Data mining scenarios for the discovery of subtypes and the comparison of algorithms

Colas, F.P.R.

Chapter 8

Does SVM Really Scale up to Large Bag of Words Feature Spaces?

In this chapter, we aim at developing better understanding of Support Vector Machines (SVM) in text categorization problems represented by sparse bag of words feature spaces. First, we identify those experimental settings that lead to best performing SVM solutions. Second, we discuss the nature of the solutions in these situations. Then, we describe a performance drop for SVM that occurs for particular combination of number of documents and of features. Next, we propose to relate the performance drop to classification noise in the data and we validate this hypothesis by additional experiments. Finally, before concluding, we discuss related work.

8.1 Introduction

We are concerned with the problem of learning classification rules in text categorization. Many authors presented Support Vector Machines (SVM) as the leading classification method [Joa98; Dum98; Yan99b; Zha01; Yan03; For03]. A number of studies, however, pointed out that in some situations SVM is outperformed by simpler methods such as naive Bayes or the nearest-neighbor rule [Dav04; Sch06; Col06b]. In this chapter, we study in detail the performance of SVM and the number of support vectors when varying the training set size, the number of features and, unlike existing studies, also the SVM free parameter C, which is the upper bound of SVM's dual representation (Lagrange multipliers).

In Chapter 7, we only searched for optimized versions of SVM in large *bag of words* feature spaces and we used subsequently these SVM for a comparative analysis when both the number of training documents and of features were varied.

This new study is different because we decided to also vary the parameter C and to study in detail the nature of the SVM solutions in terms of the number of Support Vectors (SV).

If in Chapter 7, we did not find C to influence the performance, here we show that tightly constrained SVM solutions with small C are high performers. However, most training instances are then bounded support vectors in these SVM solutions, which means that instances tend to be equally weighted with $\alpha_i = C$. Yet, an SVM solution where all the training instances share an equal weight is similar to the one of a nearest mean classifier. Because for such SVM solutions no training is necessary, it raises an interesting question on the merits of SVM in sparse *bag of words* feature spaces. In our experiments, we also report that SVM suffer from performance deterioration for particular training set size/number of features combinations.

The outline of this chapter is as follows. First, we identify the experimental settings that lead to the best performing SVM solutions. Second, we study in detail the nature of the SVM solutions for those experimental settings. Next, we report results that show a performance drop for SVM and we try to explain these results. Finally, we discuss related work and we conclude.

8.2 Experimental data

For the experiments, we use the data introduced in Chapter 6 (section 6.5.2): `C01-C21` from the `Ohsumed-all` datasets, the `20ng` from the `20newsgroups` datasets and the `RCV1` (Reuters Corpus Version 1).

8.3 Best performing SVM

In Figure 8.1 (a) and (c), we illustrate the performance behavior of SVM solutions when experimental settings are varying. In Table 8.3

Table 8.1: *Summary of the experimental conditions of Figures 8.1 (a) and (c).*

task	train (d)	test	features (f)	C	plotted data
(a) `20ng`	$d = 1448$	200	$f = 2^{\frac{i}{2}+b}, b = 3$	10^i	$maF_1(f, C)$
(c) `C01-C21`	$d = 4096$	547	$f = 2^{\frac{i}{2}+b}, b = 3$	10^i	$\frac{maF_1(f,C)}{max_C\ maF_1(f,C)}$

In (a), SVM performance is illustrated by classical learning curves. The size of the feature space is increasing and we choose different values of C. Similarly, the performance pattern of SVM on `C01-C21` is illustrated in (c) but this time, in order to discard the performance variability associated to the size of the *bag of words*, performance is normalised relative to the best C setting given a number of features. Thus, the contour line labelled 0.94 should be understood as 6% below

the performance of the best performing C setting ($C = 0.01$), whose contour line equals 1. In Table 8.2 we describe the experimental settings for Figure 8.1 (b) and (d).

