Machine Learning SVM Report

Hanwen Liu 522030910109

2024 年 6 月 7 日

1 Algorithm

1.1 Classic SVM Problem

The SVM algorithm is invented to solve the classification problem with linear classifier. The classification can be described as: Given N training data points:

$$x_1, ..., x_i, ..., x_N \in R^m$$

each with a label of

$$d_i = +1 \; or \; d_i = -1$$

find a hyperplane

$$w^T x + b = 0 \; (w \in R^m, b \in R)$$

which separates data into two groups:

$$\{x_i | d_i = +1\} \; and \; \{x_j | d_j = -1\}$$

As shown in figure 1, define the support vector as x_p and x_n and the vector on the hyperplane as x_0, we have:

$$\begin{cases} w^T x_p + b = 1 \\ w^T x_0 + b = 0 \\ x_p = x_0 + \frac{w}{||w||} \dot{\gamma} \end{cases}$$

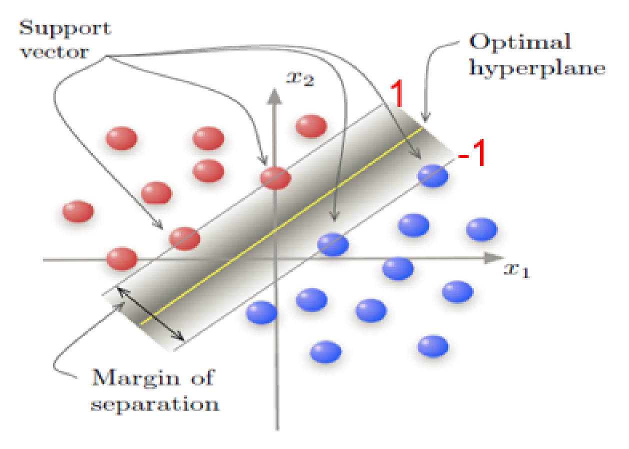

图 1: SVM

We can derive from the above equations that the distance form the support vector to the hyperplane:

$$\gamma = \frac{1}{||w||}$$

Since the goal of SVM is to maximize the margin of separation and the margin of separation $\rho \propto \gamma$. Hence, the SVM problem can be formulated as:

$$\begin{aligned} & min \; \frac{1}{2} w^T w \\ & s.t. \; d_i(x_i w + b) \geq 1 \; i = 1, 2, ..., m \end{aligned}$$

It's Lagrangian function is:

$$L : \frac{1}{2}w^T w - \sum_{i=1}^{N} \alpha_i [d_i(x_i^T w + b) - 1]$$

From the KKT condition, we can obtain its dual problem:

$$max : \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j x_i^T x_j$$
$$s.t. \quad \alpha_i \geq 0, \ \sum_{i=1}^{N} \alpha_i d_i = 0$$

1.2 Slack Variable

In various situation, the ideal hyperplane may not exist since the data is not linearly separable. Hence, slack variable ξ is introduced to optimize the original SVM problem as shown in figure 2, Then the original problem can be written as:

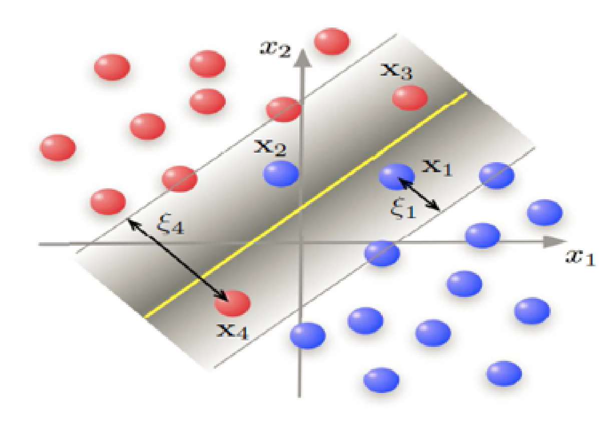

图 2: Slack Variable in SVM

$$min \ \frac{1}{2}w^T w + C\sum_{i=1}^{N}\xi_i$$
$$s.t. \ d_i(w^T x_i + b) \geq 1 - xi_i$$
$$xi_i \geq 0$$

Applying the Lagrangian function and KKT condition, we can obtain its dual problem

$$max : \sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j x_i^T x_j$$
$$s.t. \ \sum_{i=1}^{N} \alpha_i d_i = 0$$
$$0 \leq \alpha_i \leq C$$

where C is set by user to control the trade-off between a large margin and a small error penalty.

