Tackling Multiple-Instance Problems in Safety-Related Domains by Quasilinear SVM

Christian Moewes[1], Clemens Otte[2], and Rudolf Kruse[1]

[1] Department of Knowledge and Language Engineering, University of Magdeburg, Universitätsplatz 2, D-39106 Magdeburg, Germany
{cmoewes,kruse}@iws.cs.uni-magdeburg.de
[2] Siemens AG, Corporate Technology, Otto-Hahn-Ring 6, 81739 Munich, Germany
clemens.otte@siemens.com

In this paper we introduce a preprocessing method for safety-related applications. Since we concentrate on scenarios with highly unbalanced misclassification costs, we briefly discuss a variation of multiple-instance learning (MIL) and recall soft margin hyperplane classifiers; in particular the principle of a support vector machine (SVM). According to this classifier, we present a training set selection method for learning quasilinear SVMs which guarantee both high accuracy and model complexity to a lower degree. We conclude with annotating on a real-world application and potential extensions for future research in this domain.

1 Introduction

Safety-related systems can be found in manifold fields where a failure may lead to fatalities or severe injuries to human beings, loss or very bad damage of equipment, or environmental harm [9]. The usage of machine learning methods is not that straightforward compared to other applications where learning machines have been applied very successfully.

Main differences to other classification domains are highly unbalanced classification costs and the infrequency of positive events, e.g., trigger events, and alarms. We try to compare this domain with multiple-instance (MI) learning [5] of which problems partly resemble safety-related applications. In contrast to single-instance supervised learning where one given example is represented by one feature vector (so-called instance), here an example is a set of feature vectors. Therefore, this setting of the learning problem is called *multiple-instance learning* problem. A set of multiple instances is named *bag*.

In binary pattern recognition with class labels $\{+1, -1\}$, a bag will be classified as positive if at least one of its instances is positive. It is negative if all of its instances are negative. This is also called the *MI assumption* [13].

This assumption is too general for safety-related applications where the final model must be highly interpretable. Thence we will tighten the MI assumption to have a MIL framework for the present domain. Before we will introduce our assumption and a possible approach to tackle safety-related problems, let us briefly recall MIL and safety-related applications.

1.1 Multiple-Instance Learning

In multiple-instance problems, one single training example (a positive or negative bag) is constituted by many different feature vectors, so-called *instances*. At least one is responsible for the observed class of the given example. Hence the class label is attached to the bag instead of the instances themselves.

Let us denote positive bags as B_i^+ and the jth observation of this bag as $\mathbf{x}_{ij}^+ \in \mathbb{R}^n$ where n is the dimensionality of the input space $\mathcal{X}$. The bag B_i^+ consists of l_i^+ instances $\mathbf{x}_{ij}^+$ for $j = 1, \ldots, l_i^+$. Consequently, the ith negative bag is denoted by B_i^-, its jth observation by $\mathbf{x}_{ij}^-$. Likewise, l_i^- symbolizes the number of instances in this negative bag. We denote the number of positive and negative bags as N^+ and N^-. The overall number of instances is referred to $l = l^+ + l^- = \sum_{i=0}^{N^+} l_i^+ + \sum_{i=0}^{N^-} l_i^-$. Thus the sample of all instances in negative and positive bags is listed by $\mathbf{x}_1, \ldots, \mathbf{x}_l$.

Nowadays many learning problems have been treated as MI problems, i.e., drug activity prediction [5, 6], stock market prediction [7], image retrieval [14, 15], natural scene classification [7], text categorization [1], and image categorization [4]. With the application to safety-related domains, another type of problem is identified as MI formulation under certain requirements.

1.2 Safety-Related Applications

Safety-related applications can be found in many real-world problems, e.g., condition monitoring of plants, automobiles, airplanes, and trains. These systems are supervised by many sensors collecting a (nearly) continuous multidimensional signal in time, e.g., speed, temperature, pressure, global position. Every time series itself describes one certain event of multiple instances. Regarding the MIL setting, we can state that every event corresponds to one bag which is either positive (e.g., a machine breakdown, alarm) or negative (e.g., proper machine operation, no-alarm). Thus it is necessary to binary classify these multivariate time series.

