ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

Very Simple Classifier: a Concept Binary Classifier to Investigate Features Based on Subsampling and Locality

Luca Masera and Enrico Blanzieri

University of Trento, Italy

Abstract.

We propose Very Simple Classifier (VSC) a novel method designed to incorporate the concepts of subsampling and locality in the definition of features to be used as the input of a perceptron. The rationale is that locality theoretically guarantees a bound on the generalization error. Each feature in VSC is a max-margin classifier built on randomly-selected pairs of samples. The locality in VSC is achieved by multiplying the value of the feature by a confidence measure that can be characterized in terms of the Chebichev inequality. The output of the layer is then fed in a output layer of neurons. The weights of the output layer are then determined by a regularized pseudoinverse. Extensive comparison of VSC against 9 competitors in the task of binary classification is carried out. Results on 22 benchmark datasets with fixed parameters show that VSC is competitive with the Multi Layer Perceptron (MLP) and outperforms the other competitors. An exploration of the parameter space shows VSC can outperform MLP.

1 Introduction

The notion of *locality* in learning has a long history. Local models appeared first in density-estimation [8] and regression models [7], where kernels were used to control the influence of the samples to the overall model. The classical k-Nearest Neighbors classifier [3] is inherently a local method. In Nearest Neighbor and derived methods the attention focuses on ways of defining the distances or metrics to be used to find the set of neighbors and on the transformations of the space [4]. Moreover, theoretical results on k-Nearest Neighbors [5] gave a glimpse of the power of local models and, more generally, Vapnik and Bottou [2] established a fundamental result demonstrating that the local versions of base learners have better bounds on the generalization errors. The Vapnik and Bottou result leaves us with an effective strategy to improve classifiers by adding locality.

In this paper we present a novel approach to binary classification that is based on the idea of locality, and combine it with a classifier architecture typical of deep approaches. The main idea is to use a number of models to define linear separators, combine them with a confidence function that incorporates the information about the position of the samples and that uses the results as input of a single-layer perceptron. The rationale of the approach, which is motivated by the theoretical bound on local models given by Vapnik and Bottou,

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

Algorithm 1 VSC learning algorithm

Input: training data $\mathbf{X}$, labels $\mathbf{y}$, number of hyperplanes k, regularization factor λ
$\mathcal{P} \leftarrow$ select k pairs of examples of opposite class
for $j \leqslant k$ **do**
  $\mathbf{h}_j \leftarrow$ compute max margins hyperplanes for $p_j \in \mathcal{P}$
end for
for $i \leqslant |\mathbf{X}|$ **do**
  for $j \leqslant k$ **do**
    $\mathbf{X}'[i,j] \leftarrow \tanh(\langle \hat{\mathbf{x}}_i, \mathbf{h}_j \rangle)\ \mathcal{C}_{p_j}(\mathbf{x}_i)$
  end for
end for
$\mathbf{w} \leftarrow (\mathbf{X}'^T\mathbf{X}' + \lambda\mathbf{I})^{-1}\mathbf{X}'^T\mathbf{y}$

is to leverage the notion of locality to achieve good features that can be used in multi-layered classifiers.

In order to test the effectiveness of this idea, we defined a "concept" classifier called *Very Simple Classifier* (VSC) that incorporates an extreme version of the approach. In the case of VSC the local models are built using just 2 samples, the confidence function is based on geometric considerations and we show that it modulates locality in a way that is based on the generalized Chebichev inequality. Finally the parameters of the final perceptron are found with a regularized pseudo inverse. VSC is tested on a battery of benchmark datasets against relevant competitors. Despite its simplicity, the results of VSC are surprisingly good, showing that VSC is competitive with the Multi Layer Perceptron (MLP) and it outperforms other classifiers in the binary classification task. An exploration in the parameter space completes the comparison with MLP.

