A Graph Theory Approach To QP Problem Reformulation: An Example With SVM

William Brendel  william.brendel@snap.com
Snap Inc. Research
Luis Marujo  luis.marujo@snap.com
Snap Inc. Research

Abstract

Quadratic programming (QP) problem reformulations have been studied for decades [Sherali and Tuncbilek(1995), Nemirovski and Shapiro(2006), Anstreicher(2009)], [Zheng et al.(2012)Zheng, Sun, Li, and Cui, Wu and Jiang(2017)], but rarely linked to Graph Theory. Indeed, typical reformulations focus on convexifying a non-convex QP problem, making the objective function differentiable, optimizing on the continuous domain while ensuring the final solution is binary, or adding regularizers and Lagrangian coefficients to optimize the dual problem. In this paper, we take SVM as an example to demonstrate that QP problems can also be reformulated using the same mechanism as P/NP problem reduction, overcoming speed and memory footprint limitations from other types of reformulation. We show that SVM is comparable to a soft weighted edge independent set problem where the amount of support vectors per class is balanced, thus it can also be reformulated as a maximum weighted clique problem (MWC) with the same class balancing constraint. After adapting the sequential minimal optimization (SMO) algorithm [Platt(1998), Fan et al.(2005)Fan, Chen, and Lin] to our new MWC formulation, we demonstrate that such reformulation leads to improved performance (7~36 time faster to train, and sparser solution for comparable accuracy).

1 Introduction

In the fields of Machine Learning, Natural Language Processing, Computer Graphics, and Computer Vision, problems involving quadratic programming (QP) formulation are abundant. It is not surprising, since many applications are formulated as graph theory problems, where typical objective functions to be optimized contain unary potentials related to nodes, and binary potentials related to edges. For the last 50 years, graph theory has simply been supporting an ocean of computer science applications [Deo(1974), Singh and Vandana(2014)]. To name a few examples, image segmentation as been formulated as minimum cut [Shi and Malik(2000), Dhillon et al.(2007)Dhillon, Guan, and Kulis], maximum weight independent set [Brendel and Todorovic(2010)], maximum weight clique [Pavan and Pelillo(2007), Zemene and Pelillo(2016)] and minimum spanning tree [Felzenszwalb and Huttenlocher(2004)] problems [Peng et al.(2013)Peng, Zhang, and Zhang]. Multi-object tracking has been formulated as maximum weight independent set [Brendel et al.(2011)Brendel, Amer, and Todorovic] and generalized minimum and maximum clique [Zamir et al.(2012)Zamir, Dehghan, and Shah, Dehghan et al.(2015)Dehghan, Modiri Assari, and Shah] problems. Since graph problems can be reduced to one another, QP problems can also be reformulated into problems that can be solved more efficiently. Various cut problems are reformulated as spectral clustering problems that can be optimized via weighted kernel k-means algorithms [Dhillon et al.(2007)Dhillon, Guan, and Kulis, Li and Chen()], achieving real-time computation performances. In Tsang et al.’s work [Tsang et al.(2005)Tsang, Kwok, and Cheung], SVM and Support Vector Clustering (SVC) are formulated as minimum enclosing ball problems, obtaining provably approximate optimal solutions in linear time, with a space complexity independent of the problem size. This work demonstrates how SVM formulation [Ben-Hur et al.(2005)Ben-Hur, Horn, Siegelmann, and Vapnik] can be viewed as an independent set problem [Tsang et al.(2005)Tsang, Kwok, and Cheung, Hager and Hungerford(2015)], and thus be reduced to other graph problems.