Table 8.2: *Summary of the experimental conditions of Figures 8.1 (b) and (d).*

task	train (d)	test	features (f)	C	plotted data
(b) `C01-C21`	$d = 2^{\frac{i}{2}+b}, b = 3$	547	$f = 2^{\frac{i}{2}+b}, b = 3$	10^{-2}	$maF_1(f, d)$
(d) `C01-C21`	$d = 2^{\frac{i}{2}+b}, b = 6$	547	$f = 2^{\frac{i}{2}+b}, b = 3$	10^{2}	$maF_1(f, d)$

In (b) and (d), the performance differences between solutions with small (0.01) and large (100) C are illustrated through the contour lines; the number of features and the size of the training set are varying.

In Figure 8.1 (a) and (c), we illustrate how C affects the performance of SVM as the number of features is changing. First, for small feature spaces, i.e. 90-256 in (a) and 11-362 in (c), most of the C settings produce equally performing SVM solutions. However, as the feature space is further enlarged, performance varies widely from one setting to another. In particular, some C settings show a performance drop that will be discussed in a following section. In large feature spaces on (a) and (c), very small C values $\{0.01, 0.001\}$ are the best performing. The difference between the best C setting and others may be relatively small as in (c), i.e. 2-4% (between contour lines 0.96 and 0.98), but this depends on the classification problem. A final remark based on (c) relates to the setting $C = 10^{-4}$ that yields systematically lower performance. As will be shown in the following section, setting C to a very small value like 10^{-4} yields an SVM solution where all training documents are equally weighted with 100% bounded SV. This SVM solution becomes similar to the one of the *nearest mean classifier*.

More generally, we illustrate through (a) and (c) that C is a parameter that should be tuned in order to obtain the best performance of SVM. However, most of the studies in text categorization use default values of the implementation and thus neglect to tune C when training SVM[1]. Additionally, provided that C is tuned, we also illustrate that the best performances are achieved for the largest feature spaces. This confirms that domain knowledge in the form of feature selection is unnecessary for SVM. Finally, (b) and (d) show that more training documents yield better performance, as this is expected.

8.4 Nature of SVM solutions

In Figure 8.2 (a) and (b), the proportion of bounded SV, denoted by $\%(\alpha_i = C)$, is characterized by the contour lines; the feature space size and C vary. In (c),

[1]SVMLight sets the default value of C according to a heuristic. However, most other SVM implementations have fixed C default values (e.g. WEKA, libbow and libsvm).

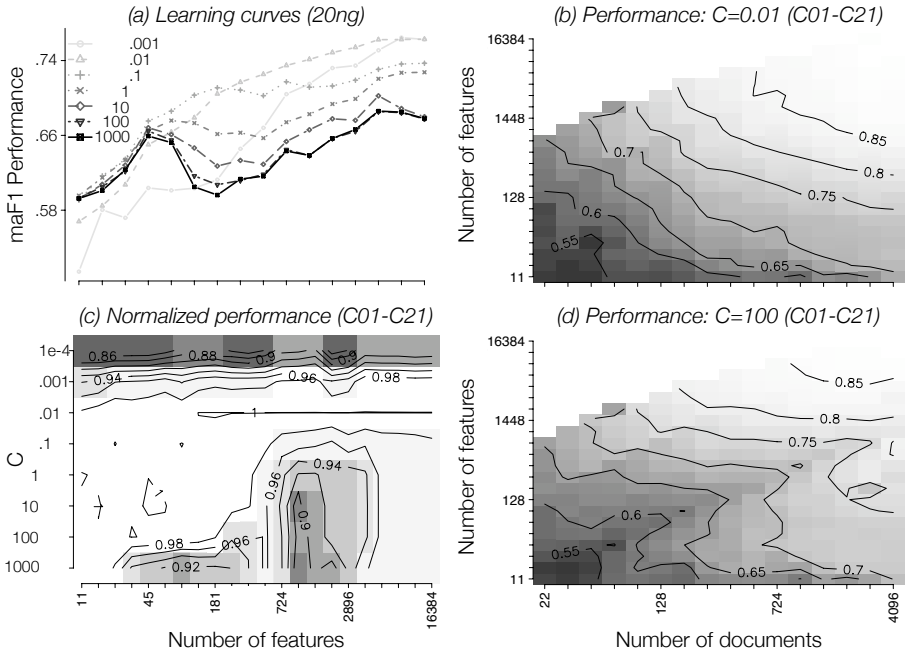

Figure 8.1: *Figure (a), (b) and (d) illustrate that SVM performance generally increases as the feature space and the training set are increasing. Depending on the C, performance may show a drop as in (a) around 724 features (`20ng`) and in (c) around 1448 features (`C01-C21`). In addition, (d) illustrates the linear dependency of the drop to the number of training documents and of features. Finally, (c) shows that the range of best performing C (in white, contour line* 1*) reduces towards small C values as the feature space is increasing.*

