Stream-Based Learning through Data Selection in a Road Safety Application

Nicolas Saunier[1,2], Sophie Midenet[1], Alain Grumbach[2*]
[1]*INRETS Institute, 2 avenue du général Malleret-Joinville, 94114, Arcueil*
[2]*ENST Paris, 46 rue Barrault, 75013, Paris*

Abstract. In this paper, we derive an incremental learning algorithm from a difficult real world task. Our application learns how to evaluate the risk of collision for road users at intersections, based on occupation measurements supplied by video sensors. The data are noisy and complex, and only a human expert can evaluate the risk on video recordings. In order to avoid arbitrary labeling, the expert is allowed to give fuzzy labels.

The performance of classical batch passive learning is not satisfying for our application. In order to improve, we were inspired by active learning and ensemble methods. The strong sequential ordering of the data entails a stream-based incremental learning. We present an original approach based on an active learner who selects a subset of the labeled data and combines by vote the hypotheses learnt during the incremental process.

1 Introduction

The work described in this paper meets the needs of a real world application. We study road safety in signalized intersections. For that purpose, we investigate new safety indicators of the risk for road users (see [1] for more details). We detect some categories of vehicle interactions and evaluate their severity, defined as the distance to the potential accident. Vehicles are interacting if they are ”close” enough in time and space, and the closer they are, the higher the severity. The severity of an interaction can be estimated through different measurable indicators [2] and the calibrated judgement of human experts. This paper deals with the learning of severity indicators based on images of the traffic at the intersection.

The input data are measurements of the spatial occupation of the intersection, supplied by multiple video sensors. The data and the traffic scenes were recorded in a complex real intersection over a period of 8 months. As real world data, they are complex and noisy (cf Fig. 1). As far as we know, there exists no automatic device to estimate the severity indicators. In our application, only a human expert can supply us with severity labels by watching the videos corresponding to the data. On account of the constraint of browsing VCR video tapes, we access the data sequentially. Our methods apply to an online setting.

Three entities interact in the learning of a classification task: an *expert*, a *learner* and a learnt *hypothesis*. Our description is inspired by [3]. Running a given algorithm, a learner is able to produce a hypothesis as close as possible to the target concept, here the severity of vehicle interactions. In classical learning tasks, the learner passively receives instances of the target concept, labeled by the expert (cf Fig. 2).

*This work is part of a PhD thesis supported by the région Ile-de-France.

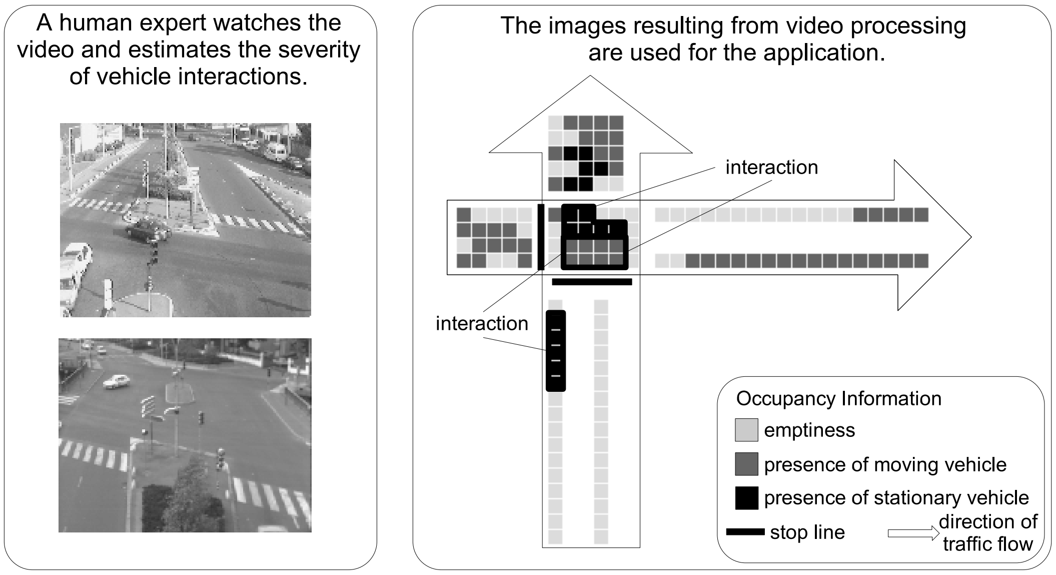
A human expert watches the video and estimates the severity of vehicle interactions.
The images resulting from video processing are used for the application.
interaction
interaction
Occupancy Information
emptiness
presence of moving vehicle
presence of stationary vehicle
stop line
direction of traffic flow

Figure 1: Video images and input data, with 2 detected interactions. The data are simplified for readability when printed in black and white. Each unit can be in 6 states, instead of 3 states in this figure: presence of moving vehicle covers presence of moving vehicle, trace of presence, beginning of presence and end of presence.

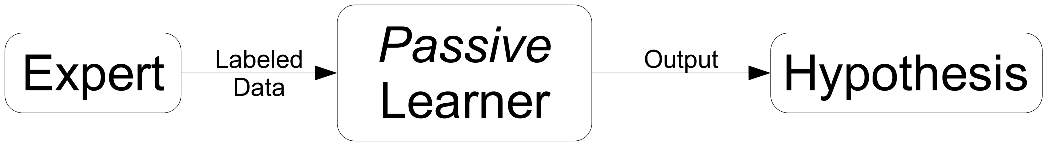

Figure 2: The learner is passive in the classical learning setting.

