Master Thesis Third Progress Report

Alhasan Abdellatif

March 2019

1 Short Literature Review

In the practical guide of SVM [1], written by the authors of LibSVM, they suggested that, when they number of features (n) is much greater than the number of instances (m), or they both are comparably large, using directly Linear kernel or LibLinear is sufficient for good accuracy and in particular when **both m and n are very large LibLinear is much faster than RBF kernel**. This justifies our use of LibLinear with L matrix, since in this case $n = m$ and the are very large for large datasets. However, in the case when n is much smaller than m , a usage of nonlinear kernel is preferred. They also emphasized on **scaling the features before applying SVM**, which can severely affect the accuracy.
A closely related work was developed by Zhang et al.(2004)[2], where they used the same sparse kernel we are working with. They pointed out some properties of the sparse kernel, developed techniques to tune the sparse parameter and performed tests on some application. One application is the **replacing of the RBF kernel by the sparse kernel in the dual problem of SVM**, they tested on a small dataset (100 instances) and reported the accuracy which were **very close to the one obtained using the RBF**. They also used the sparse kernel in training a **LS-SVM classifier**, which instead of optimizing a quadratic problem solves a linear system, and tested on relatively larger dataset of (5000) where they reported both accuracy and time, i**n the best case the sparse kernel saved the computation time by 47% while maintaining good accuracy.**

In [4], they used the sparse kernel for regression problems using LS-SVM. They tested on a sinc-toy problem with **1334 training points** in which it succeeds to obtain similar accuracy with a **speed-up of about 50% compared to the RBF kernel**. They also tested the kernel on a time-series prediction problem but it fails to give good accuracy.

2 Approximate-minimum-degree-permeation(AMD):

Algorithm used to reorder the rows and columns of a symmetric sparse matrix before applying Cholesky decomposition such that the Cholesky factor becomes sparser, $K = LL^T$.
Since the i^{th} row and column in the matrix K corresponds to the i^{th} training point in the dataset, reordering the rows and columns of K corresponds to reordering of the training points in the matrix L. Hence when using LibLinear, we have to reorder the labels of the training points.
In testing, we use the following equations :

$$(L^T D)\lambda = W \tag{1}$$

$$x \mapsto sgn(\sum_{i=1}^{m} \lambda_i y_i k(x_i, x) - b) \tag{2}$$

Since L and D correspond to a reorder version of the points, then so is λ. Hence when substituting in equation (2), where y_i is in the original order , λ needs to be reordered with the same permutation to be back in the original order as well.

3 Results summary

In the below experiments, **Sparse-RBF kernel is used in both training and testing**. In addition, **sparse-cholesky decomposition** is applied to the sparse-kernel matrix K with the **Approximate-minimum-degree-permeation algorithm**. Bold values indicates best outcomes.

Datasets Train/Test		Sparse Kernel		RBF Kernel		Sparsity	
	Sigma	Accuracy	Time(s)	Accuracy	Time(s)	K	L
Svmguide1 (3089/4000)	8	0.89	2.687	0.96	0.33	0.967	0.955
Magic (11411/7608)	8	0.81	18.8	0.73	15.8		
A6a (11220/21341)	0.5	**0.76**	**33.3**	0.77	51.4	0.999	0.998
	1	0.77	757.4	0.83	29.2	0.9486	0.645
IJCNN1 (49990/91700)	0.08	**0.905**	**200.3**	0.948	386	0.995	0.995
CodRNA (59535/80890)	1	**0.939**	**208.9**	0.94	233	0.996	0.997
	0.4	**0.933**	**225.5**	0.935	351.8	0.9989	0.999
Skin Segmentation (108783/71913)	0.4	**0.964**	**388.6**	0.988	497.5	0.9997	0.9998
	0.1	**0.967**	**359**	0.988	484.5	0.9998	0.9999

3.1 Implementation Issues

I believe that in order for the comparison to be **fair**, two points to be considered:

1- **Time** : In the website of user guide of SVM in sklearn library in python [3], they wrote that "Internally, we use libsvm and liblinear to handle all computations. These libraries are wrapped using C and Cython." Since the training time for the sparse kernel will include the time of the following:

1- Computation of the sparse RBF kernel

2- Cholesky decomposition

3- Solving Optimization problem , LibLinear

4- Solving the Linear system to obtain λ.

Steps 2, 3 and 4 use efficient libraries or a cython wrapper. However, step 1 which takes most of the time, other steps take few seconds, is implemented in python using for loop which, as I tested, is very slow for large number of iterations compared to one using cython. Hence, I

believe, for fair comparison, step 1 should be coded as a cython wrapper.
2- **Accuracy** : In [2], they combine the term 3σ as a parameter, τ, and they tested for various values of τ. I believe it is better if we do a similar analogy, we test both the sparse and RBF kernel for a range of σ, C and τ and then compare their best results in terms of time and accuracy.

4 Conclusion

We can observe that for large datasets (~50K,~60K,~100K), the algorithm reduces time compared to RBF Kernel, sometimes up to 48%, with small decrease in accuracy. However,the actual time can be much less than this, if we implemented the first step using cython wrapper. As we decrease sigma, the kernel and cholesky factor L becomes sparser which speeds up the computation.

5 References

1. https://www.csie.ntu.edu.tw/ cjlin/papers/guide/guide.pdf
2. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.294.6463rep=rep1type=pdf
3. https://scikit-learn.org/stable/modules/svm.htmlsvm-classification
4. ftp://ftp.esat.kuleuven.be/sista/hamers/ICANN3.pdf