Evolutionary Learning for Soft Margin Problems: A Case Study on Practical Problems with Kernels

WANG, Wenjun, PANG, Wei, BINGHAM, Paul <http://orcid.org/0000-0001-6017-0798>, MANIA, Mania, CHEN, Tzu-Yu and PERRY, Justin

Evolutionary Learning for Soft Margin Problems: *A Case Study on Practical Problems with Kernels*

Wenjun Wang
School of Mathematical and Computer Sciences
Heriot-Watt University
Edinburgh EH14 4AS, Scotland, UK
Email: wenjun.wang@hw.ac.uk
ORCID: 0000-0003-3579-1474

Wei Pang*
School of Mathematical and Computer Sciences
Heriot-Watt University
Edinburgh EH14 4AS, Scotland, UK
Email: w.pang@hw.ac.uk
OCRID: 0000-0002-1761-6659

Paul A. Bingham, Mania Mania and Tzu-Yu Chen
Materials and Engineering Research Institute
Sheffield Hallam University
Sheffield S1 1WB, UK
Email: p.a.bingham@shu.ac.uk
ORCID: 0000-0001-6017-0798

Justin J Perry
Department of Applied Sciences, Northumbria University
Newcastle upon Tyne, NE1 8ST, UK
Email: justin.perry@northumbria.ac.uk
ORCID: 0000-0002-3436-0093

***Abstract*—This paper addresses two practical problems: the classification and prediction of properties for polymer and glass materials, as a case study of evolutionary learning for tackling soft margin problems. The presented classifier is modelled by support vectors as well as various kernel functions, with its hard restrictions relaxed by slack variables to be soft restrictions in order to achieve higher performance. We have compared evolutionary learning with traditional gradient methods on standard, dual and soft margin support vector machines, built by polynomial, Gaussian, and ANOVA kernels. Experimental results for data on 434 polymers and 1,441 glasses show that both gradient and evolutionary learning approaches have their advantages. We show that within this domain the chosen gradient methodology is beneficial for standard linear classification problems, whilst the evolutionary methodology is more effective in addressing highly non-linear and complex problems, such as the soft margin problem.**

Keywords— evolutionary learning, soft margin, support vector, kernel function, slack variables

I. INTRODUCTION

Support vector machines (SVMs) [1] have proved to be an effective classifier for binary classification. Essentially, an SVM aims to find the specific data points (named as support vectors) located both on the positive and negative bounds that can maximise the margin between the data class and the classifier. In linear cases, the classifier is supposed to be a line or hyperplane in feature space, as shown in Fig. 1. Given $\boldsymbol{w}$ and b, a hyperplane can be defined as follows [2, 3]:

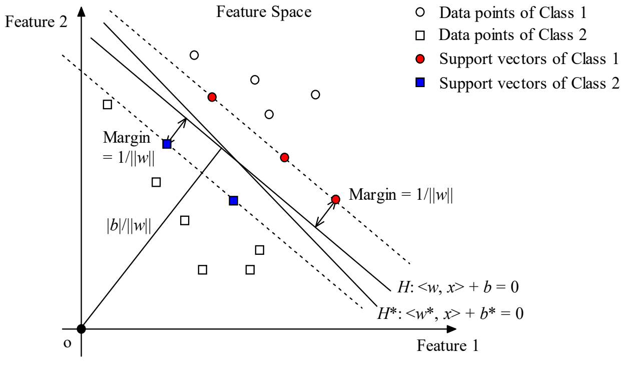

Fig. 1. Linear classifier, support vectors and their margins

$$H = \{\, \boldsymbol{x} \mid <\boldsymbol{w}, \boldsymbol{x}> + \, b = 0\}, \tag{1}$$

where $\boldsymbol{x}$ is a m dimension variable vector in feature space, $\boldsymbol{w}$ is the normal vector of hyperplane, $<\boldsymbol{w}, \boldsymbol{x}>$ stands for the inner product, and b is a constant parameter. The hyperplane H divides the feature space into two parts standing for the two classes, thus a new data point x_i can be classified by calculating:

$$f\,(\,\boldsymbol{w}, b, \boldsymbol{x}_i) = \operatorname{sgn}\,(\,<\boldsymbol{w}, \boldsymbol{x}_i> + \, b\,)\}, \tag{2}$$

where *sgn* stands for the signum function, and $i = 1,\ldots, n$ is the index of observations. As shown in Fig 1, for the same training data set, different parameters $\boldsymbol{w}$ and b will lead to different classifiers with varying margins, while a larger one is more profitable to achieve higher prediction accuracy and lower risk of error. Furthermore, if we term the predicted value $f\,(\boldsymbol{w}, b, \boldsymbol{x}_i)$ as y_i, and normalize $\boldsymbol{w}$ and b in a way to let all the data points closest to the hyperplane satisfy the following:

$$\forall i,\ y_i\,(\,<\boldsymbol{w}, \boldsymbol{x}_i> + \, b\,) \geq 1, \tag{3}$$

the standard linear SVM optimization will be deduced as:

$$\min\ \tfrac{1}{2}\|\boldsymbol{w}\|^2, \tag{4}$$

$$s.t.\ \forall i,\ y_i\,(\,<\boldsymbol{w}, \boldsymbol{x}_i> + \, b\,) \geq 1, \tag{5}$$

where $\|\boldsymbol{w}\|$ denotes the Euclidean norm of vector $\boldsymbol{w}$. For solving the quadratic problem (4) under inequality constraints (5), we usually combine (4) and (5) by Lagrange multipliers [4] as follows:

$$Lp\!:\ \ \min\ \tfrac{1}{2}\|\boldsymbol{w}\|^2 + \textstyle\sum_{i=1}^{n} \alpha_i y_i(\,<\boldsymbol{w}, \boldsymbol{x}_i> + \, b) \tag{6}$$

where α_i (i = 1,…, n) are n positive Lagrange multipliers. Finding a minimum of this combined objective requires a gradient method and it is necessary to transfer the inequality constraints (5) to be inequality constraints by using Wolfe

This research is supported by the Engineering and Physical Sciences Research Council (EPSRC) funded project New Industrial Systems: Manufacturing Immortality (EP/R020957/1).

