Lecture 11: Support Vector Machines

Haim Sompolinsky, MCB 131, Wednesday, March 1, 2017

1 The Optimal Separating Plane

Suppose we attempt to learn rules that are linearly separable using a perceptron architecture, with a number of examples that are significantly smaller than N . In this case, finding a separating plane will not be difficult; however, the generalization performance may be problematic. This is because the volume of possible solutions is so large that it is likely that different solutions vary significantly in their performance on test examples. On the other hand, there is no guarantee that the PLA chooses a particularly good generalizer. We would therefore like to devise a learning algorithm that will yield not only a separating plane of our training data but one which is a good candidate for generalization. Strictly speaking, this is impossible to achieve because the identity and properties of a good generalizer varies from one problem to another, in a way that is not known to the learner. Nevertheless, we can use heuristic arguments which allow us to deduce which solution is a candidate for a reasonable generalization on the basis of the properties of the hyperplane with respect to the training set. This will be formulated below by the concept of *optimal hyperplane.*

Within the framework of Support Vector Machines (SVMs), the optimal hyperplane is defined as the separating plane (of a given training set) that maximizes the distance between the closest input vector to it among all hyperplanes that correctly separate the training data. Using the notation introduced in previous lecture, the optimal hyperplane is a solution with maximum margin. The geometry of the hyperplane with maximum margin, and its margin, δ, is illustrated in Fig. 1. It has two parallel planes on both sides, with equal distance from it, called the supporting planes. They which contain the examples nearest to the optimal hyperplane.

The intuition behind the choice based on the maximum margin is as follows. We expect that to be a good generalizer the solution should classify new inputs that happen to be near one of the training examples with the same label as that example. This is reasonable for rules and input distribution that enjoy some smoothness properties. While this is guaranteed for examples away from the separating plane, the extent in which this holds for inputs near the separating

plane depends on the size of the margin. Thus, maximizing the margin ensures robustness of the classification to a significant jitter in the location of all examples. The Note that SVMs is an example of *large margin* classifiers, all of which attempt to maximize the distances of the training points from the separating plane. SVM is characterized by focussing on *maximizing the minimum distance*.

The choice of optimality by SVM is further supported by the existence of a bound on the expected generalization error of for any separating plane w of some training sets, of the form

$$\langle \epsilon_g(w) \rangle \leq \frac{1}{P} \langle \frac{X^2}{\delta^2(w)} \rangle \tag{1}$$

where X is the maximum norm of the minimum margin vector(s) and the averaging is over all possible sampling of training data.

We will now recapitulate the above discussion, by a a formal definition of the optimal hyperplane. For $\vec{W}$ to be the optimal hyperplane, it must obey two requirements. First,

$$y_0^\mu \vec{W} \cdot \vec{x}^\mu > 0, \forall \mu 1...P \tag{2}$$

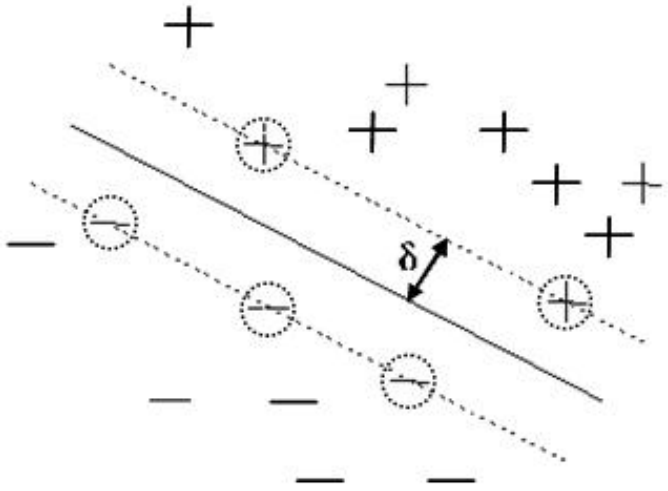
i.e., the plane correctly separates the training data. Second,

$$\vec{W} = \arg\max_{\vec{W}} \delta(\vec{W}) \tag{3}$$

where,

$$\delta(\vec{W}) = \min_\mu \delta^\mu(\vec{W}) \tag{4}$$

$$\delta^\mu = \frac{y_0^\mu \vec{W} \cdot \vec{x}^\mu}{||\vec{W}||} > 0 \tag{5}$$



1.1 *Uniqueness of the optimal hyperplane

Theorem: *The hyperplane with maximum margin is unique* (Vapnik, 1998)

