ON EFFICIENT LEARNING AND CLASSIFICATION KERNEL METHODS

S.Y. Kung and Pei-yuan Wu

Princeton University

ABSTRACT

Improving learning and classification efficiency has become increasingly important for machine learning. If the traditional RBF kernel is adopted, the learned kernel-based classifier usually delivers better performance by engaging a large training dataset. However, such a high performance comes at the expense of costly learning and classification complexities, which grow drastically with the training size N. To overcome this curse of dimensionality, we propose a so-called TRBF kernel(with finite intrinsic degree J) which approximates the RBF kernel. The contributions of this paper are as follows. First, the optimal classification efficiency attainable is shown to be $J' \approx J$. To improve learning efficiency, we propose a fast PDA algorithm with learning complexity linearly growing with N. We adopt pruned-PDA (PPDA) to improve the accuracy by removing harmful "anti-support" vectors from the training set. Experiments on ECG dataset showed that TRBF-PPDA delivers nearly optimal performance with very low power.

***Index Terms*— SVM, PDA, PPDA, anti-support vectors, intrinsic degree of kernels, learning efficiency, classification efficiency, low-power on-line ECG detection**

1. INTRODUCTION

An overall machine learning system includes two phases, learning phase and prediction phase, as depicted in Figure 1. In most literatures in kernel-based machine learning, performance or prediction accuracy has always been the primary concern. This paper, however, places a main focus on the computational efficiency from two aspects: (1) learning efficiency and (2) on-line classification.

The Gaussian RBF kernel, being one of the most popular and effective kernels adopted, suffers from the so-called the curse of dimensionality that its learning and classification complexities grow drastically with the training size N.

This renders the RBF kernel unsuitable when computing cost is a concern, thus a substitute kernel is needed to provide a proper balance between the performance and complexity. We shall demonstrate that the complexities are closely dependent on (1) the kernel selection, (2) the choice of classifiers, and (3) the size of training dataset. Moreover, the very same factors will also play a pivotal role in the prediction accuracy.

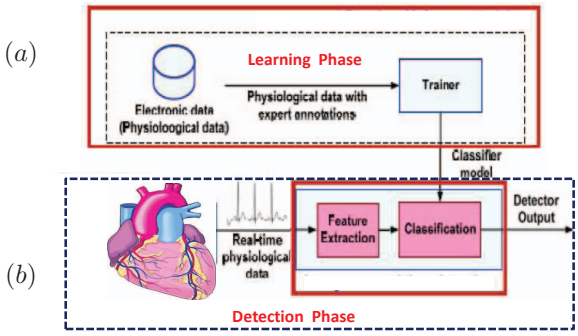

Fig. 1. (a) In the learning phase, the training data (with known class labels) are used to train a desired classifier. (b) In the prediction/detection phase, the class of a test signal is predicted by the trained classifier.

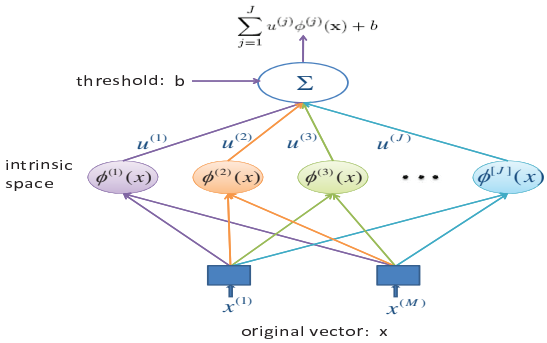

Fig. 2. This two-layer network shows how any vector $\mathbf{x}$ can be mapped to a new representative vector in its intrinsic space.

Given a finite-decomposable kernel:

$$K(\mathbf{x}, \mathbf{x}') = \sum_{j=1}^{J} \phi^{(j)}(\mathbf{x})\phi^{(j)}(\mathbf{x}'),$$

where the intrinsic degree J represents the number of independent basis functions in the associated intrinsic space, which offers a new representation layer exemplified by Figure 2. For example, a p-th order polynomial kernel (abbreviated as POLY_p) is $K(\mathbf{x}, \mathbf{x}') = \left(1 + \frac{\mathbf{x}\cdot\mathbf{x}'}{\sigma^2}\right)^p$. Denote $x^{(M+1)} = 1$, barring a scaling factor, each basis function has a appearance as follows:

$$\left(x^{(1)}\right)^{d_1} \cdots \left(x^{(M)}\right)^{d_M} \left(x^{(M+1)}\right)^{d_{M+1}}, \text{with} \sum_{m=1}^{M+1} d_m = p.$$

There are $J = J^{(p)} = \frac{(M+p)!}{M!\ p!}$ different such combinations.