No instance in time of a negative bag must be classified as positive. A false positive in such an application usually involves severe injuries or harm to humans or machines. On the other hand, all positive events or bags have to be correctly classified before a certain limiting time has passed (e.g., time to exchange a machine before breakdown). If a positive event is recognized early enough, then certain countermeasures can be performed to prevent or moderate heavy accidents. These requirements meet the MIL setting. The following ones tighten the general MI assumption.

Since tests of such complex systems are very expensive and thus quite rare, there does not exist a vast of data (especially positive events). Hence a main disadvantage in those domains is the fact that formal proofs of the correctness of the learned classifier are not feasible [9]. Therefore, the model has to be enriched by experts' knowledge to ensure security requirements. Furthermore, we find unbalanced misclassification costs in safety-related domains very often s.t. constraints have to be added to the model as well.

It is not trivial to find the best trade-off between accuracy and model complexity. There exist some classifier for instance, support vector machine (SVM), that implicitly tries to satisfy both criteria. Taking advantages of the SVM's flexibility, we can even incorporate knowledge to meet unbalanced misclassification costs. This soft computing method will be introduced as extension of a linear separating hyperplane classifier in Sect. 2.

In safety-related applications, the model complexity in terms of simple functional dependencies is frequently the most important point. Quasilinear functions[1] with a good generalization performance must be found to establish a physical interpretation of human experts. Section 3 describes requirements for simple models regarding SV machines. After that, a combination of two methods is proposed to obtain a somehow simple and still accurate classifier. We conclude and discuss potential future work in Sect. 4.

2 Support Vector Machines

Let us formally introduce the basic concepts that we are going to talk about. Suppose we are given the input space $\mathcal{X}$ (not necessarily a vector space) and the output space $\mathcal{Y}$. Since we deal with a binary classification problem, $\mathcal{Y} = \{\pm 1\}$. We observe l training patterns $(x_i, y_i) \in \mathcal{S} \subseteq \mathcal{X} \times \mathcal{Y}$ where $i = 1, \ldots, l$. They have been drawn i.i.d. from an unknown distribution. If $\mathcal{X} \subset \mathbb{R}^n$, then $x_i \mapsto \mathbf{x}_i$. Our goal is to separate the data with a linear hyperplane $\{\mathbf{x} : \langle \mathbf{w}, \mathbf{x} \rangle + b = 0\}$ where $\mathbf{w} \in \mathbb{R}^n$ and $b \in \mathbb{R}$ are the norm vector and the bias of the hyperplane, respectively. The decision function of a hyperplane classifier which shall predict y' for any $\mathbf{x}$ corresponds to

$$f(\mathbf{x}) = \operatorname{sgn}\left(\langle \mathbf{w}, \mathbf{x} \rangle + b\right). \tag{1}$$

We are looking for the hyperplane that maximizes the margin between every training pattern and the hyperplane. Such a hyperplane is called optimal since it is unique and has the best generalization performance on unseen data. If all points $(x_i, y_i) \in \mathcal{S}$ can be separated linearly by a hyperplane, we can obtain the optimal hyperplane by solving a quadratic optimization problem with linear inequality constraints. Usually not all training patterns can be

[1] Quasilinear functions should be monotonic. They can be approximated with very few linear functions.

separated perfectly. Therefore we introduce slack variables ξ_i with $i = 1, \ldots, l$ in order to relax the optimization problem to

$$\underset{\mathbf{w},b,\boldsymbol{\xi}}{\text{minimize}}\ \tau(\mathbf{w}, \boldsymbol{\xi}) = \tfrac{1}{2}\left\|\mathbf{w}\right\| + C\sum_{i=1}^{l}\xi_i \tag{2}$$

$$\text{subject to} \quad y_i(\langle \mathbf{w}, \mathbf{x}_i\rangle + b) \geq 1 - \xi_i \tag{3}$$

$$\text{and} \quad \xi_i \geq 0,\ \forall i = 1, \ldots, l. \tag{4}$$

Here, $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_l)$ corresponds to the slack variables ξ_i and C is a global parameter that has to be determined by the user. The bigger C, the easier training patterns may violate the constraint (3). By introducing the Lagrangian of the primal problem (2), we end up solving the dual