Proof: For each separating plane define

$$\rho(\vec{W}) \equiv min_\mu y_0^\mu \vec{W} \cdot x^\mu$$

(this is not the margin because of the lack of division by the size of $\vec{W}$). Two simple but useful properties are that ρ is continuous and that $\rho(c\vec{W}) = c\rho(\vec{W})$ for every $c > 0$. Now, we are concerned with hyperplanes, so the size of $\vec{W}$ does not matter, just its orientation (the normalized margin cancels out the size of the hyperplane). So let us look at the set of vectors $\vec{W}$ with norm $\left|\vec{W}\right| \leq 1$. ρ is continuous, and this set is compact, so ρ attains a maximum at some point there. Can this be an interior point (i.e. $\left|\vec{W}\right| < 1$)? No, because then $\rho\left(\frac{\vec{W}}{|\vec{W}|}\right) = \frac{1}{|\vec{W}|}\rho\left(\vec{W}\right) > \rho\left(\vec{W}\right)$ – in other words, this cannot be the maximum because $\frac{\vec{W}}{|\vec{W}|}$ has a larger value of ρ. Thus the maximum is attained on the sphere $\left|\vec{W}\right| = 1$. This is useful because on the sphere, ρ is of course equal to the normalized margin, so the maximal point(s) of ρ on the sphere are precisely those of the true, normalized margin.

We must now prove that the maximum can only be attained at one point. To do this, we first note that ρ is concave, since $min_\mu(\alpha y_0^\mu \vec{W}^1 \cdot x^\mu + (1-\alpha) y_0^\mu \vec{W}^2 \cdot x^\mu \geq$

$\alpha min_\mu y_0^\mu \vec{W^1}\cdot x^\mu + (1-\alpha) min_\mu y_0^\mu \vec{W^2}\cdot x^\mu)$ Now, assume that there are two points on the sphere on which the maximum is attained. By the concavity of ρ, there must be a point on the line between them – which lies inside the sphere – on which ρ is also maximal. But, as we saw before, this means that there is a maximum in the interior of the sphere which is impossible, as we have shown above. Thus, there can be only a single point on the sphere on which the maximum of ρ is attained, and this is (as mentioned before) the orientation of the hyperplane with maximal margin, thus proving the theorem.

Indeed, in worst cases, the convergence time of the PLA can be in worst cases can be extremely long since there might be two examples with opposite lables but very close to each othe which will make δ_{max}very small. However, the typical behavior can be evaluated from the following scenario. Imagine the inputs are generated from a gaussian distribution where each component is $\langle x_i^\mu \rangle = 0$ and variance $1/N$. so that the norm of the inputs is 1 on average. Then, the probaility of a random example to have a small (relative to σ) margin δ^μ is roughly $\delta^\mu/\sigma = \delta^\mu\sqrt{N}$. Thus, the probability that there will be one example with margin δ is $P(\delta) \approx P\delta\sqrt{N}$. So the minimal margin is given by the condition $P(\delta) = 1$, from which we obtain

$$\delta \approx \frac{1}{P\sqrt{N}},\ n \approx P^2 N \tag{6}$$

2 Computing the optimal hyperplane: The Primal Problem

To find the optimal $\vec{W}$ it is useful first, to get rid of the denominators $||\vec{W}||$ in the definition of the margins 5, since it is cumbersome to deal with such a nonlinear operation. To do this we note that we can always multiply a weight vector by a positive constant without changing how it separates the training data, nor its margins (since the distances contain a normalization by the size of the weight vector). Thus, for each separating plane, we can choose its norm to be $1/\delta$, i.e.,

$$\delta(\vec{W}) = \frac{1}{||\vec{W}||} \tag{7}$$

With this normalization, the examples with the minimal margin obey,

$$\delta = \frac{y_0^\mu w^T x^\mu}{||w||} = \delta y_0^\mu w^T x^\mu \tag{8}$$

so that

$$y_0^\mu \vec{W} \cdot \vec{x}^\mu = 1 \tag{9}$$

and for all examples 2 becomes $y_0^\mu \vec{W} \cdot \vec{x}^\mu = \delta^\mu / \delta$,

$$y_0^\mu \vec{W} \cdot \vec{x}^\mu \geq 1, \forall \mu \tag{10}$$

The closest examples obey of course the equality relation. Thus we can replace the condition on the optimal plane by requiring that the corresponding weight vector obeys:

$$\vec{W} = \arg\max_{\vec{W}} \left\{ \frac{1}{||\vec{W}||} \mid y_0^\mu \vec{W} \cdot \vec{x}^\mu \geq 1, \forall \mu \right\}. \tag{11}$$

Thus, finding the optimal plane is equivalent to solving the following optimization problem:

$$w^{opt} = \arg\min_{w} \left\{ w^T w \mid y_0^\mu \vec{W} \cdot \vec{x}^\mu \geq 1, \forall \mu \right\}. \tag{12}$$

3 The Dual Problem

The dual formulation is based on **Kůhn-Tucker Theorem**. KT Theorem generalizes the method of Lagrange multipliers-which deals with optimization with equality constraints, to solving optimization problems with inequality constraints, as in our case. Further, we will utilize the fact that ours is a problem of convex optimization (where both the function to be optimized is convex and the set of constraints are convex).