the solutions for different C are described when there are 11585 features. We only show it for this number of features because through Figure 8.1 we show that feature selection is not needed for SVM. Multiplier values α_i of the SVM solutions from the 10-fold cross validation are normalised by C and in the subfigure (c), we report the quantiles of the $\frac{\alpha_i(C)}{C}$ ordered by increasing value along the x-axis.

Finally, (d) reports the variation of the cross validated mean of the total number of SV (Support Vector) / in bound SV / at bound SV, when the feature space size varies and $C = 100$. We especially look in detail at these SVM solution because in Figure 8.1 (c), SVM displays a performance drop when the number of features matches the number of training documents per class, i.e. 2048 for a training set of 4096 documents.

Table 8.3: *Summary of the experimental conditions of Figure 8.2.*

task	train (d)	test	features (f)	C	plotted data
(a) `RCV1`	$d = 16384$	2024	$f = 2^{\frac{i}{2}+b}, b = 3$	10^i	$\%(\alpha_i = C)$
(b) `C01-C21`	$d = 4096$	547	$f = 2^{\frac{i}{2}+b}, b = 3$	10^i	$\%(\alpha_i = C)$
(c) `C01-C21`	$d = 4096$	547	$f = 11585$	10^i	$Q\left(\frac{\alpha_i(C)}{C}\right)$
(d) `C01-C21`	$d = 4096$	547	$f = 2^{\frac{i}{2}+b}, b = 3$	10^i	$\#(\alpha_i), C = 100$ $\#(0 <= \alpha_i < C)$ $\#(\alpha_i = C)$

Small feature space SVM's To better understand the behavior of SVM in small feature spaces (11-362 features), we put Figures 8.1 (c) and 8.2 (b) next to each other. For 8.1 (c), we first remark that any C setting performs equally. Then, in 8.2 (b), we notice that for those settings, SVM solutions exhibit large proportions (30 to 100%) of bounded SV, which have $\alpha_i = C$.

In fact, as the *bag of word* feature space is discrete and sparse, training points are located in a finite number of positions. Then, when representing the problem with only few features, it is likely that *training points* can belong to distinct classes; yet, no hyperplane can separate these overlapping points and as a result, their α_i in SVM will tend to infinity unless a soft margin C bounds them ($\alpha_i = C$). Remarkably, we also noticed that at constant feature space size, the number of overlaps will increase along with the number of training documents.

To conclude, it seems that *small feature space SVM's* are mostly defined in terms of bounded SV because of the points from distinct classes which overlap and are non-separable. *Small feature space SVM's* are therefore very similar in terms of proportions of inactive training points / bounded and unbounded SV, which may explain why they yield the same performance.

Large feature space SVM's As illustrated in Figure 8.2 (a), (b) and (d), in contrast to the small feature space SVM which are mostly characterized by bounded SV, the *large feature space SVM* are mostly based on unbounded SV. In fact, the Figure 8.2 (d) shows that the number of unbounded SV raises along with feature space.

This demonstrates that every training point tends to define its own local class boundary as feature space increases and correspondingly, that the definition of the convex hull of each class requires more training points in higher dimensions.

In Figure 8.2 (d), the number of bounded SV stagnates as the feature space increases. These residual SV are *outliers*, e.g. due to a mis-labelling, that would lie within the other class concept, thus avoiding any possible good classification by a linear hyperplane. Furthermore, we interprete the low performance of *large*

feature space SVM having high C in Figure 8.1 by the influence of those outliers whose $\alpha_i = C$ is likely to contribute too much to the final solutions.

Tightly constrained SVM's In Figures 8.2 (a) and (c), as C become very small, the proportion of bounded SV in SVM solutions tends to 100% for all feature space sizes; here, we are especially concerned with the *tightly constrained SVM* ($C = 10^{-4}$) that exhibits 100% of bounded SV.