$$\underset{\alpha_1,\ldots,\alpha_l}{\text{maximize}}\ \sum_{i=1}^{l}\alpha_i - \tfrac{1}{2}\sum_{i,i'=1}^{l} y_i y_{i'} \alpha_i \alpha_{i'} \left\langle \mathbf{x}_i, \mathbf{x}_{i'}\right\rangle \tag{5}$$

$$\text{subject to} \quad \sum_{i=1}^{l} y_i \alpha_i = 0 \tag{6}$$

$$\text{and} \quad 0 \leq \alpha_i \leq C,\ \forall i = 1, \ldots, l. \tag{7}$$

In practice, only few problems can be solved by a linear classifier. Hence the problem has to be reformulated in a nonlinear way. This is done by mapping the input space $\mathcal{X}$ to some high-dimensional feature space $\mathcal{H}$ by $\Phi : \mathcal{X} \mapsto \mathcal{H}$ where Φ satisfies Mercer's condition [11]. We can thus solve our nonlinear optimization problem linearly in $\mathcal{H}$ by computing the scalar product $K(x, x') = \langle \Phi(x), \Phi(x')\rangle$ which is called kernel. We simply replace the occurrence of the scalar product in (5) with a chosen kernel function. Finally, the discrimination function (1) becomes $f(x) = \operatorname{sgn}\left(\sum_{i=1}^{l} y_i \alpha_i K(x, x_i) + b\right)$.

For our purpose, let us have a look at the following two kernel functions[2]. First of all, we can apply the linear kernel

$$K(\mathbf{x}, \mathbf{x}') = \langle \mathbf{x}, \mathbf{x}'\rangle = \sum_{d=1}^{n} [\mathbf{x}]_d [\mathbf{x}']_d. \tag{8}$$

which performs the identical mapping $\Phi : \mathcal{X} \mapsto \mathcal{X}$. Second, kernel functions $K(\mathbf{x}, \mathbf{x}') = K(\left\|\mathbf{x} - \mathbf{x}'\right\|)$ generate radial basis functions e.g., the Gaussian kernel

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma \left\|\mathbf{x} - \mathbf{x}'\right\|^2\right). \tag{9}$$

3 Quasilinear Support Vector Machines

With respect to the SVM, the "linearity" of a SVM is expressed by the capacity of the function (so-called hypothesis) chosen by the principle of structural

[2] See [11] for a collection of kernel functions and further details on SVMs.

risk minimization (SRM) [12]. This principle of minimizing the expected risk controls the capacity s.t. the chosen hyperplane will guarantee the lowest error on unseen instances. Thus it heavily influences the complexity of the SVM.

The classification problem we deal with does not demand to correctly classify all positive instances. A positive bag will be correctly discriminated by at least one of its instances. On the contrary, all instances of negative bags have to be correctly recognized. Thence a pruning of positive examples that are hard to classify before the actual training is a way to simplify the process of model selection. The actual decision function will be selected by any suitable binary classifier e.g., SVM. This classifier might select a quasilinear hypothesis since conflicting positive instances have been removed.

Using a SVM to classify the pruned instance, we must choose an appropriate kernel function. Without having knowledge about the underlying distribution that generates the data, the Gaussian kernel (9) has shown good results in practice [11]. It is based on the Euclidean distance metric and thus intuitive. Due to the interpretability constraint in safety-related domains, $\mathcal{H}$ should geometrically correspond to $\mathcal{X}$. In [11] the authors argue that the linearity of the Gaussian kernel only depends on γ. For small γ, the SVM will determine a quasilinear discriminant function. A rather large γ causes narrow kernels which lead to complex nonlinear functions in $\mathcal{X}$.

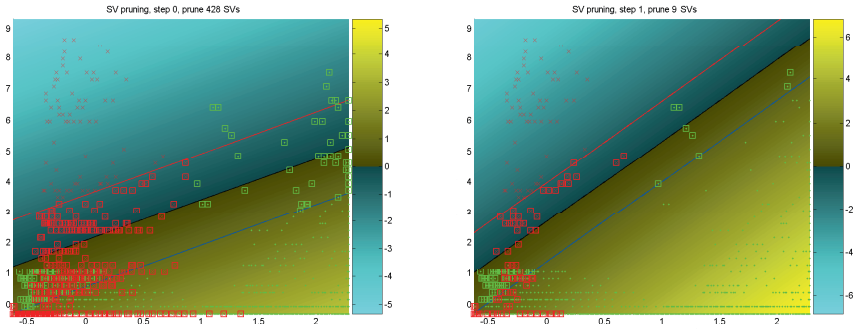
The linearity of a SV machine is a necessity to accept and approve our model. By decreasing the number of possible hypothesis, the potential solution becomes probably more linear and thus less complex. However, a too simple model might not generalize well on unseen bags. Thus the hyperplane must have the local ability to become more complex in order to ensure a higher accuracy for crucial bags.