The K-T theorem relates the solution to 12 to the following Lagrangian,

$$L(w, \alpha) = \frac{1}{2} w^T w + \sum_{\mu=1}^{P} \alpha_\mu (1 - y_0^\mu \vec{W} \cdot \vec{x}^\mu)$$

In this equation, w is N dimensional as before,α is a P dimensional vector whose components α_μ are the Lagrange multipliers associated with each of the inequality constraints. These auxiliary variables are constrained to be non-negative which we will denote as $\alpha \geq 0$. Now, let us fix w and maximize L wrt to α

$$L_{max}(w) = \max_{\alpha \geq 0} L(w, \alpha) \tag{13}$$

It is easy to see that for all w that violates the constraints in our primal problem, $L_{max} = \infty$ (because we can choose the corresponding αs to be ∞. On the other hand for w that obeys all the constraint, maximizing L will necessarily mean that for all μthat are obeyed as strict inequalities, α^{μ}should vanish. For the inequalities that are obeyed as equalities, namely they are not active, αcan be positive, i.e.,

$$y_0^{\mu} w^T x^{\mu} \geq 1, \forall \mu \tag{14}$$

$$\alpha^{\mu} \geq 0, \forall \mu \tag{15}$$

$$\alpha_{\mu}(1 - y_0^{\mu} w^T x^{\mu}) = 0, \forall \mu \tag{16}$$

Thus, for all w that obeys the constraint $L_{max}(w) = w^T w$. Hence, we can conclude that the optimal w obeys the following property

$$w^{opt} = \arg \left\{ \min_{w} \max_{\alpha \geq 0} L(w, \alpha) \right\} \tag{17}$$

Consider now the dual operation. Fix $\alpha \geq 0$ and evaluate

$$L_{min}(\alpha) = \min_{w} L(w, \alpha) \tag{18}$$

By differentiating wrt w and solving for $\partial L / \partial w = 0$, this will yield,

$$w = \sum_{\mu=1}^{P} \alpha^{\mu} y_0^{\mu} x^{\mu} \tag{19}$$

Now solve the dual problem,

$$\alpha_0 = \arg \left\{ \max_{\alpha \geq 0} \min_{w} L(w, \alpha) \right\} \tag{20}$$

By substituing 19 back into L we obtain, that the dual problem is equivalent to:

$$\alpha_0 = \arg \left\{ \max_{\alpha \geq 0} Q(\alpha) \right\} \tag{21}$$

$$Q(\vec{\alpha}) = \frac{1}{2} \left\| \sum_{\mu=1}^{p} \alpha_{\mu} y_0^{\mu} x^{\mu} \right\|^2 + \sum_{\mu=1}^{p} \alpha_{\mu} (1 - y_0^{\mu} x^{\mu T} \sum_{\nu=1}^{p} \alpha_{\nu} y_0^{\nu} x^{\nu}) \tag{22}$$

and thus

$$Q(\alpha) = \sum_{\mu=1}^{p} \alpha_\mu - \frac{1}{2} \sum_{\mu,\nu=1}^{p} \alpha_\mu \alpha_\nu y_0^\mu y_0^\nu x^{\mu T} x^\nu \tag{23}$$

Do these two calculation coincide?

It turns out for problems like ours (where both the objective function and the constraints are convex), the outcome of the primal and dual operations is the same. Specifically, for the convex case, KT theory states

1. There is a set $\alpha_0 \geq 0$ such that together with w^{opt}obey,

$$L(w^{opt}, \alpha_0) = \min_w \max_{\alpha \geq 0} L(w, \alpha) = \max_{\alpha \geq 0} \min_w L(w, \alpha) \tag{24}$$

2. The solution obeys the following $Kuhn - Tucker$ conditions:

$$w = \sum_{\mu=1}^{P} \alpha^\mu y_0^\mu x^\mu \tag{25}$$

$$y_0^\mu w^T x^\mu \geq 1, \forall \mu \tag{26}$$

$$\alpha^\mu \geq 0, \forall \mu \tag{27}$$

$$\alpha_\mu (1 - y_0^\mu w^T x^\mu) = 0, \forall \mu \tag{28}$$

3. The KT conditions are also sufficient conditions for yielding an optimal w. The importance of the dual formulation is first it provides a simpler objective function, $Q(\alpha)$, 23, since the inequalities $\alpha \geq 0$ have simpler form than the constraints on w . Once α is found, w^{opt} is calculated via 19 . Furthermore, the associated KT conditions reveal the structure of w^{opt}as we will discuss below.