1.3 Sequential Minimal Optimization

The SMO algorithm is introduced to solve the SVM dual problem. The dual problem is to maximize

$$\sum_{i=1}^{N} \alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j x_i^T x_j$$

The aim is to obtain each α_i to maximize the goal. The idea is that: iteratively select α_i, α_j and fix other α to optimize α_i, α_j until no optimization can be applied to any α.
Before the explanation, we should introduce kernel first. As shown in figure 3, sometimes nonlineary-separable patterns may be transformed into a new feature space in which they are linearly seperable.

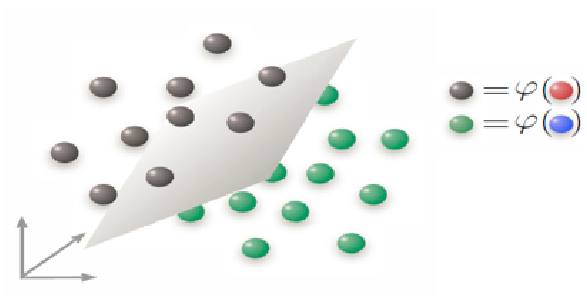

图 3: Increase Dimension

There we introduce the transformation ϕ. Then the dual probelm can be expressed as

$$\begin{aligned} max: & \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j d_i d_j \phi^T(x_i)\phi(x_j) \\ s.t. & \sum_{i=1}^{N}\alpha_i d_i = 0 \\ & 0 \le \alpha_i \le C \end{aligned}$$

Kernel is introduced here so that we can combine $\phi^T(x_i), \phi(x_j)$

$$K(x_i,x_j) = K(x_j,x_i) = \phi^T(x_i)\phi(x_j) = \phi^T(x_j)\phi(x_i)$$

The frequently used kernel includes linear kernel, polynomial kernel, gaussian kernel and sigmoid kernel.
Now suppose we choose α_1 and α_2 to optimize and fix other α. Then the problem can be formulated as

$$\begin{aligned} max\ W(\alpha_1,\alpha_2) = & \alpha_1 + \alpha_2 - \frac{1}{2}K_{1,1}d_1^2\alpha_1^2 - \frac{1}{2}K_{2,2}d_2^2\alpha_2^2 \\ & - K_{1,2}d_1d_2\alpha_1\alpha_2 - d_1\alpha_1\sum_{i=3}^{N}\alpha_i d_i K_{i,1} \\ & - d_2\alpha_2\sum_{i=3}^{N}\alpha_i d_i K_{i,2} + C \\ s.t.\ \ & \alpha_1 d_1 + \alpha_2 d_2 = -\sum_{i=3}^{N}\alpha_1 d_i \\ & 0 \le \alpha_1, \alpha_2 \le C \end{aligned}$$

Then, we have

$$\begin{aligned} \frac{\partial W}{\alpha_2} = & -(K_{1,1} + K_{2,2} - 2K_{1,2})\alpha_2^{new} + (K_{1,1} + K_{2,2} \\ & - 2K_{1,2})\alpha_2^{old} + d_2[d_2 - d_1 + f(x_1) - f(x_2)] \end{aligned}$$

where $f(x) = \sum_{i=1}^{N}\alpha_i d_i K(x_i, x) + b$
Denote $E_i = f(x_i) - d_i$ and $\eta = K_{1,1} + K_{2,2} - 2K_{1,2}$, we have

$$\frac{\partial W}{\alpha_2} = -\eta\alpha_2^{new} + \eta\alpha_2^{old} + d_2(E_1 - E_2) = 0$$

which leads to

$$\alpha_2^{new} = \alpha_2^{old} + \frac{d_2(E_1 - E_2)}{\eta}$$

Since α_1 and α_2 should satisfy the constraint, so α_2^{new} needs to be clipped as