In section 6.3.3, we introduced the *geometrical margin* and we expressed its limiting value to infinity as C would near zero. Then, if a sufficiently small value of C is taken, the *geometrical margin* will enclose all training points leading to SVM solutions solely defined in terms of bounded SV ($\alpha_i = C$). As all training points have equal weight, these solutions reduce to a simple *nearest mean classifier*.

8.5 A performance drop for SVM

In the study discussed in Chapter 7, we observed a "drop" in performance in a number of classification tasks. This means that the drop does not relate to a particular task, although we do think that its importance does. Similarly, by repeating our findings given two different SVM implementations, we could not relate the drop to the algorithm nor to its implementation. Here, we further analyse the SVM solutions in order to better understand the cause of the performance drop.

In Figure 8.1 (a), we observe a performance drop for 724 features when $C \in \{1, 10, 100, 1000\}$; on a second classification task illustrated in Figure 8.1 (c), a drop also occurs when $C \in \{1, 10, 100, 1000\}$ but for (1024-2048) features. Further, Figure 8.1 (d), which reports SVM performance in terms of color / contour lines, illustrates particularly well that the performance drop occurs consistently for particular combinations of the number of training documents and the number of features.

In fact, it seems that this drop occurs when there are as many features as there are training documents per class; thus, 724 for `20ng` in Figure 8.1 (a) and 2048 for `C01-C21` in Figure 8.1 (c). Moreover, if we compare Figures 8.1 (c) (when $C = 100$) and 8.2 (a), we even notice that the performance drop matches with the drop of the total number of SV. In that case, the number of bounded SV declines faster than the number of *within* bounds SV.

Yet, lowering C influences the nature of the solutions (inactive, bounded, unbounded SV) such that solutions are mostly defined in terms of bounded SV, eventually reducing the SVM to a *nearest mean classifier* if C is set very small. In solutions where there are many bounded SV, these SV do not transform into unbounded which may explain why the learning curves of SVM with small C are not subject to a performance drop.

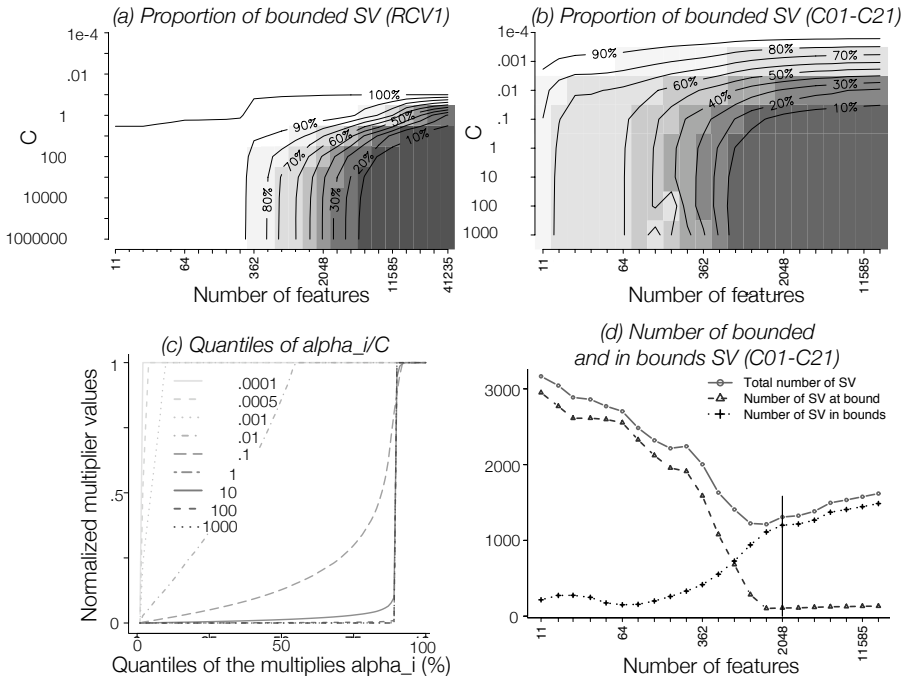

Figure 8.2: *In (a), the proportion of bounded SV ($\alpha_i/C = 1$) increases as C diminishes, and there are only bounded SV in SVM solutions for very small $C \in \{10^{-4}, 5\times10^{-4}\}$. In (b), the proportion of bounded SV is high for all C (11-362 features), but it decreases for higher feature space sizes as shown in (c). First, the total of SV follows the decrease in bounded SV (11-362). Then, the decrease is partially compensated by the appearance of unbounded SV (362-16384). The total of SV exhibits a drop slightly before 2048 features. In (b), for very small $C = 10^{-4}$, solutions consist only of bounded SV (100%) for any feature space size.*

