

# Graph-Based Semi-Supervised Learning

Olivier Delalleau, Yoshua Bengio and Nicolas Le Roux

Université de Montréal

CIAR Workshop - April 26th, 2005

# Graph-Based Semi-Supervised Learning

Yoshua Bengio, Olivier Delalleau and Nicolas Le Roux

Université de Montréal

CIAR Workshop - April 26th, 2005

# Graph-Based Semi-Supervised Learning

Nicolas Le Roux, Olivier Delalleau and Yoshua Bengio

Université de Montréal

CIAR Workshop - April 26th, 2005

# Graph-Based Semi-Supervised Learning

Olivier Delalleau, Nicolas Le Roux and Yoshua Bengio

Université de Montréal

CIAR Workshop - April 26th, 2005

# Graph-Based Semi-Supervised Learning

Yoshua Bengio, Nicolas Le Roux and Olivier Delalleau

Université de Montréal

CIAR Workshop - April 26th, 2005

# Graph-Based Semi-Supervised Learning

Nicolas Le Roux, Yoshua Bengio and Olivier Delalleau

Université de Montréal

CIAR Workshop - April 26th, 2005

# Outline

- 1 Semi-Supervised Setting
- 2 Graph Regularization and Label Propagation
- 3 Transduction vs. Induction
- 4 Curse of Dimensionality

# Semi-Supervised Learning for Dummies

- Task of **binary classification** with labels  $y_i \in \{-1, 1\}$

# Semi-Supervised Learning for Dummies

- Task of **binary classification** with labels  $y_i \in \{-1, 1\}$
- Semi-supervised = learn something about labels using **both** labeled and unlabeled data

$$X = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$$

$$Y_l = (y_1, y_2, \dots, y_l)$$

$$n = l + u$$

# Semi-Supervised Learning for Dummies

- Task of **binary classification** with labels  $y_i \in \{-1, 1\}$
- Semi-supervised = learn something about labels using **both** labeled and unlabeled data

$$X = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$$

$$Y_l = (y_1, y_2, \dots, y_l)$$

$$n = l + u$$

- **Transduction**  $\Rightarrow \hat{Y}_u = (\hat{y}_{l+1}, \hat{y}_{l+2}, \dots, \hat{y}_n)$

# Semi-Supervised Learning for Dummies

- Task of **binary classification** with labels  $y_i \in \{-1, 1\}$
- Semi-supervised = learn something about labels using **both** labeled and unlabeled data

$$X = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n)$$

$$Y_l = (y_1, y_2, \dots, y_l)$$

$$n = l + u$$

- **Transduction**  $\Rightarrow \hat{Y}_u = (\hat{y}_{l+1}, \hat{y}_{l+2}, \dots, \hat{y}_n)$
- **Induction**  $\Rightarrow \hat{y} : \mathbf{x} \rightarrow \hat{y}(\mathbf{x})$

# The Classical Two-Moon Problem

+1



?



-1



# The Classical Two-Moon Problem



# Where are Manifolds and Kernels?

What is a good labeling  $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$ ?

# Where are Manifolds and Kernels?

What is a good labeling  $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$ ?

- 1 one that is consistent with the given labels:  $\hat{Y}_l \simeq Y_l$

# Where are Manifolds and Kernels?

What is a good labeling  $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$ ?

- ① one that is consistent with the given labels:  $\hat{Y}_l \simeq Y_l$
- ② one that is smooth on the  $\mathcal{X}$  where the data lie  
( / cluster assumption):

$$\hat{y}_i \simeq \hat{y}_j \text{ when } \mathbf{x}_i \text{ close to } \mathbf{x}_j$$

# Where are Manifolds and Kernels?

What is a good labeling  $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$ ?

- 1 one that is consistent with the given labels:  $\hat{Y}_l \simeq Y_l$
- 2 one that is smooth on the *manifold* where the data lie (*manifold* / cluster assumption):

$$\hat{y}_i \simeq \hat{y}_j \text{ when } \mathbf{x}_i \text{ close to } \mathbf{x}_j$$

# Where are Manifolds and Kernels?

What is a good labeling  $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$ ?