$$\alpha_2^{new} = \begin{cases} H & \alpha_2^{new} > H \\ \alpha_2^{new} & L \le \alpha_2^{new} \le H \\ L & \alpha_2^{new} < L \end{cases}$$

where L, H are defined as:
when $y_1 \neq y_2$

$$\begin{cases} H = min(C, C + \alpha_2^{old} - \alpha_1^{old}) \\ L = max(0, \alpha_2^{old} - \alpha_1^{old}) \end{cases}$$

when $y_1 = y_2$

$$\begin{cases} H = min(C, \alpha_2^{old} + \alpha_1^{old}) \\ L = max(0, \alpha_2^{old} + \alpha_1^{old} - C) \end{cases}$$

Therefore, we can optimize α_1

$$\alpha_1^{new} = \alpha_1^{old} + d_1 d_2 (\alpha_2^{old} - \alpha_2^{new})$$

Also, we can update b

$$b^{new} = \begin{cases} b_1 = -E_1 - d_1 K_{1,1}(\alpha_1^{new} - \alpha_1^{old}) - d_2 K_{1,2} \\ \quad (\alpha_2^{new} - \alpha_2^{old}) + b^{old},\ 0 < \alpha_1^{new} < C \\ b_2 = -E_2 - d_1 K_{1,2}(\alpha_1^{new} - \alpha_1^{old}) - d_2 K_{2,2} \\ \quad (\alpha_2^{new} - \alpha_2^{old}) + b^{old},\ 0 < \alpha_2^{new} < C \\ b_3 = \frac{1}{2}(b_1 + b_2),\ otherwise \end{cases}$$

The SMO algorithm is done by iteratively updating every α and b until no change can be done. In my SVM algorithm, SMO is done for fixed times which is controlled by user to control the trade-off between efficiency and accuracy.

1.4 SVM On Multiple Classes

The classic SVM focus on classification between two classes, which prevents the application of SVM directly in the classification problem between more than two classes. Therefore, some updates need to be done to better the algorithm to suit more complex situations such as CIFAR-10 and MNIST.

The strategy used here is to train a number of SVM model and apply every one of them on a x_i to correctly find its label. For example, we have a dataset with 3 labels, then 3 SVM models will be trained to classify data between label 1 and label 2, label 1 and label 3, label 2 and label 3.
To predict the label of x_i, we need to apply the 3 trained models on x_i and count the number of predicted label. If label 2 is predicted twice and label 1 is predicted only once, than the overall prediction is label 2 since the number of label 2 is larger than the number of label 1.
We
Similarly, the data has k labels. Then we should train C_k^2 SVM models and than apply each of them on a data and count the number of each label to find the largest one be the outcome.

1.5 Multiple Kernel In SVM

Sometimes single kernel SVM performs badly when the features of the data are highly complex, so a combination of multiple kernels are introduced to transform the data into a linearly separable feature space where single kernel can not transform. It can be expressed as

$$K_{i,j}^{mul} = \sigma_1 K_{i,j}^1 + \sigma_2 K_{i,j}^2 + ... + \sigma_n K_{i,j}^n$$

where K^i means the type of i-th kernel (linear, polynomial...). The aim is to find the coefficient σ_i of each kernel.
Consider a situation where the mixed kernel consists of two kernel K_1 and K_2 with their coefficients λ, $1 - \lambda$ and their mapping function $\phi_1(x), \phi_2(x)$.

In the feature space, the distance of x_i and x_j is

$$d = ||\phi(x_i) - \phi(x_j)||$$

In classification problem, we hope to maximize d between different labels and minimize d between the same label.

$$\begin{cases} max\ d,\ d_i d_j = -1 \\ min\ d,\ d_i d_j = 1 \end{cases}$$

Expand d with $K^1_{i,j}$ and $K^2_{i,j}$, we have

$$\begin{aligned} d(\lambda) &= A\lambda^2 + B\lambda + C \\ M &= K^2_{i,i} + K^2_{j,j} - 2K^2_{i,j} \\ A &= M - 2K^1(i,j) + 2 \\ B &= -2M \\ C &= M \end{aligned}$$