4 The Support Vectors

From 19 we learn that the optimal weight vector can be expressed as a linear combination of the input vectors that appear in the training set, and this holds even when $P < N$. This is reasonable since adding a component orthogonal to them will increase its magnitude without affecting any of the constraints. Furthermore, the optimal weight vector is specified by only a subset of the examples, which have the minimal margin with respect to it, i.e., they lie on the supporting hyperplanes . This makes sense, since it is clear from the geometry of the problem that changing slightly the positions of the other vectors is not

going to change the location of the optimal hyperplane. The vectors with *active* constraints, namely those for which $\alpha_\mu > 0$ are called the support vectors.

These results have important consequences for the generalization problem. First, the solution is specified by a number of parameters that is bounded by the number of examples, P, and not by the number of components, N. This is important when P is much less than N. Furthermore, in many problems the number of support vectors is substantially smaller than P which means that the solution has a sparse representation in terms of the support vectors, once again, a good sign for generalization. This is expressed by the following bound on the generalization error of the optimal plane,

$$\langle \epsilon_g(P-1) \rangle \leq \frac{\langle K(P) \rangle}{P} \tag{29}$$

where K is the number of *essential* support vectors of the training set (essential support vecors are those that appear in all possible expansion of w^{opt}) and the average is over all samplings of the P training sets.

5 The Role of Dot-Products

A by-product of the SVM is the fact that if we focus on the optimal hyperplane we can characterize the entire operation of the system by dot products and support vector coefficients without explicit reference to the w itself. To see this note that substituting Eq.19 into $w^T x$, where x is any vector in R^N, we can write the output of the system after learning, for any for any input vector x as

$$y(x) = sign \left\{ \sum_{\mu=1}^{P} \alpha^\mu y_0^\mu x^{\mu T} x \right\} \tag{30}$$

Similarly, for learning all we need is to compute the PxP matrix

$$K_{\mu,\nu} = x^{\mu T} x^\nu \tag{31}$$

This will be the basis of an important generalization of the SVM which will be discussed in the next lecture.

5.1 *The Adatron Algorithm

Here we describe a simple learning algorithm for solving the Dual Problem, called Adaptive Perceptron (Adatron). As we will show, the Adatron algorithm asymptotically approaches the global maximum of Q. While this is not the most efficient algorithm for solving this problem (better ones exist), it is the simplest.

Like the perceptron learning algorithm, this is an incremental learning method that looks at the training examples one by one, and presents them cyclically until an entire cycle is completed without any alterations made. However, unlike the Perceptron Learning, Adatron does not act directly on the components of the weight vector but acts on the parameters $\vec{\alpha}$. In fact, as we will see, in Adatron, at each step, we receive an example, say ν and update only the estimate of the corresponding coefficient α_ν.

The algorithm is as follows:

1) Begin with the values $\left\{\alpha_\mu^0 = 0\right\}_{\mu=1}^P$ (with some initial guess of P).

2) Present examples one by one. At each step $n+1$ an example ν is presented. Update the value of α_ν, according to the following rule:

$$\Delta\alpha_\nu^n = \alpha_\nu^{n+1} - \alpha_\nu^n = \max\left\{-\alpha_\nu^n, \eta(1-h_\nu^n)\right\} \tag{32}$$

where $h_\nu^n \equiv y_0^\nu \vec{W}^n \cdot \vec{x}^\nu$ is the total signed input to the decision neuron. These quantities are evaluated by substitution of **??**, yielding

$$h_\nu^n = \sum_\mu y_0^\nu y_0^\mu \vec{x}^\mu \cdot \vec{x}^\nu \alpha_\mu^n \tag{33}$$

Adatron Theorem: For $0 < \eta < \frac{2}{D}$ [where $D = \max_\mu ||\vec{x}^\mu||$], Adatron converges asymptotically (i.e., in the limit $n \to \infty$) to a set $\{\alpha_\mu\}$ that solves the Dual Problem.

Proof of convergence:

First, note that $\alpha_\mu^n \geq 0, \forall n, \mu$, since at each step we add at least enough to keep things equal to zero or higher. Second, let us compute the change in the value of Q, **??** at each step. To do this, we will consider $\Delta Q^n \equiv Q^{n+1} - Q^n$:

$$\begin{aligned}
\Delta Q^n &= \Delta\alpha_\nu^n - \Delta\alpha_\nu^n(\sum_{\mu=1}^P \alpha_\mu y_0^\nu y_0^\mu \vec{x}^\nu \cdot \vec{x}^\mu) - \frac{1}{2}(\Delta\alpha_\nu^n)^2 \left\|\vec{x}^\nu\right\|^2 &(34)\\
&= \Delta\alpha_\nu^n - \Delta\alpha_\nu^n h_\nu^n - \frac{1}{2}(\Delta\alpha_\nu^n)^2 \left\|\vec{x}^\nu\right\|^2 &(35)\\
&= \Delta\alpha_\nu^n(1 - h_\nu^n - \frac{1}{2}(\Delta\alpha_\nu^n) \left\|\vec{x}^\nu\right\|^2) &(36)
\end{aligned}$$

Our goal is to show that this is positive. We will now consider the possible cases.