8.6 Relating the performance drop to *outliers* in the data

In previous section, we showed that SVM is subject to a performance drop for particular combination of feature space size and number of training documents. Here, we conduct additional experiments on a classification task (`RCV1`, [Lew04]) in order to further validate the existence of the performance drop.

Performance drop on `RCV1` If we consider Figures 8.1 (c) and 8.3 (a), we notice that 8.3 (a) also exhibits performance drops for different C values $\{10^{-4}, 10^{-3}, 10^{-2}, 10^{-1}, 10^{3}, 10^{4}, 10^{5}\}$. Similarly, we see in 8.3 (a) that the best performing C values (where the contour lines equal 1) tend to lower as the feature space size increases.

However, the picture differs substantially for *tightly constrained SVM* (100% of bounded SV as illustrated in Figure 8.2 (a)) which are clearly affected by *several* performance drops. We may explain the many drops by the change in feature space (compared to our previous experiments). Here, we use the `RCV1` dataset, which features were transformed by *tfidf* weighing scheme as explained in [Lew04], whereas in our previous experiments, we used the raw frequencies.

In addition, by processing the features, the `RCV1` classification task differs significantly from `C01-C21` and `20ng` because of the change in geometry of the space. Accordingly, as C values nearing zero can control the width of the geometrical margin, the range of C values that conducts to *tightly constrained SVM* is influenced.

Influence of misclassified data on the performance drop To assess how misclassified data influences the performance drop, we duplicated 740 of the positive documents in the `RCV1` dataset and then, we labelled them as negative; this results in about 7% of intentionally misclassified and overlapping documents in the negative class. We report our results in Figures 8.3 (a) and (c), where (c) describes the performance on the data with misclassified documents.

In Figure 8.3 (c) the contour lines drop till .85 for 16384 features with $C = 10^6$, whereas in (a) the performance only drops till .95, the drop is more extreme in (c) than in (a). Therefore, our experiment confirms that for those feature spaces, along with large C values, the amount of misclassified or unseparable (e.g. overlapping) training points controls the importance of the performance drop. This also shows the *outlying* nature of the bounded SV residual in the SVM solutions illustrated in Figure 8.2 (a), (b) and (d) (`C01-C21` and `RCV1`).

Stability of SVM solutions In Figures 8.3 (b) and (d) we report the 10-fold cross-validated standard deviation of the performance for different C values and different feature space sizes.

First, we remark that *small feature space SVM's* is generally much less stable than *large feature space SVM's*. Further, as classification noise is added to the

