Sparse Fuzzy Techniques Improve Machine Learning

Reinaldo Sanchez\textsuperscript{1}, Christian Servin\textsuperscript{1,2}, and Miguel Argaez\textsuperscript{1}

\textsuperscript{1}Computational Science Program
University of Texas at El Paso
500 W. University
El Paso, TX 79968, USA
reinaldosanar@gmail.com, christians@utep.edu
margaez@utep.edu

\textsuperscript{2}Information Technology Department
El Paso Community College, El Paso, Texas, USA
1. Machine Learning: A Typical Problem

- In machine learning:
  - we know how to classify several known objects, and
  - we want to learn how to classify new objects.

- For example, in a biomedical application:
  - we have microarray data corresponding to healthy cells and
  - we have microarray data corresponding to different types of tumors.

- Based on these samples, we would like to be able, given a microarray data, to decide
  - whether we are dealing with a healthy tissue or with a tumor, and
  - if it is a tumor, what type of cancer does the patient have.

- Each object is characterized by the results \( x = (x_1, \ldots, x_n) \) of measuring several \( (n) \) different quantities.

- So, in mathematical terms, machine learning can be described as a following problem:
  
  - we have \( K \) possible labels 1, \ldots, \( K \) describing different classes;
  - we have several vectors \( x(j) \in \mathbb{R}^n, j = 1, \ldots, N; \)
  - each vector is labeled by an integer \( k(j) \) ranging from 1 to \( K; \)
  - vectors labeled as belonging to the \( k \)-th class will be also denoted by \( x(k, 1), \ldots, x(k, N_k); \)
  - we want to use these vectors to assign, to each new vector \( x \in \mathbb{R}^n, \) a value \( k \in \{1, \ldots, K\} \).

- Often, each class $C_k$ is *convex*: if $x, x' \in C_k$ and $\alpha \in (0, 1)$, then $\alpha \cdot x + (1 - \alpha) \cdot x' \in C_k$.

- It all $C_k$ are convex, then we can separate them by using linear separators.

- For example, for $K = 2$, there exists a linear function $f(x) = c_0 + \sum_{i=1}^{n} c_i \cdot x_i$ and a threshold value $y_0$ such that:
  - for all vectors $x \in C_1$, we have $f(x) < y_0$, while
  - for all vectors $x \in C_2$, we have $f(x) > y_0$.

- This can be used to assign a new vector $x$ to an appropriate class: $x \rightarrow C_1$ if $f(x) < y_0$, else $x \rightarrow C_2$.

- For $K > 2$, we can use linear functions separating different pairs of classes.

- In practice, the classes $C_k$ are often not convex.
- As a result, we need nonlinear separating functions.
- The first such separating functions came from simulating (non-linear) biological neurons.
- Even more efficient algorithms originate from the Taylor representation of a separating function:

$$f(x_1, \ldots, x_n) = c_0 + \sum_{i=1}^{n} c_i \cdot x_i + \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} \cdot x_i \cdot x_j + \ldots$$

- This expression becomes linear if we add new variables $x_i \cdot x_j$, etc., to the original variables $x_1, \ldots, x_n$.
- The corresponding Support Vector Machine (SVM) techniques are the most efficient in machine learning.
- For example, SVM is used to automatically diagnose cancer based on the microarray gene expression data.
5. There Is Room for Improvement

- In SVM, we divide the original samples into a training set and a training set.
- We train an SVM method on the training set.
- We test the resulting classification on a testing set.
- Depending on the type of tumor, 90 to 100% correct classifications.
- 90% is impressive, but it still means that up to 10% of all the patients are misclassified.
- How can we improve this classification?
6. Our Idea

- Efficient linear algorithms are based on an assumption that all the classes $C_k$ are convex.
- In practice, the classes $C_k$ are often not convex.
- SVM uses (less efficient) general nonlinear techniques.
- Often, while the classes $C_k$ are not exactly convex, they are somewhat convex:
  - for many vectors $x$ and $x'$ from each class $C_k$ and for many values $\alpha$,
  - the convex combination $\alpha \cdot x + (1-\alpha) \cdot x'$ still belongs to $C_k$.
- In this talk, we use fuzzy techniques to formalize this imprecise idea of “somewhat” convexity.
- We show that the resulting machine learning algorithm indeed improves the efficiency.
7. Need to Use Degrees

- “Somewhat” convexity means that if \( x, x' \in C_k \), then \( \alpha \cdot x + (1 - \alpha) \cdot x' \in C_k \) with some degree of confidence.
- Let \( \mu_k(x) \) denote our degree of confidence that \( x \in C_k \).
- We arrive at the following fuzzy rule: If \( x, x' \in C_k \) and convexity holds, then \( \alpha \cdot x + (1 - \alpha) \cdot x' \in C_k \).
- If we use product for “and”, we get
  \[
  \mu_k(\alpha \cdot x + (1 - \alpha) \cdot x') \geq r \cdot \mu_k(x) \cdot \mu_k(x').
  \]
- So, if \( x'' \) is a convex combination of two sample vectors, then \( \mu_k(x'') \geq r \cdot 1 \cdot 1 = r \).
- For combination of three sample vectors, \( \mu_k(x'') \geq r^2 \).
- For \( y = \sum_{j=1}^{N_k} \alpha_j \cdot x(k, j) \), we have \( \mu_k(y) \geq r^{||\alpha||_0 - 1} \), where \( ||\alpha||_0 \) is the number of non-zero values \( \alpha_j \).
8. Using Closeness

- If $y \in C_k$ and $x$ is close to $y$, then $x \in C_k$ with some degree of confidence.