- ① one that is consistent with the given labels:  $\hat{Y}_l \simeq Y_l$
- ② one that is smooth on the *manifold* where the data lie (*manifold / cluster assumption*):

$$\hat{y}_i \simeq \hat{y}_j \text{ when } \mathbf{x}_i \text{ close to } \mathbf{x}_j$$

⇒ Cost function

$$C(\hat{Y}) = \sum_{i=1}^l (\hat{y}_i - y_i)^2 + \frac{\mu}{2} \sum_{i,j=1}^n \mathbf{W}_{ij} (\hat{y}_i - \hat{y}_j)^2$$

with  $\mathbf{W}_{ij} = W_X(\mathbf{x}_i, \mathbf{x}_j)$  a positive weighting function (e.g. )

# Where are Manifolds and Kernels?

What is a good labeling  $\hat{Y} = (\hat{Y}_l, \hat{Y}_u)$ ?

- 1 one that is consistent with the given labels:  $\hat{Y}_l \simeq Y_l$
- 2 one that is smooth on the *manifold* where the data lie (*manifold / cluster assumption*):

$$\hat{y}_i \simeq \hat{y}_j \text{ when } \mathbf{x}_i \text{ close to } \mathbf{x}_j$$

⇒ Cost function

$$C(\hat{Y}) = \sum_{i=1}^l (\hat{y}_i - y_i)^2 + \frac{\mu}{2} \sum_{i,j=1}^n \mathbf{W}_{ij} (\hat{y}_i - \hat{y}_j)^2$$

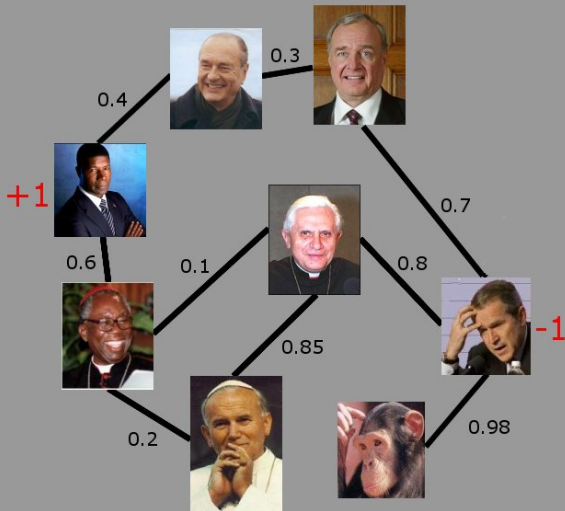
with  $\mathbf{W}_{ij} = W_X(\mathbf{x}_i, \mathbf{x}_j)$  a positive weighting function (e.g *kernel*)

# Graphs Would be Cool too!

Nodes = points, edges  $(i, j) \Leftrightarrow \mathbf{W}_{ij} > 0$ .

# Graphs Would be Cool too!

Nodes = points, edges  $(i, j) \Leftrightarrow \mathbf{W}_{ij} > 0$ .



# Regularization on Graph

Graph Laplacian:  $\mathbf{L}_{ii} = \sum_{j \neq i} \mathbf{W}_{ij}$  and  $\mathbf{L}_{ij} = -\mathbf{W}_{ij}$ .

$$\begin{aligned} C(\hat{\mathbf{Y}}) &= \sum_{i=1}^l (\hat{y}_i - y_i)^2 + \frac{\mu}{2} \sum_{i,j=1}^n \mathbf{W}_{ij} (\hat{y}_i - \hat{y}_j)^2 \\ &= \|\hat{\mathbf{Y}}_l - \mathbf{Y}_l\|^2 + \mu \hat{\mathbf{Y}}^\top \mathbf{L} \hat{\mathbf{Y}} \end{aligned}$$

# Regularization on Graph

Graph Laplacian:  $\mathbf{L}_{ii} = \sum_{j \neq i} \mathbf{W}_{ij}$  and  $\mathbf{L}_{ij} = -\mathbf{W}_{ij}$ .

$$\begin{aligned} C(\hat{Y}) &= \sum_{i=1}^l (\hat{y}_i - y_i)^2 + \frac{\mu}{2} \sum_{i,j=1}^n \mathbf{W}_{ij} (\hat{y}_i - \hat{y}_j)^2 \\ &= \|\hat{Y}_l - Y_l\|^2 + \mu \hat{Y}^\top \mathbf{L} \hat{Y} \end{aligned}$$

$C(\hat{Y})$  is minimized when

$$(\mathbf{S} + \mu \mathbf{L}) \hat{Y} = Y$$

$\Rightarrow$  linear system with  $n$  unknowns and equations.