2 Very Simple Classifier

Let us assume an input normed space $\mathbb{R}^n$ and a set (of labels) $L = \{-1, 1\}$, and N samples $(\mathbf{x}_i, \mathbf{y}_i) \in S \times L$ for $i = 1, \ldots, N$ such that the $\mathbf{x}_i$ are i.i.d. variables of an unknown distribution $f(\mathbf{x})$. Let $\mathbf{y}_i = y(\mathbf{x}_i)$ with $y : S \to L$ be an unknown function that associates the sample $\mathbf{x}_i$ with its label $\mathbf{y}_i$.

From a structural point of view, VSC is similar to a three-layer MLP with $n + 1$ nodes in the first layer, $k + 1$ nodes in the second, and just one in the third. The extra nodes in the first and second layer are used as biases. Procedurally VSC introduces significant novel differences based on subsampling and locality. The main steps of VSC are (I) *the pair selection procedure*, (II) *the pre-computation of the separating hyperplanes*, (III) *the confidence measure for the hyperplanes*, and (IV) *the regularized weights learning*.

As shown by the pseudo-code in Algorithm 1, the learning procedure starts with the selection of k pairs of examples $p := (\mathbf{x}_p^+, \mathbf{x}_p^-)$ such that $y(\mathbf{x}_p^+) = 1$ and $y(\mathbf{x}_p^-) = -1$. Following the parallel with the MLP, the precomputed hyperplanes are used as fixed weights for the network between the first and the second layer. The activation function of the second layer is an hyperbolic tangent which is down-weighted by a confidence measure (to be defined in Section 2.2). Being

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

the weights fixed, the output of the second layer can be computed without further learning procedures. With the matrix of the outputs $\mathbf{X}'$ and the labels for the training set, the weights between the second and the third layer can be easily learned with the product of pseudo inverting the matrix $\mathbf{X}'$ with the vector of labels $\mathbf{y}$.

2.1 Hyperplane selection

Given a pair of samples $p := (\mathbf{x}_p^+, \mathbf{x}_p^-)$ in the input space, a good separating hyperplane is the one that maximizes the margin. In this simple condition the maximum margin separating-hyperplane $\mathbf{h}_p$ is uniquely identified as the hyperplane perpendicular to $\mathbf{v}_p = \mathbf{x}_p^+ - \mathbf{x}_p^-$ and passing for their center $\mathbf{c}_p = (\mathbf{x}_p^+ + \mathbf{x}_p^-)/2$. $\mathbf{h}_p = (\mathbf{v}_p^1, \ldots, \mathbf{v}_p^n, \langle \mathbf{v}_p, \mathbf{c}_p \rangle)^T$ where $\langle \cdot, \cdot \rangle$ is the inner product. There are, however, infinite formulations for this hyperplane. The canonical formulation for $\mathbf{h}_p$ by VSC is the hyperplane with unitary norm.

2.2 Hyperplane confidence

Each hyperplane selected at the previous stage depends only on 2 training samples, it is therefore important to add a confidence measure to limit its influence area. Let $\mathbf{x}_p^+$ and $\mathbf{x}_p^-$ be the samples used to build the hyperplane, and let $\mathbf{x}$ be the point to be classified with $\mathbf{h}_p$. Then the confidence measure $\mathcal{C}_p : \mathbb{R}^n \to (0, 1)$ is

$$\mathcal{C}_p(\mathbf{x}) = \sigma\left(\frac{d}{||\mathbf{x}_p^+ - \mathbf{x}||^2} + \frac{d}{||\mathbf{x}_p^- - \mathbf{x}||^2} - \frac{2d}{d^2}\right)$$

where $d = ||\mathbf{x}_p^+ - \mathbf{x}_p^-||/2$ and σ is the sigmoid function $\sigma(x) = 1/(1 + e^{-x})$. In the implementation a small $\epsilon = 0.01$ was added to each denominator in order to avoid divisions by zero. The geometric intuition is that the confidence of $\mathbf{h}_p$ for the point $\mathbf{x}$ is high if $\mathbf{x}$ is close to $\mathbf{x}_p^+$ or to $\mathbf{x}_p^-$. The value of d plays the role of smoothing the confidence around $\mathbf{x}_p^+$ and $\mathbf{x}_p^-$, such that the higher the value of d, the wider and smoother the confidence region will be.