It is our finding that the intrinsic degree dictates the learning/classification complexities and prediction performance. In other words, a kernel can be quantitatively and qualitatively characterized by its intrinsic degree J.

2. IMPROVING CLASSIFICATION COMPLEXITY OF KERNEL-BASED CLASSIFIERS

After the classifier's parameters are learned, the class of a test signal will be determined by the trained classifier. The incurring classification complexity dictates the on-line processing power, so it may become a critical concern for green IT applications. To be thorough, we shall explore all three candidate classification schemes before concluding that the best scheme will require $J' \approx J$ operations.

2.1. Classification Complexity for RBF Kernels

For RBF-kernel classifiers, the decision function is

$$f(\mathbf{x}) = \sum_{n=1}^{N} a_n K(\mathbf{x}, \mathbf{x}_i) + b = \mathbf{a}^T \overrightarrow{\mathbf{k}}(\mathbf{x}) + b. \qquad (1)$$

To compute the squared-distance $||\mathbf{x}||^2 + ||\mathbf{x}_i||^2 - 2\mathbf{x}^T\mathbf{x}_i$ in the RBF function $K(\mathbf{x}, \mathbf{x}_i)$, it requires roughly M operations, each operation involves one MAC (multiplication-and-addition). The more training data the higher the model complexity. More exactly, the complexity is NM.

2.2. Inner-Product in Intrinsic Space

Note that the decision function may also be computed from the inner-product in the intrinsic space because

$$f(\mathbf{x}) = \sum_{i=1}^{N} a_i \overrightarrow{\phi}(\mathbf{x}_i)^T \overrightarrow{\phi}(\mathbf{x}) \ + b = \mathbf{u}^T \overrightarrow{\phi}(\mathbf{x}) \ + b$$

Given a test pattern $\mathbf{x}$, it (1) requires a minimum of $J^{(p)}$ operations to produce all the elements of $\overrightarrow{\phi}(\mathbf{x})$; and then (2) costs $J^{(p)}$ operations to compute $\mathbf{u}^T \overrightarrow{\phi}(\mathbf{x})$. The total classification complexity amounts to $2 \times J^{(p)}$, i.e. it is independent of N.

2.3. Consecutive Tensor Operations

Let us now introduce a new method to further save almost half of the above classification complexity. As an example, we treat the POLY_3 case in full detail:

$$\begin{aligned} f(\mathbf{x}) &= \sum_{n=1}^{N} a_n K(\mathbf{x}, \mathbf{x}_n) + b = \sum_{n=1}^{N} a_n \left(\tilde{\mathbf{x}}^T \tilde{\mathbf{x}}_n\right)^3 + b \\ &= \sum_{i=1}^{M+1} \sum_{j=1}^{M+1} \sum_{k=1}^{M+1} \tilde{w}_{ijk} \tilde{x}^{(i)} \tilde{x}^{(j)} \tilde{x}^{(k)} + b, \end{aligned}$$

where $\tilde{w}_{ijk} \equiv \sum_{n=1}^{N} a_n \tilde{x}_n^{(i)} \tilde{x}_n^{(j)} \tilde{x}_n^{(k)}$, with $\tilde{x}_n^{(i)}, i = 1, \cdots, M+1$ being the variables in the expanded vector: $\tilde{\mathbf{x}} \equiv \left[\sigma^{-1}\mathbf{x}^T \ \ 1\right]^T$.