Therefore, we can define the evaluating function $L(\lambda)$

$$\begin{aligned} max\ L(\lambda) &= \sum_i^N \sum_{j=1}^{i-1} d(\lambda)|_{d_i d_j = -1} \\ &- \sum_i^N \sum_{j=1}^{i-1} d(\lambda)|_{d_i d_j = 1} \\ &= -\sum_i^N \sum_{j=1}^{i-1} d(\lambda) d_i d_j \end{aligned}$$

Since we can express $d(\lambda)$, we can express $L(\lambda)$ as

$$\begin{aligned} max\ L(\lambda) = &\sum_i^N \sum_{j=1}^{i-1} A d_i d_j \lambda^2 + \sum_i^N \sum_{j=1}^{i-1} B d_i d_j \lambda \\ &+ \sum_i^N \sum_{j=1}^{i-1} C \end{aligned}$$

When $L(\lambda)$ achieves its maximum

$$\lambda = -\frac{\sum_i^N \sum_{j=1}^{i-1} M d_i d_j}{\sum_i^N \sum_{j=1}^{i-1} (M - K^1_{i,j} + 2) d_i d_j}$$

We can then extend the method to multiple kernel with a decay coefficient d. The pseudo-code of Algorithm 1 is as follows.

Algorithm 1 Coefficient Calculation

Input: number of kernels n and their correspond types K^i

```
1: Initialization: $K1$, $K2$, $M \leftarrow 0$, $N \leftarrow 0$, $d$
2: $K1_{i,j} \leftarrow K^1_{i,j}$
3: for $i = 2$ to $n$ do
4:   $K2_{i,j} \leftarrow K^i_{i,j}$
5:   for $j = 1$ to number of data do
6:     for $k = 1$ to $j - 1$ do
7:       $M \leftarrow M + (K^2_{i,i} + K^2_{j,j} - 2K^2_{i,j}) d_i d_j$
8:       $N \leftarrow N + (2 - K^1_{i,j}) d_i d_j$
9:     end for
10:  end for
11:  for $l = 1$ to $i$ do
12:    if $\sigma_l = 0$ then
13:      $\sigma_l \leftarrow \frac{M}{M+N}$
14:      $flag \leftarrow True$
15:    else
16:      $\sigma_l \leftarrow \sigma_l \cdot \frac{M+d \cdot N}{M+N}$
17:      $flag \leftarrow False$
18:    end if
19:  end for
20:  if $flag = True$ then
21:    $\sigma_i \leftarrow \frac{N}{M+N}$
22:  else
23:    $\sigma_i \leftarrow \frac{(1-d)N}{M+N}$
24:  end if
25:  $K1_{i,j} \leftarrow \frac{M+d \cdot N}{M+N} K1_{i,j} + \frac{(1-d)N}{M+N} K2_{i,j}$
26: end for
```

Output: $\sigma_1, \sigma_2, ..., \sigma_n$

The idea is to calculate the first two kernels and the view the first two kernels as a large kernel to compute with the 3rd kernel... The decay coefficient here is to prevent the sharp decrease of the coefficients of the first several kernels as they iteratively plus with a coefficient $0 < \frac{M+d\cdot N}{M+N} < 1$.

2 Experiment

2.1 SVM With Single Kernel

The SVM has multiple kernel to choose and different types of kernel and different parameters matters. For a specific kernel, we can determine its ideal parameters by ablation experiment. Taking an example of polynomial kernel $K(x_i, x_j) = (\zeta + \gamma x_i^T x_j)^d$, the parameter to be determined are mainly its degree d and its scaler γ. So we can compute the accuracy in test set with different parameters and then compare to obtain the optimal parameters.

The dataset is CIFAR-10, in which every image has a size of $3 \times 32 \times 32$ and their HOG features are extracted to be the input data. I also tried CIFAR-10 without extracting its features and SIFT features. However, both of them performs worse than HOG features and each image's SIFT features contains different number of key points which causes some difficulties on processing these features to the same dimension. The result is shown in figure 4,

We can see from figure 4 that for a fixed γ, the accuracy drops to 0.1 as the degree increases and the larger of the γ, the faster accuracy drops to 0.1. Meanwhile, we also know that when $\gamma = 0.7$, degree=4, the accuracy reaches its maximal value 0.546.