2.3 Learning the hyperplane weights

Once the hyperplanes $\mathbf{H}$ of the first layer have been selected, we can construct the matrix $\mathbf{X}'$: $\mathbf{X}' = (\mathbf{x}'_{i,j}) = \big(\tanh(\langle \hat{\mathbf{x}}_i, \mathbf{h}_j \rangle)\, \mathcal{C}_{p_j}(\mathbf{x}_i)\big)$ where $\hat{\mathbf{x}} = (1, \mathbf{x}^1, \ldots, \mathbf{x}^n)^T$. $\mathbf{X}'$ is an $N \times k$ matrix where each entry $\mathbf{x}'_{i,j}$ is the result of the prediction for i-th training example $\mathbf{x}_i$ with only the j-th hyperplane $\mathbf{h}_j$ and the confidence measure. The weights for each hyperplane could be obtained by inverting the matrix $\mathbf{X}'$. In most cases, however, $k \neq N$, thus $\mathbf{X}'$ is not square and invertible. In order to compute the hyperplanes weights VSC takes advantage of the regularized pseudoinverse, also referred to as Tichonov regularization. This choice is common in RBF networks and it has been used more recently in Extreme Learning machines (ELM) [6] where the emphasis is on the speed of the computation. Thus $\mathbf{w} = (\mathbf{X}'^T\mathbf{X}' + \lambda\mathbf{I})^{-1}\mathbf{X}'^T\mathbf{y}$ where $\mathbf{I}$ is the identity matrix of size $N \times N$. The effect of λ is to smooth the decision boundary, otherwise very prone

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

to overfit: the higher the λ, the higher will be the regularization. In order to enhance the expressiveness of the VSC, a bias is added to this computation by adding 1 at the beginning of each line of the matrix $\mathbf{X}'$. The decision function for the VSC is thus:

$$y_{\text{VSC}}(\mathbf{x}) = \text{sign}\left(\sum_{p\in P} \mathbf{w}_p \tanh(\langle \hat{\mathbf{x}}, \mathbf{h}_p \rangle)\ \mathcal{C}_p(\mathbf{x})\right) + \mathbf{w}_0.$$

3 Results

VSC has been implemented in Python 2.7 following the scikit-learn standards. This choice allowed us to easily compare the VSC with other 8 well-known classifiers implemented in the scikit-learn suite, i.e. MLP, SVM with linear and RBF kernel, AdaBoost, naive Bayes, decision tree, random forests, and k-nearest neighbours classifiers. Moreover, we compared the performances of VSC with the ones of the Python implementation[1] of ELM [6].

The experiments have been conducted on 22 datasets retrieved from the Keel archive [1] with the only criteria of being binary classification problems with no categorical features. The data have been normalized by removing the mean and scaling to unit variance for each feature. The performances have been assessed with a 10-fold cross validation. The folds have been randomly generated keeping the positive-negative proportion unchanged. No parameter selection has been done, neither for VSC nor for the competitors. Thus all the experiments have been conducted with the parameters that are provided as default in the scikit-learn implementation. The SVM with RBF kernel has been trained with fixed $\gamma = 1$ instead of the adaptive version proposed in the implementation in order to be consistent with the choice of fixing the meta-paramenters of the classifiers. The $F1$ score has been preferred to the accuracy metric for presenting the results for its better robustness to unbalanced classes. The statistical significance is assessed with a paired two-tailed t-test with significance level $\alpha = 0,05$.

Each classifier has been trained with its default parameters. In particular for VSC we have used $k = 100$ hyperplanes and regularization factor $\lambda = 1$, these values have been decided a-priori, before the testing phase. For a fair comparison MLP and ELM have been trained with 100 hidden nodes. Table 1 reports the complete results, and a graphical representation of the statistically-significant results is shown in Figure 1a.

In order to test the effect of the confidence measure in VSC, we performed additional runs on the same 22 datasets of a modified version of VSC with confidence identically forced to 1, namely $\mathcal{C}_p(\mathbf{x}) \equiv 1$. The comparison with the modified VSC is showed in Figure 1b. With the exception of 7 datasets VSC outperforms the modified VSC. In particular, the only dataset in which the VSC is outperformed by the modified version with statistical significance is monk-2, which is a synthetic dataset with discrete features that, as shown in Table 1, is particularly hard for VSC.