For best computational efficiency, the order of computations had better be rearranged as follows:

$$f(\mathbf{x}) = \sum_{i=1}^{M+1} \tilde{x}^{(i)} \left[\sum_{j=1}^{M+1} \tilde{x}^{(j)} \left(\sum_{k=1}^{M+1} \tilde{w}_{ijk} \tilde{x}^{(k)} \right) \right] + b.$$

By exploiting the (3-way) symmetry of the tensor represented by $\tilde{w}_{ijk}$, we have

$$f(\mathbf{x}) = \sum_{i=1}^{M+1} \tilde{x}^{(i)} \left[\sum_{j=1}^{i} \tilde{x}^{(j)} \left(\sum_{k=1}^{j} u_{ijk} \tilde{x}^{(k)} \right) \right] + b,$$

where $u_{ijk} = \gamma_{ijk} \tilde{w}_{ijk}$ and $\{\gamma_{ijk}\}$ denote the multinomial coefficients. [1] Therefore, the classification complexity amounts to $J' = J^{(3)} + J^{(2)} + J^{(1)} + 1$.

Classification Complexity. By induction, the classification complexity for a POLY_p kernel is

$$J' = \sum_{q=1}^{p} J^{(q)} \ +1 = \sum_{q=1}^{p} \left(\begin{array}{c} M+q \\ q \end{array} \right) + 1 = \left(\begin{array}{c} M+p+1 \\ p \end{array} \right) .$$

With a complexity $J' \approx J$, it is clearly the most cost-effective choice when the training size N is large.

3. FINITE-J-DEGREE APPROXIMATION OF RBF

We now face a dilemma that, on one hand, the RBF kernel can deliver the best performance but it has an indefinite intrinsic degree; on the other hand, polynomial kernels may compromise the performance but it offers cost-effective implementation due to their finite intrinsic degree.

Fortunately, polynomial kernels are not the only kernels that have a finite intrinsic degree. A simple and intuitive way to combine the best of (the performance of) RBF and (the finite degree of) POLY kernels is by the following truncated-RBF (TRBF_p) kernel:

$$K(\mathbf{x}, \mathbf{y}) = \exp\left\{-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right\} \left[\sum_{k=1}^{p} \frac{1}{k!} \left(\frac{\mathbf{x} \cdot \mathbf{y}}{\sigma^2}\right)^k\right] \exp\left\{-\frac{\|\mathbf{y}\|^2}{2\sigma^2}\right\},$$

with each basis function, barring a factor, having the form:

$$\exp\left\{-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right\} \left(x^{(1)}\right)^{d_1} \cdots \left(x^{(M)}\right)^{d_M} \left(x^{(M+1)}\right)^{d_{M+1}},$$

which strictly speaking is non-polynomial. Nevertheless, the intrinsic degree of TRBF_p remains to be

$$J = J^{(p)} = \left(\begin{array}{c} M+p \\ p \end{array} \right) = \frac{(M+p)!}{M!p!}. \qquad (2)$$

Moreover, the classification complexity for TRBF_p is $J' \approx J = J^{(p)}$, i.e. it is exactly the same as POLY_p kernel.

Note that TRBF is simply a finite-order Taylor-expansion, ~~for a more sophisticated~~ RBF approximation, see [1].

[1] Note that u_{ijk}'s, being exactly the same as the coefficients of the intrinsic decision vector $\mathbf{u}$, can be obtained via Eq. 4.

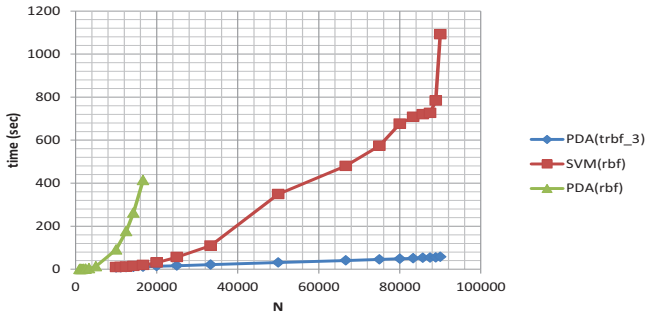

Fig. 3. The computer cycles recorded with the MatLab classification codes on Intel-Core-I5-2410M Microprocessor (2.3 GHz), when training the ECG dataset. It shows that direct PDA requires $O(N^3)$ and SVM $O(N^2)$ operations. while the fast-PDA curve grows linearly with N, more exactly J^2N.

