Transductive Learning: Motivation, Model, Algorithms

Olivier Bousquet
Centre de Mathématiques Appliquées
Ecole Polytechnique, FRANCE
olivier.bousquet@m4x.org

University of New Mexico, January 2002
Goal

- Provide motivation/potential applications
- Sketch algorithmic issues
- Sketch theoretical problems
Overview

→ Induction vs Transduction

• Algorithms

• Formalization

• Open issues
The learning problem

Induction

We consider a phenomenon $f$ that maps inputs (instances) $x$ to outputs (labels) $y = f(x)$ (here $y \in \{-1, 1\}$)

- Given a set of example pairs (training set) $\{(x_i, y_i) : i = 1, \ldots, n\}$,
- the goal is to recover $f$

→ This will allow to predict the label $y_{n+1}$ of a previously unseen instance $x_{n+1}$.

**Example:** Face recognition
Train on pictures of a person and recognize him/her the next day
But there are situations in which

- Obtaining labels is expensive
- Obtaining instances is cheap
- We know in advance the instances to be classified
- We do not care about the classification function

→ Transduction applies
Examples

Information retrieval with relevance feedback

- User enters a query
- Machine returns sample documents
- User labels the documents (relevant/non-relevant)
- Machine selects most relevant documents from database

Relevance

- Obtaining labels requires work from the user
- Obtaining documents is automatic (from database)
- Instances to be classified: documents of the database
- No need to know the classification function (changes for each query)
The learning problem

Transduction

We consider a phenomenon $f$ that maps inputs (instances) $\mathbf{x}$ to outputs (labels) $y = f(\mathbf{x})$ (here $y \in \{-1, 1\}$)

- Given a set of labeled examples $\{(\mathbf{x}_i, y_i) : i = 1, \ldots, n\}$,
- and a set of unlabeled examples $x'_1, \ldots, x'_m$
- the goal is to find the labels $y'_1, \ldots, y'_m$

→ No need to construct a function $f$, the output of the transduction algorithm is a vector of labels.

→ Transfer the information from labeled examples to unlabeled.
Using Transduction for Prediction

Given training data and data to be classified, one can either
• Use induction: build $\hat{f}$ and classify the data with it
• Use transduction directly for classifying data

Even in an inductive setting, one can use transduction.

**Example:** News filtering
• First day user classifies news according to interest
• Subsequent days, machine classifies incoming news based on first day labels
  → Train on the fly, when receiving the data to be classified
  Retrain the machine every day
  → Maximally use the information and tune the result to the news of the day
Three Learning Tasks

- Induction: \( \{(x_i, y_i) : i = 1, \ldots, n\} \mapsto f \)
- Induction with unlabeled data: \( \{(x_i, y_i) : i = 1, \ldots, n\} \cup \{x'_1, \ldots, x'_m\} \mapsto f \)
- Transduction: \( \{(x_i, y_i) : i = 1, \ldots, n\} \cup \{x'_1, \ldots, x'_m\} \mapsto (y'_1, \ldots, y'_m) \).

The choice will depend on

- Availability of unlabeled data
- Need for interpretability
- Time considerations
Overview

- Induction vs Transduction

→ Algorithms

- Formalization

- Open issues
Linear classification

Instances represented in $\mathbb{R}^d$. Find a linear separation.
Large margin classification

Margin = distance from the hyperplane to the closest point

Maximize the margin → leads to ’robust’ solution
→ Support Vector Machines
• Assumption: separated classes

• Maximize the margin on unlabeled instances.
Goal: Maximize the margin on all examples

Algorithmic issues

- no unlabeled data $\rightarrow$ quadratic optimization ($n^3$)
- unlabeled data $\rightarrow$ combinatorial problem (NP)

$\rightarrow$ Need heuristics

$\rightarrow$ Greedy optimization
Algorithms

Greedy

- Only the examples in the margin have an influence
- Label the ones with largest confidence (largest margin)

→ May add backtracking
Comments

• Influenced by starting point (induction)

• Not fully transductive because builds an $\hat{f}$

• Assumption that data is separated

→ Can we make the data separated?
Kernel Machines

Support Vector Machines

• Map data into a feature space

\[ \mathbf{x} \in \mathcal{X} \rightarrow \Phi(\mathbf{x}) \in \mathcal{F} \]

• Perform maximal margin classification in feature space

Kernel trick

• Algorithm can be implemented by computing inner products

\[ \Phi(\mathbf{x}) \cdot \Phi(\mathbf{x}') = k(\mathbf{x}, \mathbf{x}') \]

