b_i)(b_q - b_i)^T$ [ML10]:

$$\langle \phi(q,o_i); \mathbf{w}\rangle = -(b_q - b_i)^T \mathbf{w}(b_q - b_i) = -d_{\mathbf{w}}(b_q; b_i) \tag{5}$$

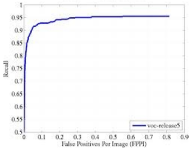
- $\mathbf{w}$ s'interprète alors comme la matrice de distance (Mahalanobis)
  - On veut que $d_{\mathbf{w}}(b_q; b_i) < d_{\mathbf{w}}(b_q; b_j)$ si $yij = +1$
- Souvent : imposer $\mathbf{w}$ semi-definie positive (SDP) pour avoir une vraie métrique
- Modifie l'espace des solutions (cÃťne SDP) lors de l'algorithme d'apprentissage

SSVM : instantiation

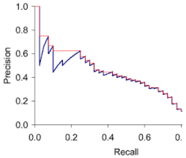
Ranking : apprentissage $\Rightarrow \Delta(\mathbf{y}_i, \mathbf{y}) = ?$

- $\Delta(\mathbf{y}_i, \mathbf{y}) = 1 - R(y)$, $R(y)$ mesure la qualité de ranking, *e.g.*
  - AUC, aire sous la courbe courbe ROC (faux positifs vs vais positifs)
    - $\Delta(\mathbf{y}_i, \mathbf{y}) = \frac{1}{n^2} \sum_{ij} y_{ij} \neq y_{ij}$
    - Ne tient pas compte de la position
  - Précision-at-k : ratio de doc pertinents retournés parmi les k premiers
  - Mean Average Precision (MAP), courbe sous la courbe rappel précision
  - *etc*

ROC



MAP



- Question : comment déterminer $\arg\max_{y \in \mathcal{Y}} \left[\langle w, \Psi(x,y)\rangle + \Delta(y, y_i)\right]$?

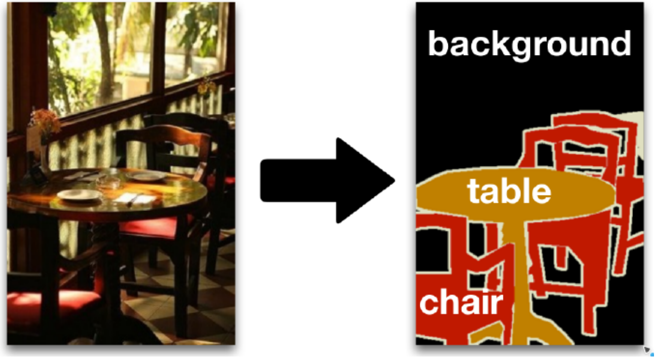
SSVM : instantiation en Ranking

Inference et "loss-augmented" inference

- $\Psi(x,y) = \sum_{ij} y_{ij} \langle \phi(o_i) - \phi(o_j); \mathbf{w} \rangle$
- Inférence : comment déterminer $\hat{y}(x,w) = \underset{y \in \mathcal{Y}}{\arg\max} \langle w, \Psi(x,y) \rangle$?
  - On peut montrer que $\hat{y}$ revient à trier par ordre décroissant de $\langle \mathbf{w}; \phi(o_i) \rangle$
- "Loss-augmented" inference : $\underset{y \in \mathcal{Y}}{\arg\max} \left[\langle w, \Psi(x,y) + \Delta(y, y_i) \rangle \right]$?
  - Dépend de $\Delta(y, y_i)$
  - Des algorithmes existent pour AUC, MAP (plus compliqué) [YFRJ07]

SSVM : instantiation

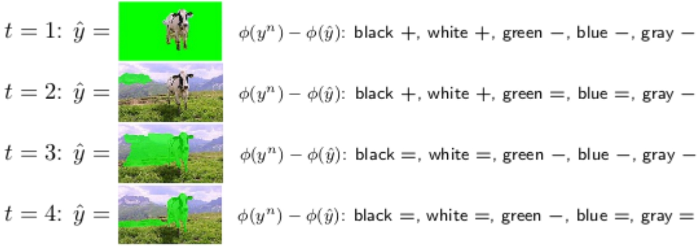
Segmentation sémantique d'images : prédiction $\Rightarrow \Psi(x, y) = ?$