Review of SVM Dual Formulation And SMO: SVM aims at learning boundaries between feature vectors \( \{x^k\}_{k=1}^n \) associated to different classes \( \{y^k\}_{k=1}^n \). When classes are binary, the SVM dual formulation has the

The dual SVM formulation is the same as the maximum weighted independent (MWI) set formulation. The independent set QP formulation has the form of

$$
\alpha^* \leftarrow \text{arg max}_{\alpha \in [0, 1]^n} \quad \mathcal{F}(\alpha) : 1^T \alpha - \frac{1}{2} \alpha^T H \alpha
$$

s.t. \( y^T \alpha = 0 \) with \( y \in \{-1, 1\}^n \)

where \( H_{ij} = y_i y_j K(x^i, x^j) \) and \( x^k \in \mathbb{R}^d \)

and

$$
\mathcal{F}(\alpha) = \frac{1}{2} \alpha^T H \alpha + \frac{1}{2} \nu \alpha^T \alpha
$$

where \( \alpha_k > 0 \) means that \( x^k \) is a support vector for class \( y_k \). I.e. the boundary is defined as a linear combination of \( \{x^k | \alpha_k > 0\} \) in a vector space where the dot product is induced by the kernel function \( K \). SMO was originally designed to train a support vector machine that requires the solution of the very large SVM dual QP optimization problem [Platt(1998)] defined above. SMO belongs to the family of SQP algorithms [Gill et al.(2005)Gill, Murray, and Saunders]. It breaks a large QP problem into a series of smaller QP problems, each of which optimizes a quadratic model of the objective subject to a linearization of the constraints. The method is equivalent to applying Newton’s method to the Karush–Kuhn–Tucker (KKT) conditions of the QP problem. In SMO, the small QP problems involve only two variables. These small QP problems are solved analytically, thus avoiding the use of a time-consuming numerical QP optimization at each iteration. The amount of memory required for SMO is linear in the training set size \( n \), which allows SMO to handle very large input sets.

**Contributions:** In this work we show how to reformulate a QP problem using graph theory. More precisely, we relate the SVM dual QP formulation Eqn. (1) to a maximum independent set formulation, which gives a new interpretation on the support vector selection process. Then, in the same way we reduce an independent set problem to a maximum clique problem by taking the complement of a graph, we reformulate the SVM dual QP formulation as a dominant set QP [Pavan and Pelillo(2007), Zemene and Pelillo(2016)], the later being used to define the maximum weighted clique (MWC) problem [Hager et al.(Hager, Phan, and Zhang, Hager and Hungerford(2015)]. We show that our MWC formulation involves Mercer distance kernels, instead of Mercer similarity kernels, and demonstrate how to construct such kernel, leading to new families of kernels. Finally, we show that our MWC formulation has computational advantages while preserving comparable accuracy compared to the standard LIBSVM implementation of the SVM dual QP formulation.

## 2 Problem Setup And Reformulation

The independent set QP formulation has the form of

$$
\alpha^* \leftarrow \text{arg max}_{\alpha \in [0, 1]^n} \quad \mathcal{F}(\alpha) : 1^T \alpha - \frac{1}{2} \alpha^T H \alpha
$$

s.t. \( y^T \alpha = 0 \) with \( y \in \{-1, 1\}^n \)

where \( H_{ij} = y_i y_j K(x^i, x^j) \) and \( x^k \in \mathbb{R}^d \)

and

$$
\mathcal{F}(\alpha) = \frac{1}{2} \alpha^T H \alpha + \frac{1}{2} \nu \alpha^T \alpha
$$

where \( \alpha_k > 0 \) means that \( x^k \) is a support vector for class \( y_k \). I.e. the boundary is defined as a linear combination of \( \{x^k | \alpha_k > 0\} \) in a vector space where the dot product is induced by the kernel function \( K \). SMO was originally designed to train a support vector machine that requires the solution of the very large SVM dual QP optimization problem [Platt(1998)] defined above. SMO belongs to the family of SQP algorithms [Gill et al.(2005)Gill, Murray, and Saunders]. It breaks a large QP problem into a series of smaller QP problems, each of which optimizes a quadratic model of the objective subject to a linearization of the constraints. The method is equivalent to applying Newton’s method to the Karush–Kuhn–Tucker (KKT) conditions of the QP problem. In SMO, the small QP problems involve only two variables. These small QP problems are solved analytically, thus avoiding the use of a time-consuming numerical QP optimization at each iteration. The amount of memory required for SMO is linear in the training set size \( n \), which allows SMO to handle very large input sets.