[1] `https://github.com/dclambert/Python-ELM`

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0. Available from http://www.i6doc.com/en/.

	VSC	MLP	SVM RBF	AdaBoost	SVM Linear	Random Forest	K Neighbors	ELM	Naive Bayes	Decision Tree
app	0.91	0.87	0.92	0.92	**0.92**	0.89	0.92	0.81▾	0.90	0.87
ban	**0.91**	0.91	**0.91**	0.76▾	0.71▾	0.89▾	0.90▾	0.91	0.71▾	0.86▾
ban	0.71	0.65	**0.77**	0.66	0.73	0.64▾	0.67	0.70	0.08▾	0.55▾
bup	0.77	**0.78**	0.76	0.74	0.76	0.75	0.66▾	0.72	0.51▾	0.71▾
coi	**0.96**	0.96▾	0.97▾	**0.97**	**0.9**	0.96▾	0.97▾	**0.96**▾	0.10▾	0.94▾
hab	0.81	**0.84**▵	0.84▵	0.83	0.84	0.80	0.82	0.83	0.84▵	0.73▾
hea	**0.87**	0.85	0.80▾	0.82▾	0.857	0.82▾	0.86	0.83▾	0.86	0.77▾
hep	0.90	0.86	0.91	**0.93**	0.87▾	0.90	0.90	0.80	0.64▾	0.84▾
ion	**0.93**	0.92	0.89	0.93	0.92	**0.94**	0.89▾	0.90	0.91	0.90
mag	0.88	0.90▵	**0.90**▵	0.88	0.85▾	0.90▵	0.89	0.87▾	0.81▾	0.86▾
mam	0.82	0.83	0.83	**0.84**	0.83	0.80	0.80▾	0.81	0.81	0.78▾
mon	0.90	0.99▵	0.96▵	**1.00**▵	0.81▾	0.98▵	0.89	0.89	0.87	**1.00**▵
pho	0.89	0.89	0.90▵	0.87▾	0.84▾	**0.93**▵	0.92▵	0.87▾	0.82▾	0.91▵
pim	0.83	0.82	0.81▾	0.81▾	**0.84**	0.80▾	0.80▾	0.81▾	0.82▾	0.76▾
rin	0.97	0.96▾	0.89▾	0.96▾	0.78▾	0.93▾	0.77▾	0.85▾	**0.98**▵	0.88▾
son	0.63	0.69	0.70	**0.72**	0.67	0.65	0.67	0.59	0.56	0.63
spa	0.93	**0.95**▵	0.86▾	0.95▵	0.94▵	0.95▵	0.92▾	0.81▾	0.83▾	0.91▾
spe	0.87	0.86	**0.88**	0.87	0.881	0.87	0.83▾	0.86▾	0.74▾	0.85▾
tit	0.86	0.85▾	0.86	0.85▾	0.85▾	0.86	0.83▾	**0.87**▵	0.84▾	0.86
two	0.98	0.98	0.97▾	0.96▾	0.98	0.94▾	0.97▾	0.94▾	**0.98**	0.84▾
wdb	0.97	**0.98**	0.88▾	0.97	0.98	0.97	0.98	0.91▾	0.94▾	0.95▾
wis	**0.98**	0.97	0.97	0.96▾	0.97	0.97	**0.98**	0.97	0.971	0.96▾
Avg	0.88	0.88	0.87	0.87	0.85	0.87	0.856	0.84	0.75	0.87
Med	0.88	0.89	0.89	0.88	0.85	0.89	0.89	0.85	0.83	0.86

Table 1: The table reports the experimental F1 measure we obtained on the analyzed datasets. The best results for each dataset are marked in bold. Results that are statistically better and worse with respect to our method are marked with ▾ and with ▵ respectively.