* Corresponding author.

duality. In short, here we give the final expression of SVM dual problem as follows:

$$Ld\text{:}\ \ \max\ \ \sum_{i=1}^{n}\alpha_i-\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}y_iy_j\alpha_i\alpha_j<\boldsymbol{x}_i,\boldsymbol{x}_j> \qquad (7)$$

$$s.t.\ \forall i,\ \alpha_i\geq 0\ \text{ and }\ \sum_{i=1}^{n}\alpha_iy_i=0 \qquad (8)$$

where n is the number of observations, $<\boldsymbol{x}_i, \boldsymbol{x}_j>$ stands for the kernel function which can be either linear or non-linear depending on the particular problem. By solving the dual problem (7) restricted by (8), we obtain the optimal parameters α_i^*, and further calculate the optimal parameter $\boldsymbol{w}^*$ and b^* to get the classifier for the original problem (4).

Generally, researchers use gradient-based optimization methods [2], [5] to solve (7) and (8). However, there are two tough issues usually encountered in tacking practical problems: (A) Uncertainty over choice of kernel functions. In most cases solving practical problems, such as the polymer and glass classification problems in this work, we do not have prior knowledge of the most appropriate kernel(s). Thus, choosing the kernel by either empirical models or feature mining approaches, is still challenging. (B) The failure of the traditional gradient method. First, in non-linear and large-scale cases [6], some employed kernels would be very complex and might introduce unexpected computational error. Second, for kernels that are not positive semidefinite, the unique global optimum does not exist, which means we cannot approach the optimal solution by using gradient method.

For the first issue (Issue A), a heuristic method for automatic feature selection was designed by using an evolutionary algorithm [7], where the kernels were treated as individuals in the population, and the best kernel will be presented as the final output. However, this will divide the original problem into two stages: kernel selection and problem solving. Recently, ensemble learning approaches have been developed by maintaining a set of kernels and driving them adaptively by the proposed strategies or criteria [3, 7]. However, in this way, some of the kernels in the pre-set ensemble may be not so exact to approximate the ideal kernel. Therefore slack variables were introduced which could relax the hard constraints (5) to become softer constraints [8, 10], so that the kernels used could be equally considered as the ideal unknown one. Of course, the introduced positive slack variables will form an additional objective to be minimized, thus choosing an appropriate weighting parameter for balancing the margin objective and slack objective will affect the final optimum [10, 11].

For the second issue (Issue B), evolutionary learning [13] works, and it is a type of bio-inspired method for solving highly complex, non-linear and larger-scale optimization problems by combining an evolutionary strategy with machine learning [7, 13–17]. Compared to the traditional gradient methods, evolutionary learning has shown advantages in tackling practical problems, especially those which are non-differentiable or very difficult to model mathematically. Evolutionary learning shows the effectiveness in running combinations of multiple SVMs [14], parallel evolutionary algorithms [6], and evolving an ensemble of models [17]. Moreover, the evolutionary learning method is good at dealing with multi-objective optimisation problems [18], which avoids the need of weighting parameters to balance various objectives.

In this research, aiming to tackle polymer and glass materials classification and prediction problems, both of the kernel and slack methods will be used to model the classifiers, and the evolutionary algorithm be carried out on the *Ld* model in (7) and (8). We will treat information such as properties and chemical compositions as problem features, and the types of polymer and glass materials as labels. The classification task is essentially to construct a mapping modelled from the feature data to their labels. These mappings are usually highly coupled by unknown functions and thus are so complex that a linear kernel is not competent. One way to approach this is to use non-linear kernels, such as polynomial, Gaussian [2, 9] and ANOVA [19] kernels.

The rest of this paper is organised as follows: Section II briefly reviews three common non-linear kernels and the soft margin problem by introducing slack variables. An evolutionary learning method is introduced in Section III, and this is compared with a traditional gradient method on two practical problems by using standard, dual, and soft margin SVMs in Section IV. The experimental investigation on different kernels is also carried out in Section IV. Finally, conclusions are drawn, and future work outlined in Section V. The contributions of this paper are summarized as follows:

(1) Two practical problems are presented in this work as a case study for the soft margin problem-solving by evolutionary learning.

(2) The performance of both evolutionary learning and gradient-based methods are demonstrated to compare their particular advantages.

II. Kernel and Slack Methods

A. Kernel Functions for SVM

In the feature space shown by Fig. 1, as we suppose a hyperplane to be the binary classifier, in (7) we have a linear kernel function $<\boldsymbol{x}_i, \boldsymbol{x}_j> = \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j$. However, in most practical cases, the data is non linearly separable, and it needs to be transferred to another feature space by:

$$\phi\text{: }\boldsymbol{x}\in R^M\longmapsto\phi(\boldsymbol{x})\in F\in\mathbb{R}^m \qquad (9)$$

In the new feature space $F \in \mathbb{R}^m$, the data are expected to be separated by a linear SVM. For simplicity, we briefly list three common kernels [9] which will be tested in our subsequent experiments.