**Contributions:** In this work we show how to reformulate a QP problem using graph theory. More precisely, we relate the SVM dual QP formulation Eqn. (1) to a maximum independent set formulation, which gives a new interpretation on the support vector selection process. Then, in the same way we reduce an independent set problem to a maximum clique problem by taking the complement of a graph, we reformulate the SVM dual QP formulation as a dominant set QP [Pavan and Pelillo(2007), Zemene and Pelillo(2016)], the later being used to define the maximum weighted clique (MWC) problem [Hager et al.(Hager, Phan, and Zhang, Hager and Hungerford(2015)]. We show that our MWC formulation involves Mercer distance kernels, instead of Mercer similarity kernels, and demonstrate how to construct such kernel, leading to new families of kernels. Finally, we show that our MWC formulation has computational advantages while preserving comparable accuracy compared to the standard LIBSVM implementation of the SVM dual QP formulation.

## 2 Problem Setup And Reformulation

The independent set QP formulation has the form of

$$
\alpha^* \leftarrow \text{arg max}_{\alpha \in [0, 1]^n} \quad \mathcal{F}(\alpha) : 1^T \alpha - \frac{1}{2} \alpha^T H \alpha
$$

s.t. \( y^T \alpha = 0 \) with \( y \in \{-1, 1\}^n \)

where \( H_{ij} = y_i y_j K(x^i, x^j) \) and \( x^k \in \mathbb{R}^d \)

and

$$
\mathcal{F}(\alpha) = \frac{1}{2} \alpha^T H \alpha + \frac{1}{2} \nu \alpha^T \alpha
$$

where \( \alpha_k > 0 \) means that \( x^k \) is a support vector for class \( y_k \). I.e. the boundary is defined as a linear combination of \( \{x^k | \alpha_k > 0\} \) in a vector space where the dot product is induced by the kernel function \( K \). SMO was originally designed to train a support vector machine that requires the solution of the very large SVM dual QP optimization problem [Platt(1998)] defined above. SMO belongs to the family of SQP algorithms [Gill et al.(2005)Gill, Murray, and Saunders]. It breaks a large QP problem into a series of smaller QP problems, each of which optimizes a quadratic model of the objective subject to a linearization of the constraints. The method is equivalent to applying Newton’s method to the Karush–Kuhn–Tucker (KKT) conditions of the QP problem. In SMO, the small QP problems involve only two variables. These small QP problems are solved analytically, thus avoiding the use of a time-consuming numerical QP optimization at each iteration. The amount of memory required for SMO is linear in the training set size \( n \), which allows SMO to handle very large input sets.

**Contributions:** In this work we show how to reformulate a QP problem using graph theory. More precisely, we relate the SVM dual QP formulation Eqn. (1) to a maximum independent set formulation, which gives a new interpretation on the support vector selection process. Then, in the same way we reduce an independent set problem to a maximum clique problem by taking the complement of a graph, we reformulate the SVM dual QP formulation as a dominant set QP [Pavan and Pelillo(2007), Zemene and Pelillo(2016)], the later being used to define the maximum weighted clique (MWC) problem [Hager et al.(Hager, Phan, and Zhang, Hager and Hungerford(2015)]. We show that our MWC formulation involves Mercer distance kernels, instead of Mercer similarity kernels, and demonstrate how to construct such kernel, leading to new families of kernels. Finally, we show that our MWC formulation has computational advantages while preserving comparable accuracy compared to the standard LIBSVM implementation of the SVM dual QP formulation.