Similarly, we repeat the ablation experiment with every kernel and then apply it on MNIST (original 28×28)and CIFAR-10 (HOG features). The result on test set is shown in table 1 The result of Sigmoid kernel, Linear kernel as well as Gaussian kernel on CIFAR-10 and MNIST is shown in figure 5, figure 6 and figure 7. Additionally, we found the efficiency of Gaussian kernel is extremely low compared with other three kernels due to its complex calculation. The efficiency of the three kernels is similar and all higher than the efficiency of Gaussian kernel. The efficiency on MINIST is overall higher than the efficiency on CIFAR-10 due to its small size of only 28×28 compared with HOG features of CIFAR-10 with a size of 1568×1.

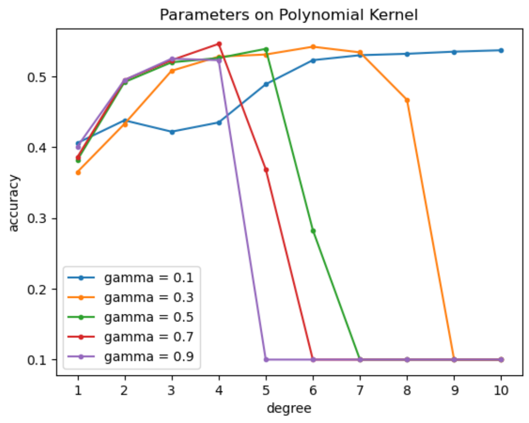

图 4: Result of Polynomial Kernel

Kernel	MNIST	CIFAR-10
Linear	0.899	0.450
Polynomial	0.939	**0.546**
Gaussian	**0.942**	0.533
Sigmoid	0.782	0.384

表 1: Result of Different Kernel

2.2 SVM With Multiple Kernels

When SVM combines multiple kernels together, it is capable to various complex situa-

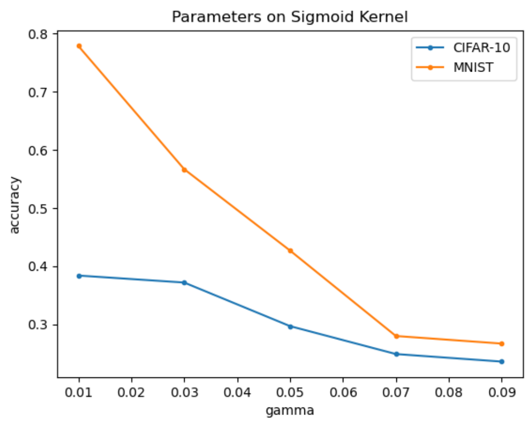

图 5: Result of Sigmoid Kernel

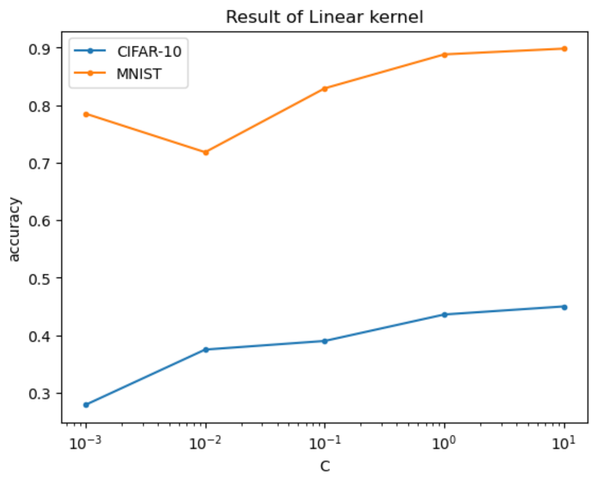

图 6: Result of Linear Kernel

tions. The type of kernel and its weights as well as each kernel's parameters become the key to this method.

In our experiment, we mainly focus on the combination of different kernels with each weight determined by the above-mentioned algorithm, and test their accuracy on both MNIST and CIFAR-10. The parameters we choose are according to the experiment of single kernel and select the parameters with which single kernel performs well. The result on MNIST test set is shown in table 2 and the result on CIFAR-10 test set is shown in table 3.