4. FAST LEARNING KERNEL METHODS

All kernel-based classifiers work with a kernel matrix which is tightly linked to the intrinsic data matrix $\mathbf{\Phi}$:

$$\mathbf{K} = \mathbf{\Phi}^T\mathbf{\Phi}, \text{ where } \mathbf{\Phi} = \left[\begin{array}{cccc} \overrightarrow{\phi}(\mathbf{x}_1) & \overrightarrow{\phi}(\mathbf{x}_2) & \cdots & \overrightarrow{\phi}(\mathbf{x}_N) \end{array} \right].$$

4.1. Learning Complexities for PDA and SVM

For PDA and SVM classifiers, the learning complexities grow with N at very different rates.

- **Learning Complexity of PDA.** PDA is a kernel-based variant of LDA [2], with its decision function given as $f(\mathbf{x}) = \mathbf{a}^T\overrightarrow{\mathbf{k}}(\mathbf{x}) + b$, where the decision vector $\mathbf{a}$ can be learned by the following formula:

$$\mathbf{a} = [\mathbf{K}+\rho\mathbf{I}]^{-1}(\mathbf{y}-b\mathbf{e}), \text{ where } b = \frac{\mathbf{y}^T[\mathbf{K}+\rho\mathbf{I}]^{-1}\mathbf{e}}{\mathbf{e}^T[\mathbf{K}+\rho\mathbf{I}]^{-1}\mathbf{e}}. \quad (3)$$

  where $\mathbf{y} \equiv [y_1 \cdots y_N]^T$, $\mathbf{e} \equiv [1 \cdots 1]^T$, and ρ denotes the perturbation variance. The direct PDA solution involves the inversion of the $N \times N$ matrix $(\mathbf{K} + \rho\mathbf{I})$ and therefore incurs a high complexity of $O(N^3)$.

- **Learning Complexity of SVM.** The SVM learning involves a quadratic programming problem. By invoking the SMO scheme, the SVM learning cost is reportedly to grow at a modest rate near $O(N^2)$. [3]

For a huge N, neither PDA's $O(N^3)$ nor SVM's $O(N^2)$ is affordable. We must design a numerically more efficient method whose learning complexity grows linearly with N.

4.2. Fast Algorithm for PDA

The scatter matrix in the intrinsic space is $\mathbf{S} = \mathbf{\Phi}\mathbf{\Phi}^T$. Since

$$\begin{aligned} [\mathbf{K}+\rho\mathbf{I}]^{-1} &= [\mathbf{\Phi}^T\mathbf{\Phi}+\rho\mathbf{I}]^{-1} \\ &= \rho^{-1}\mathbf{I} - \rho^{-1}\mathbf{\Phi}^T\left[\rho\mathbf{I}+\mathbf{S}\right]^{-1}\mathbf{\Phi}, \end{aligned}$$

therefore,

$$\begin{aligned} \mathbf{\Phi}[\mathbf{K}+\rho\mathbf{I}]^{-1} &= \rho^{-1}\mathbf{\Phi} - \rho^{-1}\mathbf{\Phi}\mathbf{\Phi}^T\left[\rho\mathbf{I}+\mathbf{S}\right]^{-1}\mathbf{\Phi} \\ &= \rho^{-1}\left(\mathbf{I} - \mathbf{S}\left[\rho\mathbf{I}+\mathbf{S}\right]^{-1}\right)\mathbf{\Phi} = \left[\mathbf{S}+\rho\mathbf{I}\right]^{-1}\mathbf{\Phi}, \end{aligned}$$

and the decision vector $\mathbf{u}$ may be derived as:

$$\begin{aligned} \mathbf{u} = \mathbf{\Phi}\mathbf{a} &= \mathbf{\Phi}[\mathbf{K}+\rho\mathbf{I}]^{-1}[\mathbf{y}-b\mathbf{e}] \\ &= \left[\mathbf{S}+\rho\mathbf{I}\right]^{-1}\mathbf{\Phi}[\mathbf{y}-b\mathbf{e}], \end{aligned} \quad (4)$$

where $b = \frac{\mathbf{y}^T\mathbf{e}-(\mathbf{\Phi}\mathbf{y})^T(\mathbf{S}+\rho\mathbf{I})^{-1}(\mathbf{\Phi}\mathbf{e})}{\mathbf{e}^T\mathbf{e}-(\mathbf{\Phi}\mathbf{e})^T(\mathbf{S}+\rho\mathbf{I})^{-1}(\mathbf{\Phi}\mathbf{e})}$. The decision function is

$$f(\mathbf{x}) = \mathbf{u}^T\overrightarrow{\phi}(\mathbf{x}) \ + b\,. \quad (5)$$