# From Matrix Inversion to Label Propagation

Linear system rewrites for a labeled point

$$\hat{y}_i = \frac{\sum_j \mathbf{W}_{ij} \hat{y}_j + \frac{1}{\mu} y_i}{\sum_j \mathbf{W}_{ij} + \frac{1}{\mu}}$$

and for an unlabeled point

$$\hat{y}_i = \frac{\sum_j \mathbf{W}_{ij} \hat{y}_j}{\sum_j \mathbf{W}_{ij}}.$$

# From Matrix Inversion to Label Propagation

Linear system rewrites for a labeled point

$$\hat{y}_i^{(t+1)} = \frac{\sum_j \mathbf{W}_{ij} \hat{y}_j^{(t)} + \frac{1}{\mu} y_i}{\sum_j \mathbf{W}_{ij} + \frac{1}{\mu}}$$

and for an unlabeled point

$$\hat{y}_i^{(t+1)} = \frac{\sum_j \mathbf{W}_{ij} \hat{y}_j^{(t)}}{\sum_j \mathbf{W}_{ij}}.$$

**Jacobi** or **Gauss-Seidel** iteration algorithm.

# No, I didn't come up with that last week-end

- X. Zhu, Z. Ghahramani and J. Lafferty (2003): *Semi-supervised learning using Gaussian fields and harmonic functions*
- D. Zhou, O. Bousquet, T. Navin Lal, J. Weston, B. Schölkopf (2004): *Learning with local and global consistency*
- M. Belkin, I. Matveeva and P. Niyogi (2004): *Regularization and Semi-supervised Learning on Large Graphs*

## Remember your Physics' class?

**Electric network** analogy (Doyle and Snell, 1984; Zhu, Ghahramani and Lafferty, 2003).

Graph  $\Leftrightarrow$  electric network with resistors between nodes:

$$R_{ij} = \frac{1}{W_{ij}}$$

## Remember your Physics' class?

**Electric network** analogy (Doyle and Snell, 1984; Zhu, Ghahramani and Lafferty, 2003).

Graph  $\Leftrightarrow$  electric network with resistors between nodes:

$$R_{ij} = \frac{1}{W_{ij}}$$

Ohm's law (potential = label):

$$\hat{y}_j - \hat{y}_i = R_{ij} I_{ij}$$

## Remember your Physics' class?

**Electric network** analogy (Doyle and Snell, 1984; Zhu, Ghahramani and Lafferty, 2003).

Graph  $\Leftrightarrow$  electric network with resistors between nodes:

$$\mathbf{R}_{ij} = \frac{1}{\mathbf{W}_{ij}}$$

Ohm's law (potential = label):

$$\hat{y}_j - \hat{y}_i = \mathbf{R}_{ij} \mathbf{I}_{ij}$$

Kirchoff's law on an unlabeled node  $i$ :

$$\sum_j \mathbf{I}_{ij} = 0$$

## Remember your Physics' class?

**Electric network** analogy (Doyle and Snell, 1984; Zhu, Ghahramani and Lafferty, 2003).

Graph  $\Leftrightarrow$  electric network with resistors between nodes:

$$\mathbf{R}_{ij} = \frac{1}{\mathbf{W}_{ij}}$$

Ohm's law (potential = label):

$$\hat{y}_j - \hat{y}_i = \mathbf{R}_{ij} \mathbf{I}_{ij}$$

Kirchoff's law on an unlabeled node  $i$ :

$$\sum_j \mathbf{I}_{ij} = 0$$

$\Rightarrow$  Same linear system as minimizing  $C(\hat{Y})$  over  $\hat{Y}_u$  only.

# From Transduction to Induction

Solving the linear system  $\Rightarrow \hat{Y}$  (**transduction**).

# From Transduction to Induction

Solving the linear system  $\Rightarrow \hat{Y}$  (**transduction**).

From a new point  $\mathbf{x}$  and *already computed*  $\hat{Y}$ :

$$C(\hat{y}(\mathbf{x})) = C(\hat{Y}) + \frac{\mu}{2} \sum_{i=1}^n W_X(\mathbf{x}_i, \mathbf{x})(\hat{y}_i - \hat{y}(\mathbf{x}))^2$$

$$\Rightarrow \hat{y}(\mathbf{x}) = \frac{\sum_{i=1}^n W_X(\mathbf{x}_i, \mathbf{x}) \hat{y}_i}{\sum_{i=1}^n W_X(\mathbf{x}_i, \mathbf{x})}$$

(**Induction** like Parzen Windows, but using estimated labels  $\hat{Y}$ ).

# Faster Training from Subset

- Previous algorithms are at least quadratic in  $n$ .