(1) Assume that $1 - h_\nu^n > 0$. Then $\Delta\alpha_\nu^n = \eta\,(1 - h_\nu^n)$ (since we change by the maximum of this value and something non-positive, and this itself is positive). In this case

$$\Delta Q^n = \Delta\alpha_\nu^n \left(\frac{1}{\eta}\Delta\alpha_\nu^n - \frac{1}{2}\Delta\alpha_\nu^n \left\|\vec{x}^\nu\right\|^2\right) = \frac{(\Delta\alpha_\nu^n)^2}{\eta}(1 - \frac{1}{2}\eta \left\|\vec{x}^\nu\right\|^2)$$

It is now clear that if we choose η: $\eta < \frac{2}{\max_{\mu} \|\vec{x}^{\mu}\|}$, then $\Delta Q^n > 0$.

(2) Assume that $1 - h_{\nu}^{n} \leq 0$. Thus $\Delta\alpha_{\nu}^{n} \leq 0$. Recall that it is always true that $\Delta\alpha_{\nu}^{n} \geq \eta\,(1 - h_{\nu}^{n})$. Multiplying this by $\Delta\alpha_{\nu}^{n} \leq 0$, we get

$\Delta\alpha_{\nu}^{n}\,(1 - h_{\nu}^{n}) \geq \frac{(\Delta\alpha_{\nu}^{n})^2}{\eta}$ and thus,

$$\Delta Q^n \geq \frac{(\Delta\alpha_{\nu}^{n})^2}{\eta} - \frac{1}{2}(\Delta\alpha_{\nu}^{n})^2 \left\|\vec{x}^{\nu}\right\|^2 = \frac{(\Delta\alpha_{\nu}^{n})^2}{\eta}(1 - \frac{1}{2}\eta \left\|\vec{x}^{\nu}\right\|^2)$$

Like in the previous case, if $\eta < \frac{2}{\max_{\mu} \|\vec{x}^{\mu}\|}$, then $\Delta Q^n \geq 0$.

We have shown that the Adatron algorithm increases Q over time. Since Q is clearly bounded from above (Exercise: Explain why we know that Q is bounded from above), and increases at each step, the algorithm much reach a fixed point at long time. Substituting $\Delta\alpha_{\nu} = 0$ for all ν into the algorithm results in:

(1) All examples must obey $1 - h_{\nu}^{n} \leq 0$ otherwise there will be necessarily an update.

(2) Those with $1 - h_{\nu}^{n} < 0$ must have $\alpha_{\mu} = 0$ so there is no update also in this case.

Together with **??**, this shows that the fixed point obeys K-T conditions, hence it is the optimal weight vector.

More directly, assume we reached the fixed point of the Learning Algorithm, denote it by α^*. Assume that this is not the maximum of Q (under the constraint $\alpha \vec{\geq} 0$. Then, since Q is a convex function it does not have local maxima. Thus, the only possibility is that at α^*there are some allowed directions which will still increase Q. To see whether this is possible, let us compute the gradient of Q, which can be easily computed,

$$\frac{\partial Q}{\partial \alpha^{\mu}} = 1 - h_{\mu}$$

Since we have shown that at the fixed point, these quantities are non-positive, the only possiblity we have to consider is the case where

$1 - h_{\mu} < 0$. Increasing Q along this direction requires *decreasing* the corresponding α_{μ}. However, we have shown that at the fixed point these components are zero, hence we are not allowed to decrease its values. Hence α^*must be a maximum of Q .

Lecture 11: Non-Linear SVMs and The Kernel Method

Haim Sompolinsky, MCB 131, March 4, 2015

1 Non-linearly separable data: The Feature Space

We now consider the second feature of Support Vector Machines, which is their ability to handle non-linearly separable data – which is a common occurance in real-world problems. The basic idea is to preprocess the training data by mapping it into a higher dimension. The lower dimensional inputs become feature vectors in a higher dimension. If the dimensionality of the feature vector space, is large enough then linear separation should be possible.

More formally, let the original training samples be $\{\vec{x}^{\mu}\}_{\mu=1}^{P}$. Let $\Phi(x)$ be a transformation from the original space of the training samples to another, $\Phi : \mathbb{R}^N \rightarrow \mathbb{R}^M$, defined by $\Phi(\vec{x}) = (\Phi_1(\vec{x}), ..., \Phi_M(\vec{x}))$. In the SVM the network operates as a perceptron in the new space, characterized by an M dim weight vector $\vec{W}$, and the final output of the network is $y(\vec{x}) = sign(\vec{W} \cdot \Phi(\vec{x}))$. A graphic illustration of this architecture is shown in Figure 1.