3.1 Support Vector Pruning

A learning machine that discriminates events of security-related systems must be rather simple to be approved of security standards. Easier machines are favored instead of more complex ones. The SVM principle is theoretically well motivated, however, the feature space $\mathcal{H}$ is never expressed explicitly. If we construct $\mathcal{H}$ geometrically similar to $\mathcal{X}$ without using the linear kernel (8), it is possible to understand the resulting machine to a higher degree.

Quasilinear classifiers are preferred to complex models by pruning instances that are very hard to classify by a linear SVM. Since we deal with security-related domains, it is strictly forbidden to prune negative instances. Pruning is, however, feasible for positive bags (cf. Sect. 3). It removes candidates for critical instances from the dataset. It does not prune complete bags since every bag corresponds to a real-world event that has to be recognized. Thence at least m instances of every positive bag are kept even if they would have been linearly misclassified.

The pruning process is motivated by the search for a quasilinear classifier since linear dependencies are geometrically easy to interpret. Furthermore,



(a) (b)

Fig. 1. Example of SV pruning's first two steps on an artificial MI problem. Positive (negative) instances are shown as red crosses (green dots). The black line represents the decision boundary. The red (blue) line symbolizes the class margin of 1. Positive (negative) SVs are distinguished by red (green) squares around their instances. The color legends on the right side of the plots clarify the distance to the hyperplane. (a) Initial step found 428 positive instances for pruning. (b) Second step with pruned dataset determined 9 further positive instances which will be removed.

misclassified points will automatically become support vectors. The farthest positive SVs on the negative side of the hyperplane have a big influence on the model selection step. It is particularly very probable that those points will become support vectors even with a more sophisticated kernel.

The pruning can be explained briefly by the following 4 procedures:

1. Train a linear SV machine with all positive and negative patterns.
2. Identify misclassified positive support vectors.
3. Create a training set without these positive samples.
4. Repeat training until a stable model is obtained.

The third procedure has to assure that none of the bags will get empty. This is done by only pruning the farthest wrong positive support vectors of every bag s.t. the number of remaining instances is at least m. After all bags have been inspected, a new linear classifier is trained. The procedure begins again until no SV has been pruned. This algorithm converges relatively fast after approximately 6 iterations.

Fig. 1 shows an artificial application of SV pruning. The training of the linear SV machines has been performed with $C = 10$. At least $m = 10$ instances of every bag had to remain after pruning. In the first step, more than 400 instances of some positive bags have been pruned. Then 9 instances have been removed and thus not less than one linear SVM would have been trained for further pruning.

3.2 Bag Weighting

Quasilinear SV machines are very nice to have. On the contrary, the model shall still deploy all positive bags and prevent deployment of every negative bag. Thus a trade-off between simplicity and complexity has to be found. This section will introduce a modification of the standard SVM i.e., we locally allow the discriminant function to become more complex.

Reviewing (2) we find the global parameter C that expresses misclassification costs of all patterns. In particular, there is no a priori preference or priority of any pattern. Thence solving

$$\underset{\mathbf{w},b,\boldsymbol{\xi}}{\text{minimize}}\ \tau(\mathbf{w},\boldsymbol{\xi}) = \tfrac{1}{2}\left\|\mathbf{w}\right\| + C\sum_{i=1}^{l} C_i\xi_i$$

subject to (3) and (4), we introduce weights C_i for $1 \leq i \leq l$. It is straightforward to assign weights to complete bags as well. The user can influence the learning step by incorporating experts' knowledge in form of bag weights. The choice of the C_i is performed heuristically since these weights differ from problem to problem.

In combination with SV pruning, bag weighting can be a powerful tool to ensure both a quite simple model and the fulfillment of customer requirements i.e., high accuracy. It might be a good procedure to first apply the pruning method and then assign weights to misclassified bags. Remaining conflicts due to global model simplicity might thus be either removed or resolved.