# Faster Training from Subset

- Previous algorithms are at least quadratic in  $n$ .
- Induction formula  $\Rightarrow$  could train only on subset  $S$ .

# Faster Training from Subset

- Previous algorithms are at least quadratic in  $n$ .
- Induction formula  $\Rightarrow$  could train only on subset  $S$ .
- Better: minimize **the full cost** over  $\hat{Y}_S$  only.

# Faster Training from Subset

- Previous algorithms are at least quadratic in  $n$ .
- Induction formula  $\Rightarrow$  could train only on subset  $S$ .
- Better: minimize **the full cost** over  $\hat{Y}_S$  only.

For  $\mathbf{x}_i \in R = X \setminus S$ :

$$\hat{y}_i = \frac{\sum_{j \in S} \mathbf{W}_{ij} \hat{y}_j}{\sum_{j \in S} \mathbf{W}_{ij}}$$

i.e.  $\hat{Y}_R = \overline{\mathbf{W}}_{RS} \hat{Y}_S$ : the cost  $C(\hat{Y})$  now only depends on  $\hat{Y}_S$ .

# Nice Cost isn't it?

$$\begin{aligned} C(\hat{Y}_S) = & \underbrace{\frac{\mu}{2} \sum_{i,j \in R} \mathbf{w}_{ij} (\hat{y}_i - \hat{y}_j)^2}_{C_{RR}} \\ & + \underbrace{2 \times \frac{\mu}{2} \sum_{i \in R, j \in S} \mathbf{w}_{ij} (\hat{y}_i - \hat{y}_j)^2}_{C_{RS}} \\ & + \underbrace{\frac{\mu}{2} \sum_{i,j \in S} \mathbf{w}_{ij} (\hat{y}_i - \hat{y}_j)^2}_{C_{SS}} \\ & + \underbrace{\sum_{i \in L} (\hat{y}_i - y_i)^2}_{C_L} \end{aligned}$$

# Let's Make it Simpler

- Computing  $C_{RR}$  is quadratic in  $n \Rightarrow$  just get rid of it.
- Linear system with  $|S| = m \ll n$  unknowns  $\Rightarrow$  much faster
- Still need to do matrix multiplications  $\Rightarrow$  scales as  $O(m^2n)$
- This slide looked pretty empty with only three points
- It is important to have reading material when nobody understands your accent

# Subset Selection

① **Random:**

# Subset Selection

- 1 **Random:** fast,

# Subset Selection

- 1 **Random:** fast, easy,

# Subset Selection

- ① **Random:** fast, easy, crappy.  
Main problem = does not “fill the space” well enough  $\Rightarrow$   
bad approximation by the induction formula (some points  
have no near neighbors in the subset)

# Subset Selection

- 1 **Random:** fast, easy, crappy.  
Main problem = does not “fill the space” well enough  $\Rightarrow$  bad approximation by the induction formula (some points have no near neighbors in the subset)
- 2 **Heuristic:** greedy construction of subset by starting with the labeled points and iteratively minimizing over  $i$

$$\sum_{j \in S} w_{ij}$$

i.e. choose the point  $\mathbf{x}_i$  farthest from the current subset.  
(Additional tricks to eliminate outliers and sample more points near decision surface)

# Experimental Results

This is not last-minute research so we do have results!

# Experimental Results

This is not last-minute research so we do have results!

Table: Comparative Classification Error (Induction)

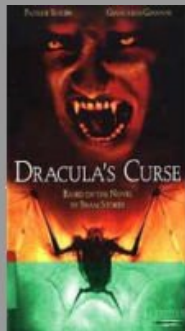
% labeled	LETTERS	MNIST	COVTYPE
<b>1%</b>			
<i>NoSub</i>	56.0	35.8	47.3
<i>RandSub<sub>subOnly</sub></i>	59.8	29.6	<b>44.8</b>
<i>RandSub</i>	57.4	27.7	75.7
<i>SmartSub</i>	<b>55.8</b>	<b>24.4</b>	45.0
<b>5%</b>			
<i>NoSub</i>	<b>27.1</b>	12.8	37.1
<i>RandSub<sub>subOnly</sub></i>	32.1	14.9	<b>35.4</b>
<i>RandSub</i>	29.1	12.6	70.6
<i>SmartSub</i>	28.5	<b>12.3</b>	35.8
<b>10%</b>			
<i>NoSub</i>	<b>18.8</b>	<b>9.5</b>	34.7
<i>RandSub<sub>subOnly</sub></i>	22.5	11.4	<b>32.4</b>
<i>RandSub</i>	20.3	9.7	64.7
<i>SmartSub</i>	19.8	<b>9.5</b>	33.4

More comparisons between *RandSub* and *SmartSub* on 8 more UCI datasets ⇒

*SmartSub* always performs better.