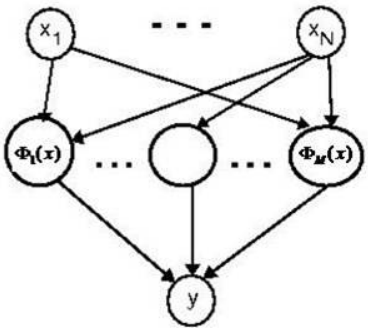


Figure 1: Embedding Input in a High Dimensional Feature Space

Typically we will choose $M \gg N$, so that the training examples will be separable in the very high-dimensional space that Φ maps to, due to Cover's theorem (as long as M is larger than the size of the training set, P). It must be noted that not all choices of Φ will work. In particular, linear functions are not useful as they map the inputs into an N dimensional hyperplane embedded in the M dim space, so this will not change their separability properties. In general, Φ must be a nonlinear function that embeds the P N-dim vectors into points which are in general position in R^M. The example feature vectors that will be mentioned later all have the desired property.

The idea that preprocessing inputs by mapping them into a high dimensional space, renders them more easily separable (and in particular, separable by a linear classifier) is one of the oldest ideas in computational neuroscience. As we will see in a later section, Marr's theory of the Cerebellum, discussed later suggests that the first stages of Cerebellar circuitry do precisely this operation. More generally, as shown in section (Introduction to Neural Representations), many brain systems exhibits expansion of representation, and this might be partly their function.

Although the use of the above feature space will make the examples linearly separable, implementing it, we are faced with two problems:

1. Generalization: Mapping to a larger space, generalization error may suffer, because M is large in relation to P.

2. **Computation:** Working in high dimension substantially increases the computational costs. Also, as we will see, some useful feature spaces are of infinite dimensions, so we need an alternative compact architecture to represent this computation.

SVMs offer the following solutions to these two problems:

Generalization: The weights for the second layer should be chosen not simply by demanding that w separates the examples, but that it is the optimal hyperplane in the feature space. This ensures that w has a 'compact' representation in terms of at most P support vectors (regardless of M) and is a good candidate for yielding a good generalization.

Computation: To solve this problem, the SVMs uses the *Kernel* method, which implements the same computation as the system of Fig. 1 but with a more compact architecture.

2 The Kernel Method

We assume that in the large feature space we choose W to be the optimal hyperplane. Therefore, the output of the network is $y = sign(\vec{W} \cdot \Phi(\vec{x}))$, where

$$\vec{W} = \sum_{\mu=1}^{P} \alpha_\mu y_0^\mu \Phi(\vec{x}^\mu) \tag{1}$$

Thus what appears inside the *sign* operation is $\vec{W} \cdot \Phi(\vec{x}) = \sum_{\mu=1}^{P} \alpha_\mu y_0^\mu \Phi(\vec{x}) \cdot \Phi(\vec{x}^\mu)$. Note that the only way that Φ appears here is as part of an inner product with another Φ. Let us therefore define

$$K(\vec{x}, \vec{y}) = \Phi(\vec{x}) \cdot \Phi(\vec{y}) \tag{2}$$

Here $\vec{x}, \vec{y}$ are of dimension N, and $\Phi(\vec{x}), \Phi(\vec{y})$ are of dimension M, as before. Thus the output of the network is

$$y(\vec{x}) = sign\left(\sum_{\mu=1}^{P} \alpha_\mu y_0^\mu K(\vec{x}, \vec{x}^\mu)\right) \tag{3}$$

The function K is called a *Kernel* function. Note that K is a transformation from two vectors of size N to a scalar value – the large space of dimension M is not a problem here, in the sense that if we know the form of the function $K(\cdot, \cdot)$ then we don't need to perform operations in the large space of dimension M!

What about the learning stage? As is seen in 3 all we need to know is the the values of α_μ . Can we evaluate those without working directly in the feature space? The answer is positive if we use the dual method. The reason is that the dual objective function has the form

$$Q(\vec{\alpha}) \quad = \quad \sum_{\mu=1}^{P} \alpha_\mu - \frac{1}{2} \sum_{\mu,\nu=1}^{P} \alpha_\mu \alpha_\nu y_0^\mu y_0^\nu \underbrace{\Phi(\vec{x}^\mu) \cdot \Phi(\vec{x}^\nu)}_{K(\vec{x}, \vec{x}^\mu)}, \ \alpha \geq 0 \tag{4}$$

Once again, the Kernel is all we need, since only dot products of pairs training feature vectors appear in Q . So, any method of computing $\alpha$from maximizing Q can be used simply by replacing the dot products of the linear separability case, $\vec{x}^\nu \cdot \vec{x}^\mu$, by the general matrix constructed from $K(\vec{x}, \vec{x}^\mu)$. When doing so, we will in effect find a hyperplane in an M-dimensional space without having to perform the numerical calculations in that space.

Graphically, the Kernel method replaces the original feature vector architecture of Figure 1 with that of Figure 2. Notice how the size of the middle layer is now only P and not M.