Line	Poly	Gaus	Sigm	Accuray
✓			✓	0.872
✓	✓			0.912
✓		✓		0.915
	✓		✓	0.886
	✓	✓		**0.927**
		✓	✓	0.876
✓	✓	✓	✓	0.862

表 2: Multiple Kernels on MNIST

Notice that the two table fix the decay parameter $d = 0.3$, in order to know effect of d, an ablation experiment is performed. By changing different d, test the accuracy of some combinations of 4 kernels on CIFAR-10. The result is shown in figure 8, notice that $S\&L\&G\&P$ means the combination of Sigmoid, Linear, Gaussian and Polynomial kernels.

Line	Poly	Gaus	Sigm	Accuray
✓			✓	0.301
✓	✓			0.348
✓		✓		0.324
	✓		✓	0.407
	✓	✓		0.355
		✓	✓	0.377
✓	✓	✓		0.363
✓	✓		✓	0.342
	✓	✓	✓	**0.465**
✓		✓	✓	0.366
✓	✓	✓	✓	0.42

表 3: Multiple Kernels on CIFAR-10

From the result of table 2, table 3 and figure 7, we know that when $d = 0.1$, the algorithm performs well and the combination of Sigmoid, Linear, Gaussian and Polynomial per-

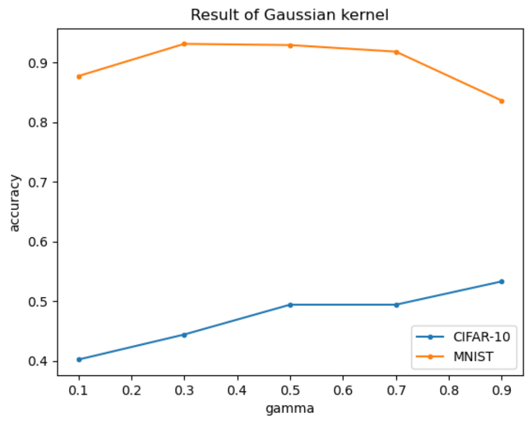

图 7: Result of Gaussian Kernel

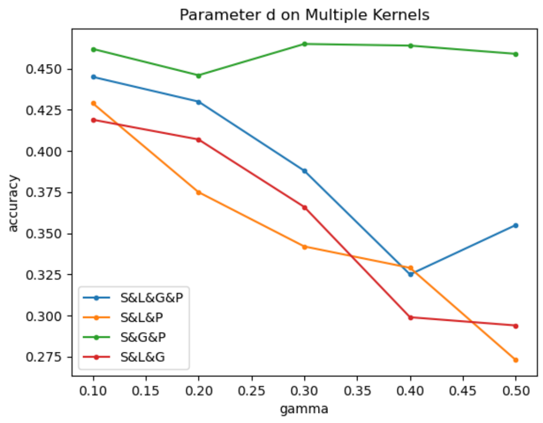

图 8: d on Multiple Kernels

forms best. Additionally, the accuracy is stable when d changes. However, the accuracy shows that the combination of multiple kernels may performs even worse than the single kernel with our method.
The efficiency is similar to single kernel: The efficiency of Gaussian kernel is extremely low and the efficiency of any combination with Gaussian kernel is much lower than the combination without Gaussian kernel.

3 Conclusion

In the project, we focus on SVM algorithm and extend the algorithm to classify multiple labels and study the algorithm of multiple kernels and the influence of different combinations and different parameters based on MNIST and CIFAR-10. In SVM with single kernel, we study the parameters and the type of kernels and found that Gaussian kernel and Polynomial kernel performs well. In SVM with multiple kernels, study the combination of kernels and the decay parameter and found that the combination of Sigmoid, Gaussian and Polynomial kernels performs well. And is less likely to be influenced by the decay parameter d. Until now, the problem of multiple kernels still lacks an optimal solution and is still waiting for an ideal method.

参考文献

[1] 王行甫, 俞璐. 混合核函数中权重求解方法. 计算机系统应用,2015,24(4):129-133

[2] https://zhuanlan.zhihu.com/p/29212107