1) Polynomial Kernel

Frequently, a polynomial kernel refers to the following case:

$$k_d\,(\boldsymbol{x},\boldsymbol{z})=(<\boldsymbol{x},\boldsymbol{z}>+R\,)^d \qquad (10)$$

where $\boldsymbol{x}$ and $\boldsymbol{z}$ denotes vector variables in R^M, d stands for the degree of polynomial kernel, R is a constant parameter. Expanding the polynomial kernel by using the binomial theorem [20] we have a general expression as:

$$k_d\,(\boldsymbol{x},\boldsymbol{z})=\sum_{s=0}^{d}\binom{\mathrm{d}}{s}R^{d-s}<\boldsymbol{x},\boldsymbol{z}>^s \qquad (11)$$

where $s = 0,1,\ldots d$ is a integer parameter.

2) *ANOVA kernel*

First we define a feature mapping as follows:

$$\phi_d : \boldsymbol{x} \in R^M \longmapsto \phi_{\boldsymbol{A}}(\boldsymbol{x})_{(|\boldsymbol{A}| = d)} \tag{12}$$

where $|\boldsymbol{A}| = d$ denotes the cardinality of set $\boldsymbol{A}$, and $\phi_{\boldsymbol{A}}(\boldsymbol{x})$ is defined as:

$$\phi_{\boldsymbol{A}}(\boldsymbol{x}) = \boldsymbol{x}_1^{i_1}\boldsymbol{x}_2^{i_2}\ldots\boldsymbol{x}_n^{i_n} \tag{13}$$

where $\boldsymbol{A} = (i_1, i_2, \ldots, i_{\mathrm{n}}) \in \{0, 1\}^n$ with a further restriction that:

$$\textstyle\sum_{j=1}^{n} i_j = d \tag{14}$$

The ANOVA kernel is defined as the summation of all expressions that satisfy$|\boldsymbol{A}| = d$, as:

$$k_d(\boldsymbol{x}, \boldsymbol{z}) = <\phi_d(\boldsymbol{x}), \phi_d(\boldsymbol{z})> = \textstyle\sum_{|A|=d} \phi_d(\boldsymbol{x})\phi_d(\boldsymbol{z}) \tag{15}$$

3) *Gaussian kernel*

Given the mean square variance of population as $\sigma > 0$, the Gaussian kernel is defined as:

$$k(\boldsymbol{x}, \boldsymbol{z}) = \exp^{\left(-\frac{||x-z||^2}{2\sigma^2}\right)} \tag{16}$$

A Gaussian kernel is the most widely used kernel and it forms the hidden units of a radial basis function (RBF) network, and hence using this kernel will mean the hypotheses are radial basis function networks [5, 7]. It is therefore also referred to as the RBF kernel.

B. *Slack Method for Non Linearly Seperatable Data*

We now return to the binary classification problem which is non-separated by a linear model, nor even by non-linear kernels. Here we take a linear case as an example. Recalling constraints (5), we now relax them by introducing slack variables as:

$$s.t.\ \ \forall i,\ y_i\,(<\boldsymbol{w}, \boldsymbol{x}_i> + b) \geq 1 - \varepsilon_i \tag{17}$$

where ε_i ($i = 1,\ldots, n$) are positive slack variables for relaxing constraints. In order to minimize the number of wrong classifications, we need to introduce the second objective as a part of (4), as follows:

$$\min\ \ \frac{1}{2}\left|\left|\boldsymbol{w}\right|\right|^2 + \mathrm{C}\textstyle\sum_{i=1}^{n} \varepsilon_i \tag{18}$$

where C is a constant factor parameter which determines the weight of wrong predictions (or not exact predictions). We expect both parts in (18) to be minimum.

By using a Lagrange multiplier and dual transform, the slack variable ε_i vanishes and we can get the dual SVM optimization problem below [18, 20]:

$$Ld\text{: max}\ \ \textstyle\sum_{\mathrm{i}=1}^{n} \alpha_i - \frac{1}{2}\sum_{\mathrm{i}=1}^{n}\sum_{\mathrm{j}=1}^{n} y_i y_j \alpha_i \alpha_j < \boldsymbol{x}_i, \boldsymbol{x}_j > \tag{19}$$

$$s.t.\ \ \forall i,\ \ 0 \leq \alpha_i \leq \mathrm{C} \text{ and } \textstyle\sum_{\mathrm{i}=1}^{n} \alpha_i y_i = 0 \tag{20}$$

This term can be solved by the evolutionary algorithm introduced in Section III.

III. Evolutionary Computation

In this section we combine evolutionary computation with SVM learning. Consider optimization problem (19) with constraint (20), for simplicity we re-term them to a standard format as:

$$max\ \ f(\boldsymbol{\alpha}) \tag{21}$$

$$s.t.\ \ \ g(\boldsymbol{\alpha})=0 \tag{22}$$

$$\forall i,\ \ 0 \leq \alpha_i \leq C \tag{23}$$

Next, we will introduce the main steps of evolutionary computation to solve the above optimization problem.

A. *Population Initialization and Evaluation*

Evolutionary computation starts from an initial population composed of a pre-specified number $\boldsymbol{N}$ of individuals representing potential solutions. In this work, solution vector $\boldsymbol{\alpha}$ is encoded by a real number bounded in $[0, \mathrm{C}]$, i.e. the searching space for this problem is $[0, \mathrm{C}]^n$. Generally, we generate individuals by uniform distribution on the hyperplane defined by (22) as the initial population, then evaluate the quality of each individual by calculating (21). For the maximum problem (21), an individual with higher fitness value calculated by (21) means that this individual will be given more chance to generate offspring in the population.