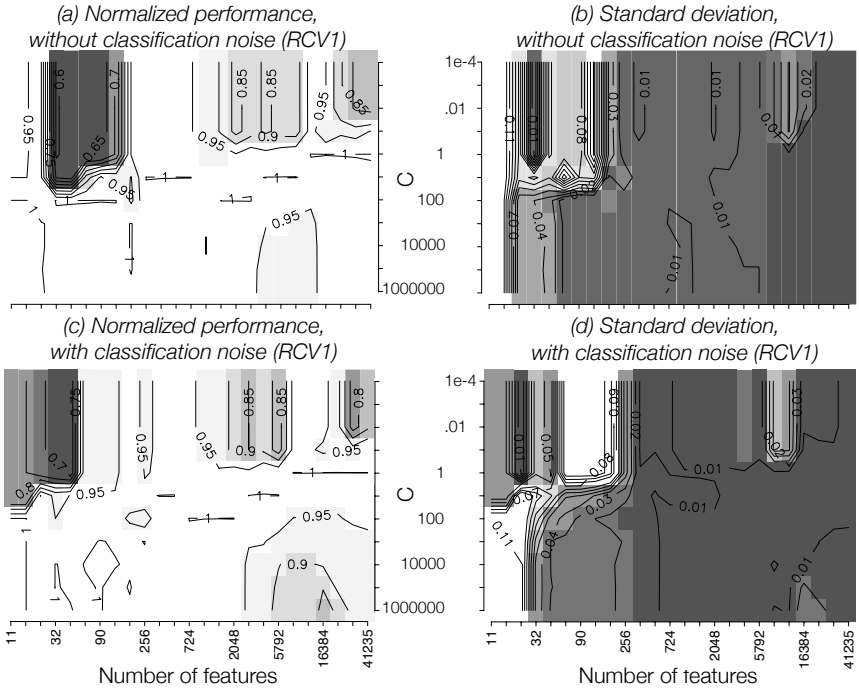

Figure 8.3: *In (c) and (d) experiments are the same as in (a) and (b) but with 740 positive documents duplicated into the negative class. As in Figure 8.1 (c), (a) and (c) detail SVM normalised performance whereas (b) and (d) report the performance standard deviation. Here, we could reproduce SVM performance drop and in view of (a) and (c), we further relate its importance to the level of classification noise in the data. Yet,* several *other performance drop occur for* tightly constrained SVM. *They may relate to the feature space geometry as here with* **RCV1**, *we operate in a* tfidf-*transformed space. Finally, (b) and (d) show that* small feature space SVM *are consistently less stable than* larger ones.

data in 8.3 (d), *small feature space SVM's* (11-256 features) are consistently less stable for all C values. Furthermore, in accordance to the several performance drops observed for small C values (illustrated in Figures 8.3 (a) and (c)), the *tightly constrained SVM's* can be less stable at identical feature space size than solutions with larger C.

8.7 Related work

In [Cri00], it is mentioned that *although the maximal margin classifier does not attempt to control the number of support vectors, [...] in practice there are frequently very few support vectors*. However, in this empirical study, we illustrate that SVM solutions are not sparse in large feature spaces. A similar observation in text classifiction was made in [Rou06] where it was observed that more than 75% of the dual variables were non-zero ($\alpha_i > 0$). In [Bur99] and [Rif99], SVM solution properties are analysed in terms of uniqueness and of degeneracy. In particular, the conditions for which all dual variables are at the bound are described, they can be referred to as "degenerate". In our experiments, we also gave experimental settings for which every training point is a bounded SV and we explained this gives trivial SVM solutions, i.e. *nearest mean classifier*. Furthermore, the study of [Mla04] raises the question whether sparsity of the feature space is a more reliable parameter than the number of features in predicting the performance of a classifier. Our experiments confirmed the specificity of the *bag of words* and its discrete nature.

8.8 Concluding remarks

Based on experiments, we systematically described the nature of SVM solutions in text classification problems when the number of training documents, the number of features and the SVM constraint parameter C are varying. In order to study SVM performance on equal grounds with other classification methods (e.g. the k nearest neighbors classifier and naive Bayes), training data was balanced and only binary classification tasks were considered.

In our experiments, we saw that SVM is consistently subject to a performance drop for particular combination of feature space size and number of training documents. This performance drop especially occurs when large C values are taken but we also observed *several* drops for small C values as we operated SVM on a classification task whose feature space was *tfidf*-transformed.

Further, we showed that this drop is a consequence of misclassified and overlapping data in the training set. In fact, as the SVM optimizer is unable to well-classify these "confusing" training points, their Lagrange multipliers (α_i) run-away until they reach the soft-margin upper bound C. Yet, although those points are useless for classification (because they overlap, they belong to distinct

classes and they counter-balance) they influence largely the final solution by having very large α_i values.

As in previous experiments we reproduced on a number of tasks a drop, it is very likely that those SVM solutions with large C values have a number of bounded SV that confuse the classification. We foresee two complementary ways to address this issue. First, the data should be cleaned before training the SVM and second, the SVM should implement a means to reduce the influence of the documents from the classification noise in the final solution.

Finally, SVM is often presented as an outstanding method in text classification; it is expected to find *sparse solutions* which would enable to classify quickly large amount of test documents at run time since SVM execution time depends on its sparsity. Yet, in our experiments, the sparsity of the solution reduces as larger feature space sizes are taken, which somewhat contradicts the common belief that all features should be used with SVM.