2.1 Example: Polynomial Kernel

Consider the following transformation:

$$\Phi(\vec{x}) = (x_1, x_2, ..., x_N \ , \ x_1x_1, x_1x_2, ..., x_1x_N \ , \ ... \ , \ x_Nx_1, ..., x_Nx_N)$$

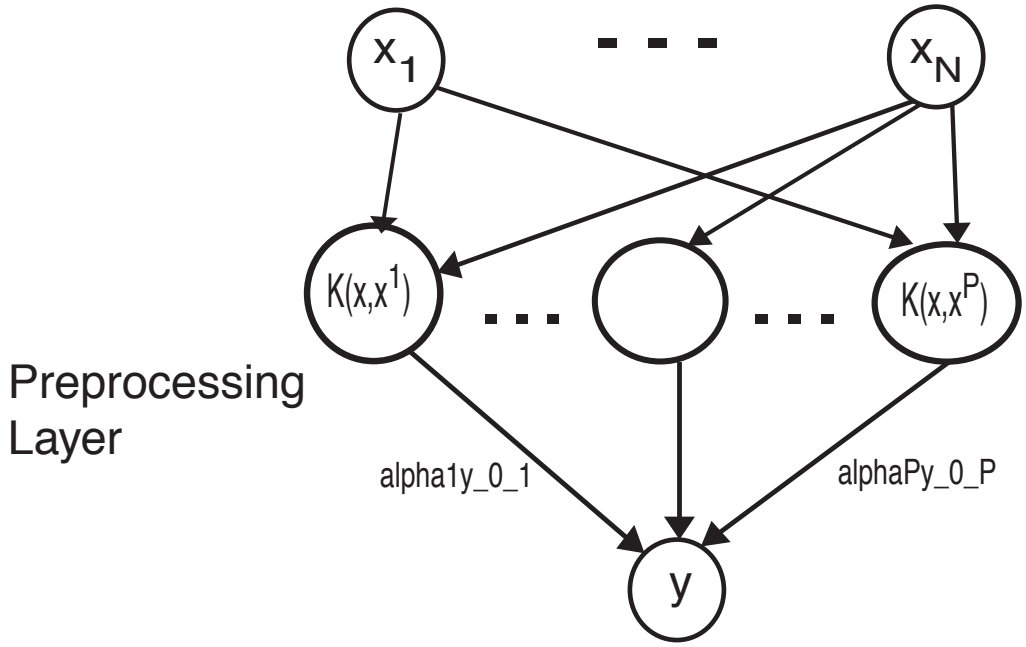


Figure 2: Kernel Architecture

In other words, Φ copies the co-ordinates of the vector, and then adds the multiplications of every two co-ordinates. Thus the dimension of the space we map to is $M = N + N^2$.

The scalar multiplication of two transformed vectors is

$$\Phi(\vec{x}) \cdot \Phi(\vec{y}) = \sum_i x_i y_i + \sum_{i,j} x_i x_j y_i y_j = \vec{x} \cdot \vec{y} + (\vec{x} \cdot \vec{y})^2$$

Thus we can write the Kernel function $K(\vec{x}, \vec{y}) = \Phi(\vec{x}) \cdot \Phi(\vec{y})$ as

$$K(\vec{x}, \vec{y}) = \left(\vec{x} \cdot \vec{y} + \frac{1}{2}\right)^2 - \frac{1}{4}$$

Such a Kernel is called a Polynomial Kernel of degree 2. By changing the exponent from 2 to something higher we get Polynomial Kernels of higher degrees, such as

$$K(\vec{x}, \vec{y}) = (\vec{x} \cdot \vec{y} + 1)^d$$

2.2 Mercer's Theorem

As we have seen, the Kernel method provide us with a compact scheme of computing with non-linear SVMs without explicitly computing with feature space units. This suggests that our *starting point* may be the network with the Kernel architecture of Figure 2. We will then use $Q(\alpha)$ to learn the output weights and the input arguments of the Kernels. This raises the question can any Kernel be used with this learning scheme? Will any kernel lead to a convergence of the dual problem to a solution for α?

We can provide the following sufficiency conditions for the Kernel: As long as the kernel function can be expanded in a basis of feature functions, we can use SVM (dual) learning . The reason is straightforward. If such expansion exists we can write down the same architecture in the feature space and use linear SVM learning, which as shown above is completely equivalent to the Kernel method. Thus, we need to know for which Kernels there exists a vector function $\Phi(x)$ (possibly of infinite dimensionality) such that $K(\vec{x}, \vec{y}) = \Phi(\vec{x}) \cdot \Phi(\vec{y})$? The answer is given by

Mercer's Theorem: For any Kernel function, $K(x, y)$, there exists a mapping from x to a set of real scalar functions (generally infinite number of them, i.e., they form a Hilbert Space) ϕ_l such that the kernel K can be written as a dot product,$K(\vec{x}, \vec{y}) = \Phi(\vec{x}) \cdot \Phi(\vec{y})$, iff;

(1) $K(\vec{x}, \vec{y}) = K(\vec{y}, \vec{x}), \forall \vec{x}, \vec{y}$ (symmetry)

(2) K is positive semi-definite, i.e.: $\int \int f(\vec{x}) K(\vec{x}, \vec{y}) f(\vec{y}) d\vec{x} d\vec{y} \geq 0$ for every function f for which $\int f(\vec{x})^2 d\vec{x} < \infty$.