B. *Evolution Strategy*

Evolutionary strategy is a bio-inspired method for search and optimisation problems, and it mimics the natural environments, criteria and processes. There are many well-used evolutionary operators, such as simulated binary crossover (SBX) [22], polynomial-based mutation (PM) [23] and others [11, 13]. Here we take SBX as an example to introduce evolutionary strategy.

1) *Parent Selection*

For passing "good" properties to offspring, elite individuals with higher fitness values have more chance to be selected as parents. For SBX, we select two parents by the tournament method with an empirical fraction parameter TF= 0.75 [23].

2) *Generation by SBX and PM*

In SBX generation, suppose that $\boldsymbol{\alpha}^1$ and $\boldsymbol{\alpha}^2$ are selected parents, for the randomly selected entry j (j = 1, 2, …, n) of $\boldsymbol{\alpha}^1$ and $\boldsymbol{\alpha}^2$, their offspring $\boldsymbol{o}^1$ and $\boldsymbol{o}^2$ are generated by:

$$\boldsymbol{o}_j^1 = 0.5\left[\left(w_j + v_j\right) - \beta_1 \times \left(w_j - v_j\right)\right] \tag{24}$$

$$\boldsymbol{o}_j^2 = 0.5\left[\left(w_j + v_j\right) - \beta_2 \times \left(w_j - v_j\right)\right] \tag{25}$$

where $w_j = \max(\alpha_j^1, \alpha_j^2)$, $v_j = \min(\alpha_j^1, \alpha_j^2)$, β_k $(k = 1, 2)$ are defined as follows:

$$\beta_k = \begin{cases} [r_k \times a_k]^{\frac{1}{\eta+1}} & \text{if } r_k \leq \frac{1}{a_k} \\ [\frac{1}{(2 - r_k \times a_k)]}^{\frac{1}{\eta+1}} & \text{otherwise} \end{cases} \tag{26}$$

where r_k (k = 1, 2) are random numbers uniformly distributed in [0, 1]. Integer k is randomly chosen as 1 or 2 to determine a final offspring from $\boldsymbol{o}^1$ and $\boldsymbol{o}^2$. η is the crossover distribution index; a_k are defined as follows (assuming that $w_j \neq v_j$):

$$a_k = \begin{cases} 2 - [1 + 2\frac{(v_j - l_j)}{(w_j - v_j)}]^{-(\eta+1)} & \text{if } k = 1 \\ 2 - [1 + 2\frac{(u_j - w_j)}{(w_j - v_j)]}^{-(\eta+1)} & \text{if } k = 2 \end{cases} \tag{27}$$

where l_j and u_j are the lower and upper bounds of the j-th decision variable, respectively.

In PM, we randomly choose the j-th entry of individual $\boldsymbol{\alpha}$ to be evolved by the following equation:

$$o_j = \alpha_j + \delta_j\left(u_j - l_j\right) \tag{28}$$

where parameter δ_j is calculated by:

$$\delta_j = \begin{cases} 2r + \frac{1}{[(1-2r)(1-\delta)^{\eta_m+1}]^{\eta_m+1}} - 1 & \text{if } r \leq 0.5 \\ 1 - \frac{1}{[2(1-r) + 2(r-0.5)(1-\delta)^{\eta_m+1}]^{\eta_m+1}} & \text{else} \end{cases} \tag{29}$$

In the above, r are uniformly distributed random numbers in [0, 1], and δ is defined as follows:

$$\delta = \min\ [\,(\alpha_j - l_j), (u_j - \alpha_j)]/(u_j - l_j) \tag{30}$$

In (29), η_m is the mutation distribution parameter which controls the expectation of disturbance. Generally, we set it as:

$$\eta_m = 100 + t \tag{31}$$

where t denotes the number of evolving generations. In this case the mutation property will be calculated as:

$$p_m = \frac{1}{n} + \frac{t}{t_{\max}}\left(1 - \frac{1}{n}\right) \tag{32}$$

where t_{max} is the maximum number of evolving generations. As parameter η_m is a polynomial with respect to t, this is called polynomial mutation.

3) *Environment Selection*

These newly generated offspring individuals are then combined with their parents to form a new population. The combined population is maintained by environmental selection which mimics natural criteria [24]. For example, individuals with higher fitness values will be more likely to survive than those with lower fitness values. Another popular criterion is the diversity metric [24, 25] of a population, e.g. an individual with higher diversity will have more chance to be maintained in the population, in order to avoid the whole population falling into local optima. It should be noted that there are many criteria for environmental selection, and each criterion has its own advantages for population maintenance, thus the selection criterion should be designed specifically for a given practical problem. In this work, we will only use the fitness value of an individual for environmental selection.

C. *Termination Criterion*

- *Criterion 1*: For a pre-specified number $\boldsymbol{N}_{\text{stay}}$, e.g. 100, when the best fitness of elite individual (or other quality metric) is not improved for $\boldsymbol{N}_{\text{stay}}$ epochs.

- *Criterion 2*: Between two consecutive epochs, the improvement of fitness of the best elite individual should not be more than a small specified number as a threshold.

- *Criterion 3*: When the epoch reaches a pre-specified number $\boldsymbol{N}_{\text{epoch}}$ representing the maximum epoch.

It should be noted that we can use one of the above criteria as a termination condition, while sometimes all three criteria should be satisfied for higher qualified convergence.