4 Conclusions

In this paper we presented an hybrid approach for preprocessing MI problems in safety-related domains. Whereas classifiers for standard MI datasets aim to be as accurate as possible, we focused on learning machines of which model simplicity is essential. We introduced SV pruning to favor quasilinear classifiers. Bag weighting has been suggested to enable both the input of expert's knowledge and the trade-off between model simplicity and accuracy. The presented idea has been successfully applied to a safety-related system in automobile industry [8]. Due to the nondisclosure of this project, however, its empirical evaluation cannot be presented.

There are many possible extensions and improvements to the proposed methods. We will focus our research on generating fuzzy rules based on SV learning since fuzzy classifiers have been successfully implemented in safety-related applications (see [9]). Some approaches recently came up to construct fuzzy graphs from support vectors [3, 10, 2].

Preprocessing bags by SV pruning and bag weighting can be the basis for the following approach. The SVM could directly output fuzzy rules. Therefore, one would either have to formulate a differentiated optimization problem or define a special kernel that already includes domain knowledge. Both ways

might result in understandable fuzzy rules that still guarantee a good generalization. In addition, SV machines allow domain experts to comprise their knowledge to model learning. The whole concept might establish a powerful framework to find less complex classifiers not only in safety-related domains.

Acknowledgement. We would like to thank Eyke Hüllermeier who inspired us to cast safety-related problems into the framework of multiple-instance learning. Furthermore we acknowledge the critical and helpful comments of two anonymous reviewers to finalize this paper.

References

1. S. Andrews, I. Tsochantaridis, and T. Hofmann. Support vector machines for multiple-instance learning. In S. Becker, S. Thrun, and K. Obermayer, editors, *NIPS*, pages 561–568. MIT Press, 2002.
2. A. F. Chaves, M. B. R. Vellasco, and R. Tanscheit. Fuzzy rule extraction from support vector machines. In N. Nedjah, L. Mourelle, A. Abraham, and M. Köppen, editors, *HIS*, pages 335–340. IEEE Computer Society, 2005.
3. Y. Chen and J. Z. Wang. Support vector learning for fuzzy rule-based classification systems. *IEEE-FS*, 11(6):716–728, December 2003.
4. Y. Chen and J. Z. Wang. Image categorization by learning and reasoning with regions. *Journal of Machine Learning Research*, 5:913–939, 2004.
5. T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artif. Intell.*, 89(1-2):31–71, 1997.
6. O. Maron and T. Lozano-Pérez. A framework for multiple-instance learning. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *NIPS*. MIT Press, 1997.
7. O. Maron and A. Ratan. Multiple-instance learning for natural scene classification. In J. Shavlik, editor, *ICML*, pages 341–349. Morgan Kaufmann, 1998.
8. C. Moewes. Application of support vector machines to discriminate vehicle crash events. Master's thesis, FIN, University of Magdeburg, 2007.
9. C. Otte, S. Nusser, and W. Hauptmann. Machine learning methods for safety-related domains: Status and perspectives. In *Proceedings of FSCS*, pages 139–148, Magdeburg, Germany, 2006.
10. S. Papadimitriou and C. Terzidis. Efficient and interpretable fuzzy classifiers from data with support vector learning. *Intell. Data Anal.*, 9(6):527–550, 2005.
11. B. Schölkopf and A. J. Smola. *Learning with Kernels*. MIT Press, 2002.
12. V. Vapnik and A. Chervonenkis. *Theory of Pattern Recognition [in Russian]*. Nauka, Moscow, 1974. (German Translation: W. Wapnik & A. Tscherwonenkis, *Theorie der Zeichenerkennung*, Akademie–Verlag, Berlin, 1979).
13. X. Xu. Statistical learning in multiple instance problems. Master's thesis, Department of Computer Science, University of Waikato, 2003.
14. C. Yang and T. Lozano-Pérez. Image database retrieval with multiple-instance learning techniques. In *ICDE*, pages 233–243, 2000.
15. Q. Zhang, S. A. Goldman, W. Yu, and J. E. Fritts. Content-based image retrieval using multiple-instance learning. In C. Sammut and A. G. Hoffmann, editors, *ICML*, pages 682–689. Morgan Kaufmann, 2002.