SSVM : instantiation

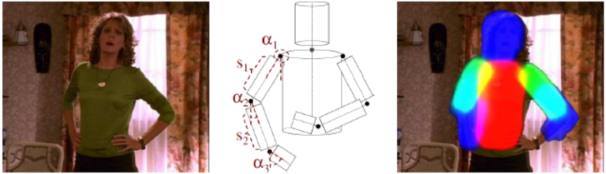
Segmentation sémantique d'images : $\Delta(\mathbf{y}_i, \mathbf{y}) = ?$

- Données d'apprentissage : $\mathcal{X}$ image, $\mathcal{Y}$ masque de segmentation
- Loss utilisé : Hamming loss $\Delta(\mathbf{y}_i, \mathbf{y}) = \frac{1}{|V|} I(\mathbf{y}_i \neq \mathbf{y})$
  - Compte le ratio de pixels mal étiquetés



SSVM : instantiation

Estimation de pose : $\Delta(\mathbf{y}_i, \mathbf{y})$?



input: image — body model — output: model fit

- input space $\mathcal{X} = \{images\}$
- output space $\mathcal{Y} = \{positions/angle\ of\ K\ body\ parts\} \triangleq \mathbb{R}^{3K}$.
- prediction function: $f : \mathcal{X} \rightarrow \mathcal{Y}$

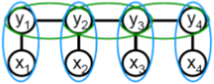
$$f(x) := \operatorname*{argmin}_{y \in \mathcal{Y}} E(x, y)$$

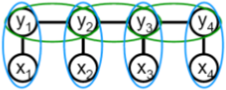
- energy $E(x, y) = \sum_i w_i^\top \varphi_{fit}(x_i, y_i) + \sum_{i,j} w_{ij}^\top \varphi_{pose}(y_i, y_j)$

[Ferrari, Marin-Jimenez, Zisserman: "Progressive Search Space Reduction for Human Pose Estimation", CVPR 2008.]

SSVM : instantiation

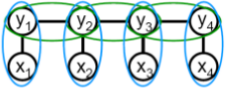
Prédiction de séquence





- Emissions (blue)
  - $f_e(x_i, y_i) = \langle w_e, \varphi_e(x_i, y_i) \rangle$
  - Can simply use the multi-class joint feature map for φ_e
- Transitions (green)
  - $f_t(y_i, y_{i+1}) = \langle w_t, \varphi_t(y_i, y_{i+1}) \rangle$
  - $\varphi_t(y_i, y_{i+1}) = \varphi_y(y_i) \otimes \varphi_y(y_{i+1})$ or $\begin{cases} [1\ 0]^T \text{ if } y_i = y_{i+1} \\ [0\ 1]^T \text{ if } y_i \neq y_{i+1} \end{cases}$

$$\begin{aligned} p(x, y) &\propto \prod_i e^{f_e(x_i, y_i)} \prod_i e^{f_t(y_i, y_{i+1})} \quad \text{for an HMM} \\ f(x, y) &= \sum_i f_e(x_i, y_i) + \sum_i f_t(y_i, y_{i+1}) \\ &= \langle w_e, \sum_i \varphi_e(x_i, y_i) \rangle + \langle w_t, \sum_i \varphi_t(y_i, y_{i+1}) \rangle \end{aligned}$$



$$\begin{aligned} w &= \begin{pmatrix} w_e \\ w_t \end{pmatrix} \\ \varphi(x, y) &= \begin{pmatrix} \sum_i \varphi_e(x_i, y_i) \\ \sum_i \varphi_t(y_i, y_{i+1}) \end{pmatrix} \\ f(x, y) &= \langle w, \varphi(x, y) \rangle \end{aligned}$$

References I

Matthew B. Blaschko and Christoph H. Lampert, *Learning to localize objects with structured output regression*, Proceedings of the 10th European Conference on Computer Vision: Part I (Berlin, Heidelberg), ECCV '08, Springer-Verlag, 2008, pp. 2–15.

T. Joachims, T. Finley, and Chun-Nam Yu, *Cutting-plane training of structural svms*, Machine Learning 77 (2009), no. 1, 27–59.

Brian Mcfee and Gert Lanckriet, *Metric learning to rank*, In Proceedings of the 27th annual International Conference on Machine Learning (ICML, 2010.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, *Large margin methods for structured and interdependent output variables*, Journal of Machine Learning Research (JMLR) 6 (2005), 1453–1484.

Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, and Yasemin Altun, *Large margin methods for structured and interdependent output variables*, J. Mach. Learn. Res. 6 (2005), 1453–1484.

Yisong Yue, Thomas Finley, Filip Radlinski, and Thorsten Joachims, *A support vector method for optimizing average precision*, Proceedings of the 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (New York, NY, USA), SIGIR '07, ACM, 2007, pp. 271–278.