- In probability theory, Central Limit Theorem leads to Gaussian degree of confidence.

- We thus assume that the degree of confidence is described by a Gaussian expression $\exp \left( -\frac{\|x - y\|^2}{2\sigma^2} \right)$.

- As a result, for every two vectors $x$ and $y$, we have
  \[
  \mu_k(x) \geq \mu_k(y) \cdot \exp \left( -\frac{\|x - y\|^2}{2\sigma^2} \right).
  \]
9. Combing Both Formulas

- Resulting formula: \( \mu_k(x) \geq \tilde{\mu}_k(x) \), where:

\[
\tilde{\mu}_k(x) \overset{\text{def}}{=} \max_{\alpha} \exp \left( - \frac{\| x - \sum_{j=1}^{N_k} \alpha_j \cdot x(k, j) \|^2}{2\sigma^2} \cdot \frac{r\|\alpha\|_0 - 1}{r} \right).
\]

- To classify a vector \( x \), we:
  - compute \( \tilde{\mu}_k(x) \) for different classes \( k \), and
  - select the class \( k \) for which \( \tilde{\mu}_k(x) \) is the largest.

- This is equivalent to minimizing \( L_k(x) = -\ln(\tilde{\mu}_k(x)) \):

\[
L_k(x) = C \cdot \left\| x - \sum_{j=1}^{N_k} \alpha_j \cdot x(k, j) \right\|^2_2 + \|\alpha\|_0.
\]
10. Towards an Efficient Algorithm

- **Reminder:** we minimize \( C \cdot \left\| x - \sum_{j=1}^{N_k} \alpha_j \cdot x(k,j) \right\|_2^2 + \| \alpha \|_0. \)

- **Lagrange multipliers:** this is equiv. to minimizing \( \| \alpha \|_0 \) under the constraint \( \left\| x - \sum_{j=1}^{N_k} \alpha_j \cdot x(k,j) \right\|_2^2 \leq C. \)

- **Problem:** minimizing \( \| \alpha \|_0 \) is, in general, NP-hard.

- **Good news:** often, minimizing \( \| \alpha \|_0 \) is equivalent to minimizing \( \| \alpha \|_1 \overset{\text{def}}{=} \sum_{j=1}^{N_k} |\alpha_j|. \)

- **Resulting algorithm:** minimize

\[
C' \cdot \left\| x - \sum_{j=1}^{N_k} \alpha_j \cdot x(k,j) \right\|_2^2 + \| \alpha \|_1.
\]
11. Taking the Specific Problem into Account

• For microarray analysis, the actual values of the vector $x$ depend on the efficiency of the microarray technique.

• In other words, with a less efficient technique, we will get $\lambda \cdot x$ for some constant $\lambda$.

• From this viewpoint, it is reasonable to use:
  
  - not just convex combinations, but also
  
  - arbitrary linear combinations of the original vectors $x(k, j)$. 
12. Towards an Efficient Algorithm (cont-d)

• We repeat $\ell_1$-minimization for each of $K$ classes.
• While $\ell_1$-minimization is efficient, it still takes a large amount of computation time; so:
  – instead of trying to represent the vector $x$ as a linear combination of vectors from each class,
  – let us look for a representation of $x$ as a linear combination of all sample vectors, from all classes:

$$
C' \cdot \left\| x - \sum_{j=1}^{N} \alpha_j \cdot x(j) \right\|^2_2 + \| \alpha \|_1 \rightarrow \min.
$$

• Then, for each class $k$, we only take the components belonging to this class, and select $k$ for which

$$
\left\| x - \sum_{j:k(j)=k} \alpha_j \cdot x(j) \right\|^2_2 \rightarrow \min.
$$
13. Interesting Observation

- This time-saving idea not only increased the efficiency, it also improve the quality of classification.
- We think that this improvement is related to the fact that all the data contain measurement noise.
- On each computation step, we process noisy data.
- Hence, the results get noisier and noisier with each computation step.
- From this viewpoint, the longer computations, the more noise we add.
- By speeding up computation, we thus decrease the noise.
- This compensates a minor loss of optimality, when we replacing $K$ minimizations with a single one.
14. Results

- The probability $p$ of correct identification increased:
  - for brain tumor, $p$ increased from 90% for the best SVM techniques to 91% for our method;
  - for prostate tumor, the probability $p$ similarly increased from 93% to 94%.

- Our method has an additional advantage:
  - to make SVM efficient, we need to select appropriate nonlinear functions;
  - if we select arbitrary functions, we usually get not-so-good results;
  - in contrast, our sparse method has only one parameter to tune: the parameter $C'$.

- Our technique is this less subjective, more reliable – and leads to better (or similar) classification results.
15. A Paper with Detailed Description of Results