Computationally, this fast-PDA algorithm incurs three main costs: (1) the computation of the $J \times J$ scatter matrix $\mathbf{S}$ requires J^2N operations; (2) the inversion of the $J \times J$ matrix $\mathbf{S} + \rho\mathbf{I}$ requires roughly J^3 operations; and (3) the matrix-vector multiplications require only a negligible order of NJ operations. (For simplicity, the exact scaling factors are omitted here.) In summary, the learning complexity is $Min\left(\ N^3, J^3 + J^2N\ \right)$. When $N \gg J$, the complexity becomes simply J^2N, which represents a drastic saving.

Simulation for Verification of Learning Efficiency. The simulation study was based on the MIT-BIH ECG dataset for arrhythmia detection. [4] The full database contains a total of 112,803 heart beats, and $M = 21$ morphology features are extracted to represent each beat. The computational costs of SVM, PDA, and fast-PDA are depicted in Figure 3. [2]The computer cycles recorded by our simulation provide the empirical evidence to support our theoretical comparative analysis. It shows that when N is huge, say $N \geq 80K$, the learning complexity becomes very costly for both direct-PDA and SVM. In contrast, the fast PDA offers a much higher learning efficiency to cope with an even larger N which will in turn lead to a higher prediction accuracy, cf. Figures 4 and 5.

5. PRUNING OF "ANTI-SUPPORT" VECTORS

The key to the success of SVM learning lies in its identification of a set of "support vectors" which exclusively determine the decision boundary. Motivated by such SVM learning principle, it is natural to postulate that PDA's performance may also be enhanced if we are more selective with the admissible training vectors. This is the principle behind the P-PDA.

For PDA and SVM, each vector $\mathbf{x}_i$ is associated with an error $\xi_i \equiv y_i - f(\mathbf{x}_i)$. [2] According to Eq. 3, the *error vector* is

$$\overrightarrow{\xi} = [\mathbf{y}-b\mathbf{e}] - \mathbf{K}\mathbf{a} = \rho\mathbf{a}.$$

The anti-support vectors $\mathbf{x}_i$ are those with error ξ_i exceeding a certain threshold. These are considered harmful because they

[2]The training time for TRBF3-SVM (not shown) is almost the same as RBF-SVM.

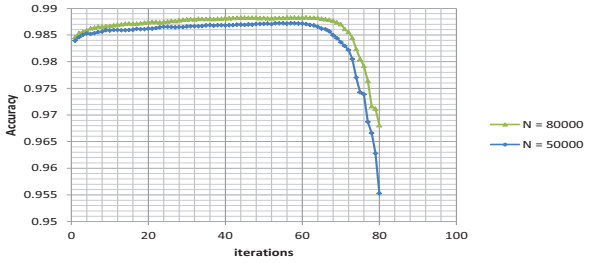

Fig. 4. Experimental results of consecutive P-PDAs show monotonic improving performance with pruning iterations.

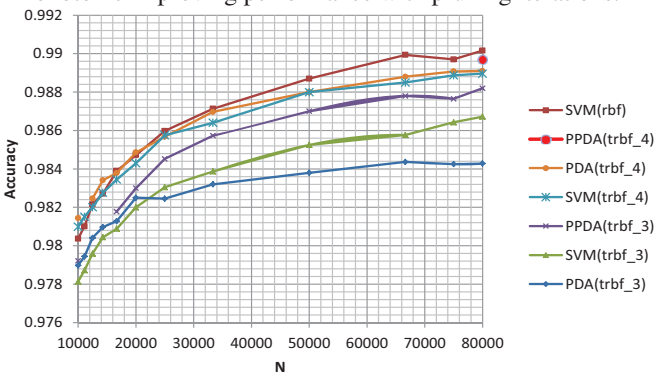

Fig. 5. Experimental results of arrhythmia detection. Accuracy for different classifiers: RBF-SVM, TRBF-SVM, TRBF-PDA, and TRBF-PPDA. The following parameters were adopted $\sigma = 2.5$ for both RBF and TRBF_3 kernels, $C = 10$ for SVM and $\rho = 0.1$ for both PDA and P-PDA.

