Label Efficient Learning by Exploiting Multi-class Output Codes

Maria-Florina Balcan, Travis Dick, Yishay Mansour
Overview

- Active algorithms for *multi-class* learning problems.
- Basic approach:
  - Assume a *supervised* algorithm (output codes) would succeed.
  - Investigate the *implicit assumptions* of that algorithm.
  - Use them to prove guarantees for our active algorithms.

- Clustering and hyperplane-detection based algorithms

![Diagram](image.png)
Output Codes

- Natural generalization of one-vs-all learning.
- Reduction from *multi-class* to *binary* classification.
- Design *m* binary partitions of the classes.
- Think of each partition as a *semantic feature*.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
</tr>
<tr>
<td>no</td>
<td>no</td>
<td>no</td>
<td>no</td>
</tr>
<tr>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
</tr>
</tbody>
</table>
Supervised O.C. Training & Prediction

• learn a binary classifier for each semantic feature.

• Result is \( h: X \rightarrow \{\pm 1\}^m \) that predicts semantic features.

• Prediction: Assign \( x \) to class with closest code word to \( \hat{h}(x) \).
What does a linear output code look like?

\[ C = \begin{bmatrix} -1 & -1 & -1 & -1 \\ +1 & +1 & +1 & -1 \\ -1 & -1 & +1 & +1 \end{bmatrix} \]
What does a linear output code look like?

\[
K_1 \quad K_2 \quad K_3
\]

\[
\begin{align*}
X &= \begin{bmatrix}
-1 & -1 & -1 & -1 \\
+1 & +1 & +1 & -1 \\
-1 & -1 & +1 & +1 \\
-1 & -1 & +1 & +1 \\
\end{bmatrix}
\end{align*}
\]
Active Learning Setting

• Instance space $X \subset R^d$.
• Unknown target function $f^*: X \rightarrow [L]$.
• Unknown data distribution $p$ on $X$.

• Algorithm receives an iid sample $x_1, ..., x_n$ from $p$ and can query the label $y_i = f^*(x_i)$ of each point.

• Goal: output $\hat{f}: X \rightarrow [L]$ with $\Pr[\hat{f}(x) \neq f^*(x)] \leq \epsilon$ without too many queries.
**Our Main Assumption**

**Assumption:** There exists an unknown **consistent** output code classifier with linear separators. Moreover, the predicted code word $h(x)$ is always (w.p. 1) within distance $\beta$ of a class code word.

- Second part ensures the OC is not *miraculously* consistent (i.e. consistent despite making terrible predictions on the binary tasks).

- This assumption relates the OC and the unlabeled data distribution:

\[
\begin{align*}
\beta &= 0 & K_1 & K_2 & K_3 \\
\beta &= 1 & K_1 & K_2 & K_3 \\
\beta &= 2 & K_1 & K_2 & K_3
\end{align*}
\]
Summary of Results

1. If the output code is *error correcting* then we are able to learn to accuracy $\varepsilon$ with label complexity independent of $\varepsilon$ by clustering.

2. If the output code is *one-vs-all* and the data is contained in the unit ball, then we are able to learn to accuracy $\varepsilon$ using exactly $L$ label queries by clustering.

3. If the output code satisfies a novel *boundary features* condition, then we can learn to accuracy $\varepsilon$ with $L$ label queries using a hyperplane detection algorithm.
Error Correcting Output Codes

• Experts often design the code matrix to be error correcting: Large Hamming dist. between code words.
• Makes the supervised output code robust to errors in the binary classification tasks.

**Assumption:** Class code words have distance at least $2\beta + d + 1$.

For clustering:

**Assumption:** Data density $p$ has $C$-thick level sets: for all $\lambda > 0$ and $\sigma > 0$, every point of $\{p \geq \lambda\}$ is within distance $C\sigma$ of the $\sigma$-interior.
ECOC Main Observation

- For points $x_1, x_2$, the distance $d_{Ham}(h(x_1), h(x_2))$ is the number of hyperplanes crossed by the line segment from $x_1$ to $x_2$.
- If $y_1 \neq y_2$ then $d_{Ham}(h(x_1), h(x_2)) \geq 2\beta + d + 1 - 2\beta = d + 1$.
- If hyperplanes are in general position, this implies $|x_1 - x_2| > 0$.
- So there is a non-zero margin $g > 0$ between all classes!
Clustering Algorithm for ECOC Setting

1. Draw an unlabeled sample of data.
2. Connect points closer than distance $r$.
3. Query the label from each cluster in decreasing order of size until at most an $\epsilon/4$-fraction of data is in unlabeled clusters.
4. Output a nearest neighbor classifier using the labeled clusters.

Let $N$ be the number of connected components of $\{p \geq \tilde{\epsilon}\}$ for $\tilde{\epsilon} \approx \epsilon$.

**Theorem:** If $r \leq g$ and $n = O\left(\frac{1}{\epsilon^2} \left(\frac{Cd}{r}\right)^{2d} + N\right)$ then with probability at least $1 - \delta$ the above algorithm will query at most $N$ labels and achieve error $\leq \epsilon$.

Label complexity is essentially independent of target error rate $\epsilon$!
What about weaker requirements on the Hamming distance between code words?

1. One-vs-all on the unit ball: Hamming dist. = 2

2. Boundary feature condition: Hamming dist. = 1
   • This means different classes can be very well connected and so clustering will fail!
Assumption: The data is in the unit ball and there exists a consistent one-vs-all classifier.

i.e., there are linear separators $h_1, \ldots, h_L$ such that $x \in B$ belongs to class $i$ if and only if $h_i(x) > 0$.

Assumption: $\beta = 0$ and $c_{lb} \leq p(x) \leq c_{ub}$ for $x$ with $d_{Ham}(h(x), C) \leq \beta$

Idea: After projecting to the surface of the ball, the classes are probabilistically separated! Find high-density clusters after projecting to the unit sphere.

Theorem: For any $\epsilon > 0$, running our alg. on unlabeled sample of size $n = \tilde{O}\left(\frac{c_{ub}^d d^d}{\epsilon^2 d c_{lb}^2 d b_{min}^2}\right)$ will query $L$ labels and have error at most $\epsilon$ w.h.p.
**Assumption:** For every semantic feature $j$, there exists a class $i$ such that flipping feature $i$ for class $j$ produces a code word not equal to any other class.

**Assumption:** $\beta = 0$ and $c_{lb} \leq p(x) \leq c_{ub}$ for $x$ with $d_{Ham}(h(x), C) \leq \beta$

- This implies that every linear separator is a linear boundary on the support of $p$.
- So we can recover the linear separators by estimating linear boundaries of the support!

**Theorem:** For any $\epsilon > 0$, running our alg. on an unlabeled sample of size $n = \tilde{O}\left(\frac{m^2 c_{ub}^2}{\epsilon^4 R^d}\right)$ will query at most $L$ labels and will have error at most $\epsilon$ w.h.p.

* $R$ is a scale parameter of the problem
Summary & Future Work

- Designed and analyzed active algorithms for multi-class problems.
- Analysis leveraged the implicit assumptions of supervised output codes.

![Diagram]

- Future Work:
  - Algorithms with non-exponential unlabeled sample complexity.
  - Similar analysis using implicit assumptions of other supervised algorithms.

Thanks!