From the analysis of Table 1, VSC emerges as the classifier whose performance is the best in the highest number of datasets (5 datasets, of which 3 are ties). For many datasets VSC presents a number of statistically-significant differences against the competitors: 73 times VSC is better and 26 times is worse. 46 out of the 73 times where VSC is significantly better occur in the 14 datasets where VSC is never significantly worse of any competitor. In other 4 datasets (magic, ring, spambase and titanic) VSC is significantly better more times than the other way around (4-3, 8-1, 5-4, 4-1 respectively). In just three datasets (haberman, monk-2 and phoneme) the number of competitors that are significantly worse is smaller than the number of competitors that are significantly better of VSC (1-3, 5-1, 3-4, respectively). Finally, in one last dataset (sonar) VSC is never significantly better or worse of any other competitor.

Considering the competitors, VSC shows good results. In fact, VSC is always significantly better more times that it is significantly worse, with MLP (3-4) as the only exception.

4 Conclusion

We have presented VSC, a classifier designed to test the idea that features which are based on the notion of locality can be effectively incorporated in a multilayer perceptron architecture. Max-margin hyperplanes are defined on a subset

ESANN 2019 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2019, i6doc.com publ., ISBN 978-287-587-065-0.
Available from http://www.i6doc.com/en/.

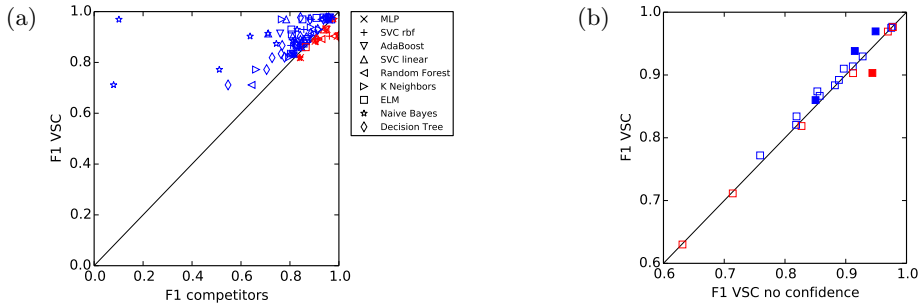

Fig. 1: (a) The datasets in which VSC significantly outperforms the competitor are marked in blue. The red marks, on the contrary, the datasets for which the competitor achieves better results. The symbol identifies the competitor. (b) The datasets in which VSC has $F1$ higher than the VSC with modified confidence are marked in blue. The red marks, on the contrary, are the datasets for which the modified confidence is better. The filled marks represent results statistically significant.

of the pairs of the samples with different classes and a confidence measure is defined. The results on 22 benchmark datasets shows that VSC outperforms the competitors with the exception of MLP, confirming the theoretical assumptions. The effectiveness of the confidence measure is also empirically verified.

The motivation of the work was to investigate the possibility that locality can produce features of high quality to be included in more complex architectures. Further studies will be important to evaluate the scalability in terms of size and dimensionality of the datasets.

References

[1] J. Alcalá, A. Fernández, et al. Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic and Soft Computing*, 17(2-3):255–287, 2011.

[2] L.n Bottou and V. Vapnik. Local learning algorithms. *Neural computation*, 4(6):888–900, 1992.

[3] T. Cover and P. Hart. Nearest neighbor pattern classification. *Information Theory, IEEE Transactions on*, 13(1):21–27, January 1967.

[4] S.t Dutta and A. Ghosh. On some transformations of high dimension, low sample size data for nearest neighbor classification. *Machine Learning*, 102(1):57–83, 2016.

[5] K. Fukunaga and L. Hostetler. k-nearest-neighbor bayes-risk estimation. *Information Theory, IEEE Transactions on*, 21(3):285–293, May 1975.

[6] G. Huang, Q. Zhu, and C. Siew. Extreme learning machine: theory and applications. *Neurocomputing*, 70(1):489–501, 2006.

[7] E. Nadaraya. On estimating regression. *Theory of Probability & Its Applications*, 9(1):141–142, 1964.

[8] E. Parzen. On estimation of a probability density function and mode. *The annals of mathematical statistics*, 33(3):1065–1076, 1962.