assume larger weights and more strongly distort the learned decision boundary. Thus it is imperative to exclude these vectors from participating in the decision rule. The remaining probational dataset can be used to learn a new decision boundary in the next iteration. This process will be repeated over many iterations.

Intuitively, the performance should improve since the new result will be free from the undue influence of the harmful vectors. Figure 4 demonstrates such improvement during the first 40-60 iterations of the pruning process (4% pruned per iteration), before the performance saturates and then eventually deteriorates when too few training vectors remain.

6. COMPARISON OF SVM, PDA, AND P-PDA

Based on the MIT-BIH ECG dataset, various combinations of classifiers and kernels are tested and their prediction accuracies, training times, and power budgets are evaluated.

Comparison of Performance. The prediction performances of SVM, PDA, and P-PDA are depicted in Figure 5.

- *Performances of Different Kernels.* It is no surprise that RBF-SVM has the highest performance and that SVM and PDA using TRBF_4 kernel outperform their counterparts using TRBF_3 kernel, i.e. TRBF_4 is a closer approximation to RBF than TRBF_3.
- *Performances Improve with N.* As confirmed by both Figures 4 and 5, the accuracies for all the classifier-kernel combinations monotonically increase with N.
- *Performances of Different Classifiers.* Note also that, for TRBF_4 kernel, SVM and PDA have about the same performance. For the TRBF_3 kernel, SVM outperforms PDA when training sizes increase. However, enforced by training vector pruning, P-PDA reclaims a slight performance advantage over SVM.
- *High-Performance and Low-Power Classifiers.* Throughout the experiments, the highest accuracy benchmark 99% is set by RBF-SVM with $N \geq 80K$. It is curious to compare RBF-SVM and RBF-PDA and help assess the performance-complexity tradeoff. Unfortunately, learning of RBF-PDA for $N \geq 80K$ is computationally infeasible. As a consolation, TRBF offers an approximate comparison. In fact, the accuracy of TRBF-PDA is already as high as 98.8%. The $.2\%$ gap may be attributed to truncation approximation incurred by TRBF. It may also be caused by the fact that SVM uses a more selective subset of (supporting) training vectors. To explore this possibility, we decided to train a TRBF4-PPDA, with almost one-day's computing, and obtain again a 99% accuracy – more exactly 98.97% – shown as the big red circle on the rightmost side of Figure 5.

Classification Efficiency of TRBF-PPDA. According to [5], the power budgeted for ECG applications is typically limited to 1-10mW for wearable devices or 10-100 μW for implantable devices. Based on the budget, while it costs only 1.56 mJ per ECG feature extraction, the energy cost per classification would be 49.52 mJ via the conventional RBF-SVM. [5] To meet the energy budget, such a high cost must be cut by 10 to 40 folds. This translates to an intrinsic degree of $J = 2000$ to 8000, corresponding to TRBF_3 and TRBF_4 respectively. This suggests that TRBF_3 can operate as a green classifier way under the allocated power budget.

7. REFERENCES

[1] A. Vedaldi and A. Zisserman, "Efficient additive kernels via explicit feature maps," *IEEE Trans. on PAMI*, 2011.

[2] S. Y. Kung, "Kernel approaches to unsupervised and supervised machine learning," Proc. PCM2009, pp. 1–32.

[3] J. C. Platt, "Using analytic QP and sparseness to speed training of support vector machines," *Advances in Neural Information Processing Systems*, 1998.

[4] PhysioNet, "http://www.physionet.org," .

[5] K. H. Lee, S.Y. Kung, and N. Verma, "Kernel-energy trade-off in machine learning for implantable and wearable biomedical applications," *ICASSP 2011*.