Note 1: As we will see below, a kernel that meets Mercer's condition can be expanded in terms of orthogonal sets of ϕ s.

Note 2: Mercer's theorem can be generalized to kernels which are functions of pairs of elements in some general sets other than vectors in R^N [e.g., Watkins (200). This has greatly extended the use of Kernels.

Proof (qualitative): Mercer's theorem is based on extending the properties of symmetric matrices in finite dimensions, to the infinite dimensional case. Thus, in analogy to symmetric matrices, symmetric Kernels have a countable orthogonal basis of eigenvectors with real and positive eigenvalues, hence:

$$K(x,y) = \sum_l u_l(x)u_l(y)k_l = \sum_l \phi_l(x)\phi_l(y) \tag{5}$$

$$K(x,y) = \sum_l u_l(x)u_l(y)k_l \tag{6}$$

where k_l real and are the spectra of K analogous to eigenvalues of finite dim matrices. Furthermore, if the kernel is positive semi definite, then all $k_l \geq 0$. Hence, one can write

$$K(x,y) = \sum_l \phi_l(x)\phi_l(y) \tag{7}$$

with,

$$\phi_l(x) = \sqrt{k_l}u_l(x) \tag{8}$$

The converse part of the theorem is straightforward (why?).

An example with infinite dimensional feature space: Consider $1D$ input space and a kernel of the form,

$$K(x,y) = e^{xy} \tag{9}$$

This corresponds to the infinite-dimensional Φspace:

$$\Phi(x) = (1, x, \frac{x^2}{\sqrt{2!}}, \frac{x^3}{\sqrt{3!}}, ...)$$

since,

$$K(x,z) = \Phi(x)\cdot\Phi(z) = \sum_{k=0}^{\infty}\frac{(xz)^k}{k!} = e^{xz}$$

Hence the quite innocent Kernel allows us to perform operations in an infinite-dimensional space.

Admissible Kernels:

1. Sums of admissible kernels is an admissible kernel.

2. Products (either element-wise or outer products) of admissible kernels is admissible.

3. Dot product kernels: iff its Taylor series has non-negative coefficients.

Examples: Polynomial kernels

3. Translational invariant kernels: iff their Fourier transform is positive.

A commonly-used Kernel is the Radial Basis Function (RBF) Kernel,

$$K(\vec{x}, \vec{z}) = \exp\left(-\frac{||\vec{x} - \vec{z}||^2}{2\sigma^2}\right)$$

which can be shown to fulfill the conditions of Mercer's Theorem, and to also correspond to a Φ that maps into an infinite-dimensional space.

4. Logistic function

$$K(x \cdot y) = \frac{1}{1 + \exp(\beta x \cdot y - \theta)}$$

in general not. Only in restricted cases: For instance if all x and y have norm 1 and $\beta \leq \theta$, in which case one can transform it into a Radial Basis form.

Take home message: The importance of the Kernel method that the effective dimensionality, i.e., the number of modifiable parameters is set by the number of available examples and not by the dimensionality of the feature basis for the Kernels, which in fact maybe infinite.

An example of this point is in the following table (from Vapnik's book) regarding application of SVM to the benchmark problem of handwritten character recognition. What is remarkable is that despite the enormous growth in the dimensionality of the feature space for polynomial kernels, the number of support vectors increases very modestly.

Bibliography:

1. The best source is the book by one of the main 'inventors' of SVMs: Vladimir N. Vapnik: Statistical Learning Theory.

2. There are several good tutorials or reviews on SVMs. One (posted on the course website) is:

Christopher Burges: A Tutorial on Support Vector Machines for Pattern Recognition.

Table 12.7. Results of experiments with polynomials of the different degrees

Degree of Polynomial	Dimensionality of Feature space	Support Vectors	Raw Error
1	256	282	8.9
2	$\sim 33,000$	227	4.7
3	$\sim 1 \times 10^6$	274	4.0
4	$\sim 1 \times 10^9$	321	4.2
5	$\sim 1 \times 10^{12}$	374	4.3
6	$\sim 1 \times 10^{14}$	377	4.5
7	$\sim 1 \times 10^{16}$	422	4.5

Figure 3: Polynomial Kernels and the number of support vectors