## 2 Problem Setup And Reformulation

The independent set QP formulation has the form of

$$
\alpha^* \leftarrow \text{arg max}_{\alpha \in [0, 1]^n} \quad \mathcal{F}(\alpha) : 1^T \alpha - \frac{1}{2} \alpha^T H \alpha
$$

s.t. \( y^T \alpha = 0 \) with \( y \in \{-1, 1\}^n \)

where \( H_{ij} = y_i y_j K(x^i, x^j) \) and \( x^k \in \mathbb{R}^d \)

and

$$
\mathcal{F}(\alpha) = \frac{1}{2} \alpha^T H \alpha + \frac{1}{2} \nu \alpha^T \alpha
$$

where \( \alpha_k > 0 \) means that \( x^k \) is a support vector for class \( y_k \). I.e. the boundary is defined as a linear combination of \( \{x^k | \alpha_k > 0\} \) in a vector space where the dot product is induced by the kernel function \( K \). SMO was originally designed to train a support vector machine that requires the solution of the very large SVM dual QP optimization problem [Platt(1998)] defined above. SMO belongs to the family of SQP algorithms [Gill et al.(2005)Gill, Murray, and Saunders]. It breaks a large QP problem into a series of smaller QP problems, each of which optimizes a quadratic model of the objective subject to a linearization of the constraints. The method is equivalent to applying Newton’s method to the Karush–Kuhn–Tucker (KKT) conditions of the QP problem. In SMO, the small QP problems involve only two variables. These small QP problems are solved analytically, thus avoiding the use of a time-consuming numerical QP optimization at each iteration. The amount of memory required for SMO is linear in the training set size \( n \), which allows SMO to handle very large input sets.

**Contributions:** In this work we show how to reformulate a QP problem using graph theory. More precisely, we relate the SVM dual QP formulation Eqn. (1) to a maximum independent set formulation, which gives a new interpretation on the support vector selection process. Then, in the same way we reduce an independent set problem to a maximum clique problem by taking the complement of a graph, we reformulate the SVM dual QP formulation as a dominant set QP [Pavan and Pelillo(2007), Zemene and Pelillo(2016)], the later being used to define the maximum weighted clique (MWC) problem [Hager et al.(Hager, Phan, and Zhang, Hager and Hungerford(2015)]. We show that our MWC formulation involves Mercer distance kernels, instead of Mercer similarity kernels, and demonstrate how to construct such kernel, leading to new families of kernels. Finally, we show that our MWC formulation has computational advantages while preserving comparable accuracy compared to the standard LIBSVM implementation of the SVM dual QP formulation.
where $\tilde{K}(\mathbf{x}_i, \mathbf{x}_j)$ is now a Mercer kernel representing the distance between the two vectors $\mathbf{x}_i$ and $\mathbf{x}_j$ instead of their similarity. Here $\nu$ controls the sparsity of the solution. Next, we demonstrate how to construct a proper Mercer distance kernel.

### 2.2 Mercer Distance Kernels

First of all, we can see that if $K(\mathbf{x}_i, \mathbf{x}_j) : \mathbb{R}^n \to \mathbb{R}$ is a similarity measure, $\exists \rho, \nu \in \mathbb{R}$ such that we can build a distance metric $\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) : \mathbb{R}^n \to \mathbb{R}$ from a similarity measure using one of the following transformations:

$$\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\rho}{\nu + K(\mathbf{x}_i, \mathbf{x}_j)} \quad \text{or} \quad \tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = \nu - K(\mathbf{x}_i, \mathbf{x}_j)$$  \hfill (3)

Note that other transformation functions like $x \to e^{-\nu x^\rho}$ can be used as well. Proof and new kernel construction can be found in [Shawe-Taylor and Cristianini(2004)]. Interestingly enough, we can build a similarity measure from a distance metric the same way.

**Lemma 1.** $\exists \rho, \nu \in \mathbb{R}$ such that Eqn. (3) always produces a valid Mercer kernel.

**Proof sketch.** We base the proof on the Gershgorin circle theorem. We can find $\nu$ and $\rho$ such that:

$$\min \text{ eigenvalue} \geq \min \left( \tilde{K}(\mathbf{x}_i, \mathbf{x}_i) - \sum_{j \neq i} \tilde{K}(\mathbf{x}_i, \mathbf{x}_j) \right) \geq 0$$  \hfill (4)