IV. Experimental Investigation

A. *Two Practical Problems with Associated Datasets*

- *Problem* 1*:* Polymer Classification [27]

The work of Huan *et al* [26] provides a set of computed polymer data with 7 materials properties, namely atom type (AT), total atom number (AN), band gap (BG), atomization energy (AE), dielectric constant of electron (DE), dielectric constant of ion (DI) and total dielectric constant (DC). Within this data set are polymers labelled as distinct classes: (1) 'organic molecular crystal' and (2) 'organic polymer crystal'. The practical problem was to construct a classifier by using SVMs with various kernels and then train and test it with training and test sets, respectively.

This dataset contains 434 polymers [27], with 124 polymers classed as being of type 'organic molecular crystal' and 310 polymers as being of type 'organic polymer crystal'. For this study, we randomly selected 40% from each set of the two types to construct one test dataset. The remaining data was combined to form the training set. This gave a training set with 262 polymer data, and a test set with 172 polymer data. Each entry of polymer data was recorded as shown in Table I.

TABLE I. Two example entries in the Polymer Dataset

ID	Feature Variables (x)							Labels (y)
	AT	***AN***	***BG***	***AE***	***DE***	***DI***	***DC***	**Type**
336	4	76	3.4	□5.8	3.0	0.78	3.8	Organic polymer
944	3	60	6.2	□5.0	2.5	0.53	3.1	Organic molecular

- *Problem 2:* Glass Classification

We used a data set for 1,441 glasses composed of SiO_2, Al_2O_3, B_2O_3, Na_2O, K_2O, MgO and CaO. We recorded their chemical compositions as feature variables (formatted so that the sum of these variables summed to 100), and converted their glass transition temperatures (Tg) to form a binary label of 'High Tg' or 'Low Tg' by follows:

$$y_i = \begin{cases} \text{High} & \text{if } Tg_i \geq \overline{Tg} \\ \text{Low} & \text{if } Tg_i < \overline{Tg}\,, \end{cases} \tag{33}$$

where $\overline{Tg} = 566$ stands for the mean value of Tg of all 1,441 samples. Examples of data entry are shown in Table II.

TABLE II. Two Example Entries in the Glasses Dataset

ID	**Compositional Variables (x)**							**Labels (y)**	
	SiO_2	Al_2O_3	B_2O_3	Na_2O	K_2O	MgO	CaO	***Tg/°C***	**Type**
25	30	0	45	15	0	0	10	475	Low
128	80	0	10	0	10	0	0	658	High

The task was to construct a model which can classify glasses with high or low *Tg* with the aim that the constructed and trained model could predict *Tg* values using the labels 'High *Tg*' or 'Low *Tg*' when presented with new glass compositions.

B. Experiment and Algorithm Setting

SVM was used to tackle the classification and prediction tasks with three commonly used kernels: the polynomial, Gaussian, and AVONA kernels. Both the traditional gradient-based training and evolutionary methods for dual soft margin problems are assessed in this research.

TABLE III. DETAILS OF THE EXPERIMENTS

Solver	**SVMs**			
	Standard: (4) & (5)	**Dual: (7) & (8)**	**Soft Margin: (19) & (20)**	
	C = NULL	C = ∞	C > 0	C > 0
Polynomial (P)	GM	EA	GM	EA
Guassian (G)	GM	EA	GM	EA
ANOVA (A)	GM	EA	GM	EA

In Table III, GM and EA stand for the gradient method and evolutionary algorithm respectively. As the standard SVM cannot be solved by EA, we only use GM to solve (4) under (5). The dual SVM (7) under (8) is essentially equal to the standard SVM (4) under (5), thus we employ EA to solve (7) under (8). It should be noted that we should generate solution individual $\boldsymbol{\alpha}$ under restriction (8), just as that used in solving (19) under (20) with C = ∞. In summary, totally we have carried out 12 experiments as shown in Table III, which basically contains three SVMs on three kernels solved by GM and EA. For the soft margin model, we used both gradient and evolutionary methods to solve them, with the primary experiment settings are shown in Table IV.

TABLE IV. PRIMARY PARAMETER SETTINGS OF EXPERIMENTS

Parameter Setting for Evolutionary Algorithm			
Population Size N	**Constant** C **in (23)**	**Max Epoch** N_{epoch}	**Non-improved Epoch No.** N_{stay}
100	{0.01, 0.1, 0, 1, 10^6}	100,000	30
Cross Validation Folds No.	**Crossover Parameter** η **in (26)**	**Mutation Parameter** η_m **in (31)**	**Tournament Fraction** *TF* **in [22]**
10	20	1	0.75
Parameter Setting for Kernel Models			
Polynomial Kernel Degree d **in (10)**	**ANOVA Kernel Degree** d **in (14)**	**Gaussian Kernel, Parameter** σ **in (16)**	**Slack Variables** ε_i (i =1, ..., n) **in (17)**
{1, 2, 3}	1	$\{\sqrt[2]{0.5},\sqrt[2]{0.05}\}$	0.1

In Table IV, the upper part shows the parameter setting for the evolutionary algorithm, where $\boldsymbol{N}$ stands for the pre-set number of individuals in the initial population, i.e. population size mentioned in section III.A. C is the parameter for bounding in (23) as well as the penalty parameter in (18), $\boldsymbol{N}_{\text{epoch}}$ stands for the pre-set maximum number of epoches, and $\boldsymbol{N}_{\text{stay}}$ denotes the pre-set epoch number for the so-called non-improvement evolution. We used 10-fold cross-validation for our experiments, and the parameters for generating new offspring are η, η_m and *TF*. The lower part of Table IV lists the primary parameters used in the kernels, where *d* stands for the kernel degrees used for polynomial and ANOVA kernels. σ denotes the mean-square variance in Gaussian kernel, and ε_i is the i^{th} slack variable in (17). It should be noted that some of the parameters arise from literature and some of them are drawn from trials. For instance, the bounding C in (23) in our experiment comes from an empirical set {0.01, 0.1, 0, 1, 10^6}, as there is no prior knowledge of these practical problems presented in this work.