# Curse of Dimensionality

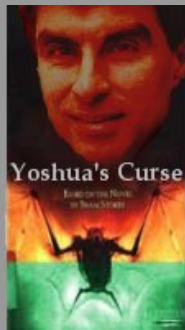
Labeling function:  $\hat{y}(\mathbf{x}) = \sum_i \hat{y}_i \overline{W}_X(\mathbf{x}_i, \mathbf{x})$



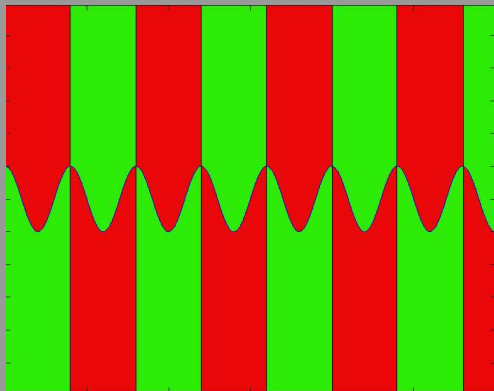
# Curse of Dimensionality

Labeling function:  $\hat{y}(\mathbf{x}) = \sum_i \hat{y}_i \overline{W}_X(\mathbf{x}_i, \mathbf{x})$

- **Locality:** far point prediction = nearest neighbor; also normal vector  $\frac{\partial \hat{y}}{\partial \mathbf{x}}(\mathbf{x})$  is approximately in the span of the nearest neighbors of  $\mathbf{x}$
- **Smoothness:** it does not vary much within a ball of radius small w.r.t.  $\sigma$  (obtained from second derivative); also form of cost  $\Rightarrow$  *the learned function varies smoothly in regions with no labeled examples*
- **Curse:** one needs lots of unlabeled examples to “fill” the region near the decision surface, and lots of labeled examples to account for all “clusters”



## “Almost Real-Life” Example



Need unlabeled examples along the *sinus* decision surface, and labeled examples in each class region.

# Conclusion

# Conclusion

- Simple non-parametric setting  $\Rightarrow$  powerful non-parametric semi-supervised algorithm

# Conclusion

- Simple non-parametric setting  $\Rightarrow$  powerful non-parametric semi-supervised algorithm
- Can scale to large datasets thanks to sparsity / subset selection

# Conclusion

- Simple non-parametric setting  $\Rightarrow$  powerful non-parametric semi-supervised algorithm
- Can scale to large datasets thanks to sparsity / subset selection
- Interesting links with electric networks / heat diffusion

# Conclusion

- Simple non-parametric setting  $\Rightarrow$  powerful non-parametric semi-supervised algorithm
- Can scale to large datasets thanks to sparsity / subset selection
- Interesting links with electric networks / heat diffusion
- Limitations of local weights: curse of dimensionality

# Conclusion

- Simple non-parametric setting  $\Rightarrow$  powerful non-parametric semi-supervised algorithm
- Can scale to large datasets thanks to sparsity / subset selection
- Interesting links with electric networks / heat diffusion
- Limitations of local weights: curse of dimensionality
- Makes it possible to be on time for lunch!!!

# References

- Belkin, M., Matveeva, I., and Niyogi, P. (2004).  
**Regularization and semi-supervised learning on large graphs.**  
In Shawe-Taylor, J. and Singer, Y., editors, *COLT'2004*. Springer.
- Delalleau, O., Bengio, Y., and Le Roux, N. (2005).  
**Efficient non-parametric function induction in semi-supervised learning.**  
In *Proceedings of the Tenth International Workshop on Artificial Intelligence and Statistics*.
- Doyle, P. G. and Snell, J. L. (1984).  
**Random walks and electric networks.**  
*Mathematical Association of America*.
- Zhou, D., Bousquet, O., Navin Lal, T., Weston, J., and Schölkopf, B. (2004).  
**Learning with local and global consistency.**  
In Thrun, S., Saul, L., and Schölkopf, B., editors, *Advances in Neural Information Processing Systems 16*,  
Cambridge, MA. MIT Press.
- Zhu, X., Ghahramani, Z., and Lafferty, J. (2003).  
**Semi-supervised learning using Gaussian fields and harmonic functions.**  
In *ICML2003*.