However the Gershgorin bound is very loose, and solving Eqn. (4) will produce a final matrix where $|\tilde{K}(\mathbf{x}_i, \mathbf{x}_j)| \gg |\tilde{K}(\mathbf{x}_i, \mathbf{x}_j)|$. Additional knowledge on the domain of $\alpha$ helps to refine the bound. Since $\alpha, \beta \in \mathbb{R}_1^n$ we can see that $\rho > 0$ and $\nu = \max_{i,j} |K(\mathbf{x}_i, \mathbf{x}_j)| \Rightarrow \tilde{K}(\mathbf{x}_i, \mathbf{x}_i) \geq 0 \Rightarrow \forall \alpha, \beta \in \mathbb{R}_1^n, \sum_{i,j} \alpha_i \beta_j \tilde{K}(\mathbf{x}_i, \mathbf{x}_j) \geq 0$.

One may want to normalize the newly constructed distance kernel, or construct a Mercer distance kernel directly from a distance metric. Let $d(\mathbf{x}_i, \mathbf{x}_j)$ an arbitrary distance metric between the two vectors $\mathbf{x}_i$ and $\mathbf{x}_j$, and let's define:

$$\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\rho + \nu d(\mathbf{x}_i, \mathbf{x}_j)}{\nu + d(\mathbf{x}_i, \mathbf{x}_j)}$$  \hfill (5)

**Lemma 2.** $\exists \nu, \nu_2, \rho \in \mathbb{R}$ such that Eqn. (5) produces a valid Mercer kernel and $\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) \in [0, 1]$.

**Proof sketch.** Let's parametrize $\rho = \nu \nu_2 - \rho_2$. Then Eqn. (5) becomes:

$$\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = \frac{\rho + \nu_2 d(\mathbf{x}_i, \mathbf{x}_j)}{\nu + d(\mathbf{x}_i, \mathbf{x}_j)} = \frac{\nu_2 \nu - \rho_2 + \nu_2 d(\mathbf{x}_i, \mathbf{x}_j)}{\nu + d(\mathbf{x}_i, \mathbf{x}_j)}$$

$$= \nu_2 - \frac{\rho_2}{\nu + d(\mathbf{x}_i, \mathbf{x}_j)}$$  \hfill (6)

We recognize the two transformations defined in Eqn. (3). We find $\nu$ and $\rho_2$ such that $\frac{\rho_2}{\nu + d(\mathbf{x}_i, \mathbf{x}_j)}$ is a Mercer similarity kernel, then we find $\nu_2$ such that $\tilde{K}(\mathbf{x}_i, \mathbf{x}_j)$ is a Mercer distance kernel and we scale $\rho_2$ and $\nu_2$ such that $\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) \in [0, 1]$.

Note that in many cases a simpler version of Eqn. (5) can be used by setting $\rho = 0$ and $\nu_2 = 1$, allowing a grid search only with respect to $\nu$ with the kernel $\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = \frac{d(\mathbf{x}_i, \mathbf{x}_j)}{\nu + d(\mathbf{x}_i, \mathbf{x}_j)}$.

### 3 Experiments And Results

We evaluated our MWC formulation using the following datasets: “adult” (adu.), “webpage” (web.), “cod-rna” (cod.), and “splice” (spl.) from [Chang and Lin(LIBSVM Data: Binary Classification)]. The adult dataset is composed of nine partitions: a1a $\rightarrow$ a9a and the webpage dataset is composed of eight partitions: w1a $\rightarrow$ w8a. For both datasets, each partition starts with a small training set and a large testing one, and ends with a large training set and a small testing one. We run our experiments on each partition. Implementation-wise, we used the standard LIBSVM code wrapped in the OpenCv library with the RBF kernel $K_{RBF}(\mathbf{x}_i, \mathbf{x}_j) = e^{-\gamma ||\mathbf{x}_i - \mathbf{x}_j||^2}$. For our approach we adapted the SMO $\nu$-SVM algorithm of [Fan et al.(2005)]Fan, Chen, and Lin] and we used $\tilde{K}(\mathbf{x}_i, \mathbf{x}_j) = 1 - K_{RBF}(\mathbf{x}_i, \mathbf{x}_j)$, $\varepsilon = 10^{-6}$. It can be shown that even with our formulation we can still compute the bias as $b = \sum_{i,j} \alpha_i \alpha_j x_i x_j \tilde{K}(\mathbf{x}_i, \mathbf{x}_j)$, and that the decision function is $D(\mathbf{x}) = b - \sum_i \alpha_i x_i \tilde{K}(\mathbf{x}, \mathbf{x})$. While $\nu > 100$ gives similar results as LIBSVM, we are interested in taking advantage of the sparsity constraint and exploring what is the minimum number of support vectors needed to maintain
a comparable accuracy. Hence, we set $\nu = 0.1$. We initialize our system with $\leq 10$ random support vectors. We also used the same grid search ($\gamma \in \{0, 1\}, \gamma_{\text{incr.}} = 10^{-5}$) for both methods for fair comparison. Table 1 shows the baseline on the aforementioned datasets and Table 2 includes the results of our algorithm. While our solution provides comparable accuracy, it is much sparser in terms of numbers of support vectors and only requires a few iterations to converge. This leads to a training time an order of magnitude smaller than the standard LIBSVM approach.