C. Metrics and Results Analysis

As both practical problems are binary classification tasks, we can record outputs as one of four commonly used variables, i.e. the false positive (FP), true positive (TP), false negative (FN) and true negative (TN) values as results. In addition, the metrics of accuracy (*A*), precision (*P*), recall (*R*) and F_1-score (F_1) are calculated as follows:

$$A = \frac{\text{TP+TN}}{\text{FP+TP+TN+FN}} \times 100\% \tag{34}$$

$$P = \frac{\text{TP}}{\text{FP+TP}} \times 100\% \tag{35}$$

$$R = \frac{\text{TP}}{\text{FP+TN}} \times 100\% \tag{36}$$

$$F_1 = \frac{2PR}{P+R} = \frac{\text{2TP}}{\text{2TP+FP+FN}} \times 100\% \tag{37}$$

Due to space limitations, we do not listed here all metrics results, but only the accuracy *A* for illustration. In the 10-fold cross-validation, we calculate the mean accuracy $\overline{A}$ and standard deviation σ_A of 10 accuracy scores as follows:

$$\overline{A} = \frac{1}{10} \sum_{i=1}^{10} A_i \tag{38}$$

$$\sigma_A = \sqrt{\frac{1}{10-1} \sum_{i=1}^{10} (A_i - \overline{A})^2} \tag{39}$$

For evolutionary computation, the mean values of 30 independent runs were used for final results.

- *Results of Polymer Classification and Prediction*

As shown in Table V, the standard SVMs (4), (5) used the gradient method to enable them to be solved. For the soft margin problems (19), (20) both gradient and evolutionary methods were used with various values of parameter C, with only the best results listed in Table V. For dual SVMs (7), (8) only the evolutionary method was used, but with C = 10^6 replacing C = ∞.

There are two points to be noted here: (1) In fact, the standard SVM (C=NULL) and dual SVM (solved by EA with C=∞) are equal. However, when C = 10^6 was used to replace C = ∞, some errors were introduced. Thus, the gradient method should be preferred for the standard and the dual SVM model. (2) Each evolutionary computation starts from a random initialized population, thus a number[1] of independent runs are

[1] In this work, 30 × independent run are carried out on each trial.

needed. However, the aim of evolutionary computation is to find the support vectors via solving (19) or (21), and all 30 runs with respect to any case had converged to give the same support vectors. Thus, we cannot obtain the statistical analysis for the classification results, though we could obtain the results of $\boldsymbol{\alpha}$ in each of the 30 runs with slight differences.

TABLE V. (A) RESULTS OF POLYMER CLASSIFICATION

Cross Validation on Training Data Set							
Model	**Kernel**	**FN**	**TP**	**FP**	**TN**	$\bar{A}$	σ_A
(4), (5) by GM	P (C=0)	24	169	17	52	84.4%	6.2%
	G (C=0)	15	169	17	61	87.8%	6.8%
	A (C=0)	15	167	19	61	87.0%	6.3%
(7), (8) by EM	P (C=10^6)	8	148	38	68	82.4%	8.5%
	G (C=10^6)	10	150	36	66	82.4%	8.0%
	A (C=10^6)	2	147	39	74	84.3%	7.0%
(19), (20) by GM	P (C=0.1)	23	169	17	53	84.8%	5.6%
	G (C=0.1)	39	180	6	37	82.8%	4.6%
	A (C=0.1)	16	167	19	60	86.6%	6.6%
(19), (20) by EM	P (C=0.01)	18	170	16	58	87.0%	6.2%
	G (C=1)	24	171	15	52	85.1%	2.9%
	A (C=0.1)	9	161	25	67	87.0%	3.8%

TABLE V. (B) RESULTS OF POLYMER PREDICTION

Prediction on Test Data Set						
Model	**Kernel**	**FN**	**TP**	**FP**	**TN**	A
(4), (5) by GM	P (C=0)	17	114	10	31	84.3%
	G (C=0)	11	116	8	37	89.0%
	A (C=0)	5	104	20	43	85.5%
(7), (8) by EA	P (C=10^6)	9	102	22	39	82.0%
	G (C=10^6)	8	104	20	40	83.7%
	A (C=10^6)	5	97	27	43	81.4%
(19), (20) by GM	P (C=0.1)	17	115	9	31	84.9%
	G (C=0.1)	29	121	3	19	81.4%
	A (C=0.1)	15	120	4	33	89.0%
(19), (20) by EA	P (C=0.01)	18	114	10	30	83.7%
	G (C=1)	19	117	7	29	84.9%
	A (C=0.1)	11	115	9	37	88.4%

With respect to the methods used, for soft margin SVM, the evolutionary method performs better at cross validation in Table IV. In Table V, GM was a completely winner for 3 kernels, when comparing '(4), (5) by GM' with '(7), (8) by EA' as these two models are equal, and win 2 out of 3 when comparing '(19), (20) by GM' to '(19), (20) by EA'. EA wins only in 1 out of 3 on the Gaussian kernel.

With respect to the performance of kernels in Table IV and V, the ANOVA kernel wins 5 out of 8 experiments, and holds rank 2 for twice; while the Gaussian and polynomial kernel win 2 out of 8 experiments respectively (there is a draw best between ANOVA and polynomial kernel). However, it should be noted that the Gaussian kernel used in basic SVM exhibited the highest prediction accuracy which was also achieved by dual SVM with ANOVA kernel.