• Simply choose a kernel and run the linear algorithm on the matrix

\[ K = (k(\mathbf{x}_i, \mathbf{x}_j))_{i,j \in \{1,\ldots,n\}} \]

→ \( k \) is a measure of similarity. Algorithm works on similarity matrix.
Alignment

- Choice of Kernel = choice of feature space
- Ideal kernel = feature space contains label
- Ideal kernel matrix
  \[ k_I(x_i, x_j) = y_i y_j \]

Measure distance from ideal kernel: Alignment

\[ A(K) = \sum_{i,j} K_{ij} y_i y_j \]

Measures the data separation:

\[ A(K) = \sum_{y_i = y_j} k(x_i, x_j) - \sum_{y_i \neq y_j} k(x_i, x_j) \]
Transduction as Optimization

- Maximize alignment on the labeled data
- Corresponds to maximizing data separation
- Diagonalize, fix eigenvectors, optimize eigenvalues
Overview

- Induction vs Transduction

- Algorithms

→ Formalization

- Open issues
Formalization

- Data is fixed
  \[ x_1, \ldots, x_{n+m} \in \mathcal{X} \]
  \[ y_1, \ldots, y_{n+m} \in \{-1, 1\} \]

- Oracle (teacher) chooses randomly a subset
  \[ I \subset \{1, \ldots, n + m\} \]

- Input to algorithm
  \[ x_1, \ldots, x_{n+m} \]
  \[ I \]
  \[ (y_i)_{i \in I} \]

- Output of algorithm
  \[ (\hat{y}_i)_{i \in \{1, \ldots, n+m\}} \]
Random choice of $I$

Randomness models

- **Fixed size**
  Choose $n$ examples among $n + m$ with uniform probability for every choice, $(\frac{n+m}{n})^{-1}$. $|I| = n$.

- **Variable size**
  For each $i \in \{1, \ldots, n + m\}$ choose independently with probability $\frac{n}{n+m}$ to include it.
  $\rightarrow \mathbb{E}[|I|] = n$.

$\rightarrow$ We want to make statements that hold with high probability over the random choice of $I$. 
Recall output $\hat{y} = \hat{y}_1, \ldots, \hat{y}_{n+m}$.

$\hat{y}$ is an $n + m$ dimensional vector in $\{-1, 1\}^{n+m}$.

- Test error

  $$R(\bar{I}, y) = \frac{1}{|\bar{I}|} \sum_{i \in \bar{I}} \mathbb{I}\{\hat{y}_i \neq y_i\}$$

- Cannot be computed: need to estimate it from the data
Formalization

Error bounds

We estimate the test error by the empirical error

\[ R(I, \hat{y}) \]

We want to prove

\[ \mathbb{P}_I \left[ R(\bar{I}, \hat{y}) - R(I, \hat{y}) > \epsilon \right] \leq \delta \]

Choose a set of vectors \( \mathcal{Y} \subset \{-1, 1\}^{n+m} \). We want to bound

\[ \mathbb{P}_I \left[ \sup_{\mathcal{Y}} R(\bar{I}, y) - R(I, y) > \epsilon \right] \]
Results

When $n = m$,

$$R(\bar{I}, \mathbf{y}) \leq R(I, \mathbf{y}) + KC(\mathcal{Y}) + O\left(\frac{1}{\sqrt{n}}\right)$$

Where $C$ Rademacher complexity of $\mathcal{Y}$.

When $m > n$,

$$R(\bar{I}, \mathbf{y}) \leq R(I, \mathbf{y}) + K\bar{C}(\mathcal{Y}_{2n}) + O\left(\frac{1}{\sqrt{n}}\right)$$

where $\bar{C}(\mathcal{Y}_{2n})$ is the average Rademacher complexity computed on subsets of size $2n$ of the data.

→ Complexity can be computed from $x_i$ only. Labels don’t play any role!
Overview

- Induction vs Transduction
- Algorithms
- Formalization

→ Open issues
Comparison

Model Selection

Induction

• Define a structure without any data
• Compute empirical complexity

Transduction

• Define a structure with all the $x_i$
• Know exact complexity of this structure

→ Data-dependent classes.

→ Justifies the margin approach.
Open Problems

• Analyze alignment algorithm in that framework

• Provide model selection methods

• Provide Rademacher estimates

• Prove that unlabeled data really help

O. Bousquet: Transduction
UNM, January 2002
Conclusion

• Different framework with potentially interesting applications

• Very few people studied it: a lot remains to be done

• Challenges
  – Good empirical evidence → justification ?
  – Algorithmic → make transduction efficient
  – Theoretical → provide guarantees