### Table 1: Dataset specifications. The adu. and web. datasets are partitioned in increasing training set size and decreasing testing set size. The symbol "a $\rightarrow$ b" indicates the size range. The same terminology will be used for Table 2.

<table>
<thead>
<tr>
<th></th>
<th># feat.</th>
<th>train. size</th>
<th>test. size</th>
</tr>
</thead>
<tbody>
<tr>
<td>adu</td>
<td>123</td>
<td>1605 $\rightarrow$ 32561</td>
<td>30956 $\rightarrow$ 61981</td>
</tr>
<tr>
<td>web.</td>
<td>500</td>
<td>2477 $\rightarrow$ 49749</td>
<td>37272 $\rightarrow$ 13951</td>
</tr>
<tr>
<td>cod.</td>
<td>8</td>
<td>59535</td>
<td>271617</td>
</tr>
<tr>
<td>spl.</td>
<td>60</td>
<td>1000</td>
<td>2175</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>acc. LIBSVM (%)</th>
<th>adu.</th>
<th>acc. ours (%)</th>
<th>web.</th>
<th>acc. ours (%)</th>
<th>cod.</th>
<th>acc. ours (%)</th>
<th>spl.</th>
</tr>
</thead>
<tbody>
<tr>
<td>adu</td>
<td>79.5 $\pm$ 0.5</td>
<td>97.7 $\pm$ 0.3</td>
<td>66.7</td>
<td>83.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>web.</td>
<td>79.1 $\pm$ 1</td>
<td>96.9 $\pm$ 0.3</td>
<td>69.8</td>
<td>83.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>cod.</td>
<td>13.5 $\pm$ 4</td>
<td>7.75 $\pm$ 4</td>
<td>18</td>
<td>3.71</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>spl.</td>
<td>28.8 $\rightarrow$ 683.8</td>
<td>92.9 $\rightarrow$ 2343.7</td>
<td>76.6</td>
<td>22.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 2: Accuracy (acc.), number of support vectors (#SV) and training time (t.t.) for the best parameter $\gamma$ on SVM binary class datasets for LIBSVM and our approach.

### 4 Conclusion

In this work, we showed that we can reformulate QP problems using graph theory problem transformations directly. Our maximum clique formulation of SVM produces similar accuracy than the original formulation, and provides significantly smaller training time and smaller number of support vectors. The main advantage of graph theory QP reformulation is a two way street: while now we can use any MWC algorithm to train support vectors, we can use SMO-like algorithm to solve other application formulated as a MWC. This generalizes to other graph problems as well, like formulating support vector clustering as a maximum flow problem, or using weighted k-mean for spectral clustering. This also allows to extend a formulation to a more general problem: for example by replacing the product $y_i y_j$ by $2\delta(y_i, y_j) - 1$ and the constraint $y^T \alpha$ by $\sum_{y_i = c} \alpha_i = k$ in Eqn. (1) or Eqn. (2), we have now a multi-class SVM formulation. In the future, we plan to tackle other applications, like multi-class SVM, image and video segmentation and, document summarization.