- *Results of Glass Classification and Prediction*

As shown in Table VI, the gradient method on standard SVM (4), (5) performs better than the evolutionary algorithm on dual SVM (7), (8), on all three kernels. While in the soft margin SVM, the evolutionary method outperforms slightly the gradient method, as shown in both cross validation in Table VI (A) and prediction in Table VI (B). Moreover, it is noted that the FN scores are obviously different between the gradient method and the evolutionary method, however, the reasons for this still require further investigation.

TABLE VI. (A) RESULTS OF GLASS CLASSIFICATION

Cross Validation on Training Data Set							
Model	**Kernel**	**FN**	**TP**	**FP**	**TN**	$\bar{A}$	σ_A
(4), (5) by GM	P (C=0)	8	223	145	633	84.8%	2.8%
	G (C=0)	6	260	108	635	88.7%	3.8%
	A (C=0)	9	219	149	632	84.3%	2.0%
(7), (8) by EA	P (C=10^6)	125	286	82	516	79.5%	5.5%
	G (C=10^6)	116	337	31	525	85.4%	3.8%
	A (C=10^6)	76	266	102	565	82.4%	3.0%
(19), (20) by GM	P (C=0.1)	8	219	149	633	84.4%	2.4%
	G (C=0.1)	4	224	144	637	85.3%	3.0%
	A (C=0.1)	4	224	144	637	85.3%	3.0%
(19), (20) by EA	P (C=0.01)	53	271	97	588	85.1%	4.4%
	G (C=1)	39	311	57	602	90.5%	3.5%
	A (C=0.1)	59	272	96	582	84.6%	2.4%

TABLE VI. (B) RESULTS OF GLASS PREDICTION

Prediction on Test Data Set						
Model	**Kernel**	**FN**	**TP**	**FP**	**TN**	A
(4), (5) by GM	P (C=0)	0	100	59	273	86.3%
	G (C=0)	3	115	44	270	89.1%
	A (C=0)	3	101	58	270	85.9%
(7), (8) by EA	P (C=10^6)	50	123	36	223	80.1%
	G (C=10^6)	37	149	10	236	89.1%
	A (C=10^6)	35	111	48	238	80.8%
(19), (20) by GM	P (C=0.1)	0	100	59	270	86.3%
	G (C=0.1)	0	97	62	273	85.7%
	A (C=0.1)	0	97	62	273	85.7%
(19), (20) by EA	P (C=0.01)	18	121	38	255	87.0%
	G (C=1)	10	138	21	263	92.8%
	A (C=0.1)	24	121	38	249	85.7%

In the soft margin problem, the Gaussian kernel solved by the evolutionary method achieved high accuracies at 90.5% (in Table VI (A)) and 92.8% (in Table VI (B)) in cross validation and predictions, respectively. Thus we can conclude that the Gaussian kernel is more promising than the other two in the glass *Tg* classification and prediction.

- *Further Analysis*

a) *Parameters.* All parameters in Table IV may affect the final results in Table V and Table VI. Thus, it is difficult to prepare absolutely fair comparisons between SVM models, kernels and solvers.

b) *Cost.* The evolutionary method consumes more computational resource than the gradient method, especially as the scale of dataset increases.

c) *Performance.* The evolutionary method achieved the best results on soft margin SVM with ANOVA kernel in Table V (A), with the Gaussian kernel in Table VI (A) and Table VI (B), and hence this exemplifies the advantages of the evolutionary method in solving complex practical problems.

V. Conclusions and Future Work

In this paper, we have reviewed three SVMs and kernel functions, as well as evolutionary methods for solving soft margin SVM model. We tested these approaches on two practical problems, i.e. the classification and prediction of polymer and glass materials datasets. By analysing experimental results, conclusions can be drawn as follows:

(1) By selecting the proper parameters, we can achieve more than 80% accuracy in both validation and prediction in nearly all cases studied here.

(2) On the polymer dataset, the kernels and SVM models performed similarly, while with the glass dataset, the Gaussian kernel on soft margin SVM outperformed the other methods.

(3) Both evolutionary and gradient methods have their specific advantages. It should be emphasized that the evolutionary method has great potential for tackling more complex practical problems, and so this method needs further investigation when solving practical problems.

For our future work, we would like to study further practical cases and feature learning models that can be solved by evolutionary methods. These current case studies suggest that when facing a practical problem, as it is unknown at the outset which model will perform the best, an ensemble strategy [8] which contains all available approaches should be used.

Acknowledgments

This research is supported by the Engineering and Physical Sciences Research Council (EPSRC) funded Project on New Industrial Systems: Manufacturing Immortality (EP/R020957/1). The authors are also grateful to the Manufacturing Immortality consortium.

References

[1] Christopher M. Bishop, *Pattern Recognition and Machine Learning*. Springer-Verlag New York, 2006.

[2] I. Mierswa, "Evolutionary learning with kernels: A generic solution for large margin problems," *GECCO 2006 - Genet. Evol. Comput. Conf.*, vol. 2, pp. 1553–1560, 2006.

[3] W. Wang, N. G. Moreau, Y. Yuan, P. R. Race, and W. Pang, "Towards machine learning approaches for predicting the self-healing efficiency of materials," *Comput. Mater. Sci.*, vol. 168, pp. 180–187, Oct. 2019, doi: 10.1016/j.commatsci.2019.05.050.

[4] Dimitri P. Bertsekas, *Nonlinear Programming*, 2nd ed. Athena Scientific, 1999.

[5] K. R. Müller, S. Mika, G. Rätsch, K. Tsuda, and B. Schölkopf, "An introduction to kernel-based learning algorithms," *IEEE Trans. Neural Networks*, vol. 12, no. 2, pp. 181–201, 2001, doi: 10.1109/72.914517.

[6] N. Arana-Daniel, A. A. Gallegos, C. López-Franco, A. Y. Alanís, J. Morales, and A. López-Franco, "Support vector machines trained with evolutionary algorithms employing kernel adatron for large scale classification of protein structures," *Evol. Bioinforma.*, vol. 12, pp. 285–302, 2016, doi: 10.4137/EBO.S40912.

[7] H. L. Huang and F. L. Chang, "ESVM: Evolutionary support vector machine for automatic feature selection and classification of microarray data," *BioSystems*, vol. 90, no. 2, pp. 516–528, 2007, doi: 10.1016/j.biosystems.2006.12.003.

[8] W. Wang *et al.*, "An Effective Ensemble Framework for Multi-Objective Optimization," *IEEE Trans. Evol. Comput.*, vol. 23, no. 4, pp. 645–659, 2018, doi: 10.1109/TEVC.2018.2879078.

[9] J. Shawe-Taylor and N. Cristianini, *Kernel Methods for Pattern Analysis*. Cambridge University Press, 2004.

[10] T.-M. Huang, V. Kecman, and I. Kopriva, *Kernel Based Algorithms for Mining Huge Data Sets*. Springer-verlag Berlin Herdelberg, 2006.

[11] G. Blanchard, O. Bousquet, and P. Massart, "Statistical performance of support vector machines," *Ann. Stat.*, vol. 36, no. 2, pp. 489–531, 2008, doi: 10.1214/009053607000000839.

[12] T. Hofmann, B. Schölkopf, and A. J. Smola, "Kernel methods in machine learning," *Ann. Stat.*, vol. 36, no. 3, pp. 1171–1220, 2008, doi: 10.1214/009053607000000677.

[13] Z.-H. Zhou, Y. Yu, and C. Qian, *Evolutionary Learning: Advances in Theories and Algorithms*. Singapore: Springer Singapore, 2019.

[14] R. Stoean, M. Preuss, D. Dumitrescu, and C. Stoean, "Evolutionary support vector regression machines," *Proc. 8th Int. Symp. Symb. Numer. Algorithms Sci. Comput. SYNASC 2006*, no. 13, pp. 330–335, 2006, doi: 10.1109/SYNASC.2006.39.

[15] M. L. Dantas Dias and A. R. R. Neto, "Evolutionary support vector machines: A dual approach," *2016 IEEE Congr. Evol. Comput. CEC 2016*, no. October 2017, pp. 2185–2192, 2016, doi: 10.1109/CEC.2016.7744058.

[16] W. K. HHrdle, D. D. Prastyo, and C. M. Hafner, "Support Vector Machines with Evolutionary Feature Selection for Default Prediction," *SSRN Electron. J.*, 2012, doi: 10.2139/ssrn.2894201.

[17] X. Yao and Y. Liu, "Evolving Neural Network Ensembles by Minimization of Mutual Information," *Int. J. Hybrid Intell. Syst.*, vol. 1, no. 1–2, pp. 12–21, 2004, doi: 10.3233/his-2004-11-204.

[18] H.-J. Luethi, *Convex Optimization*, vol. 100, no. 471. 2005.

[19] G. Wahba, Y. Wang, C. Gu, and R. Klein, "Smoothing spline ANOVA for exponential families, with application to the Wisconsin Epidemiological Study of Diabetic Retinopathy," *Ann. Stat.*, vol. 23, no. 6, pp. 1865–1895, 1995, [Online]. Available: http://www.jstor.org/stable/2242776.

[20] Vladimir N. Vapnik, *The Nature of Statistical Learning Theory*, 2nd ed. Springer-Verlag New York, 2000.

[21] W. Lu, R. Xiao, J. Yang, H. Li, and W. Zhang, "Data mining-aided materials discovery and optimization," *J. Mater.*, vol. 3, no. 3, pp. 191–201, 2017, doi: 10.1016/j.jmat.2017.08.003.

[22] K. Deb and R. B. Agrawal, "Simulated Binary Crossover for Continuous Search Space The crossover operator is believed to be the main search operator in the working of a genetic," 1994.

[23] C. A. C. Coello, G. B. Lamont, and D. A. Van Veldhuizen, *Evolutionary Algorithms for Solving Multi-Objective Problems Second Edition*. .

[24] D. Hutchison, *Evolutionary Multi-Criterion Optimization*. 2015.

[25] K. Deb, A. Member, A. Pratap, S. Agarwal, and T. Meyarivan, "A Fast and Elitist Multiobjective Genetic Algorithm :," vol. 6, no. 2, pp. 182–197, 2002.

[26] A. Dobnikar, N. C. Steele, D. W. Pearson, R. F. Albrecht, K. Deb, and S. Agrawal, "A Niched-Penalty Approach for Constraint Handling in Genetic Algorithms," *Artif. Neural Nets Genet. Algorithms*, no. 3, pp. 235–243, 1999, doi: 10.1007/978-3-7091-6384-9_40.

[27] T. D. Huan, A. Mannodi-Kanakkithodi, C. Kim, V. Sharma, G. Pilania, and R. Ramprasad, "A polymer dataset for accelerated property prediction and design," *Sci. Data*, vol. 3, pp. 1–10, 2016, doi: 10.1038/sdata.2016.12.