The human expert can only distinguish a few discrete levels of severity. In addition to the inherent human errors [4], it is hard to label instances which are close to some class boundaries. Using progressive transitions between classes is more adapted to the task. We can use membership functions to describe the classes identified by the expert. In order to avoid arbitrary labeling, the expert is allowed to label the instance as belonging with equal possibility to the neighboring classes: these labels are called fuzzy labels.

After studying and labeling some data, we noticed that some classes are very intricate and even overlap. Input data can be really close and at the same time differently labeled without hesitation by the expert. The classes are unbalanced. All these features of the data and their labels create a difficult learning problem, yielding relatively poor results with classical batch learning, where the passive learner uses a fixed set of input data only once to build a hypothesis.

In the batch setting, the learning set is randomly sampled from the underlying population, assuming simply that the distribution of the data is independent and identical [3, 5]. In the non-batch setting, called *incremental*, the learning process goes through more than one stage. In particular, in the stream-based setting, the learner has to make decisions and update the output hypothesis after each incoming instance. Learning incrementally allows the learner to make an intelligent data selection, without knowing the data distribution, and to specify the boundaries between the classes. In order to improve the results of our application, we want to select the most informative data to optimize the boundaries, especially between the intricate classes. To achieve this, we give control of the learning process to the learner to incrementally build a hypothesis. This is one of the definitions of the active learning framework.

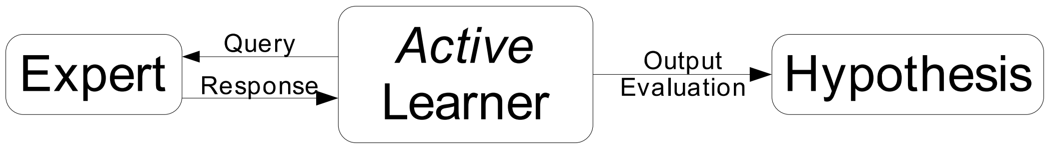

Figure 3: The active learner can ask the expert questions.

Active learning has mostly concentrated on membership queries, where the learner is allowed to query the expert for the class membership of certain instances (cf Fig. 3). Most active learning algorithms deal with the pool-based setting [3, 4, 6, 7, 8, 9, 10, 11, 12], in which the learner is presented with a fixed pool of unlabeled instances. For each step the learner chooses at least one instance from the pool to be labeled by the expert. The expert then provides the learner with the true label of this instance and the learner induces a hypothesis based on all the labeled instances seen so far. In our application, the data are accessed sequentially, which leads us to consider a variant, the stream-based setting [10, 13, 14]. In this setting, the learner is presented with a stream of unlabeled instances. For each incoming instance, the learner has to choose whether to ask the expert to label it or not. With respect to the existing active learning algorithms, we introduce the possibility of using the labeled instances for learning. This addition is not specific to the stream-based setting. The difference with the pool-based setting is highlighted in Fig. 4.

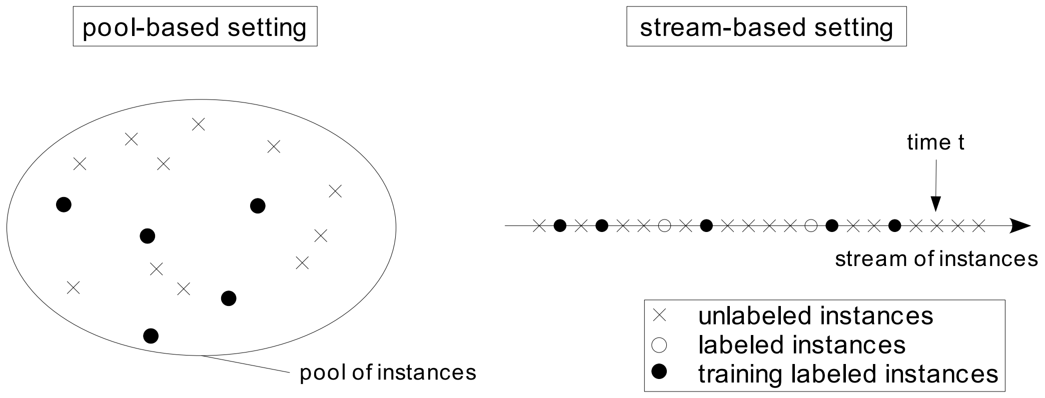

Figure 4: At time t, the learner can select *any* unlabeled instance, whereas in the stream-based setting, the learner is restricted to the *current* instance.

In this active learning framework we aim at selecting a subset of the data stream for learning in order to improve the results with respect to classical batch learning. We are not planning to reduce the number of labeled instances, but rather to enhance the learner performance for a fixed labeled set. Not using all labeled instances for training can seem to be counter-intuitive. However, we show that it can help in an application where the classes overlap.

Moreover incremental learning algorithms can provide us with intermediate results during the learning process (”anytime” algorithms), and with solutions to update hypotheses in the case of temporally evolving data. We plan to investigate these directions in future work. In the remainder of this paper, we describe the main algorithms and criteria to select the data (part 2), then evaluate our method in a real world application (part 3).

2 Stream-Based Incremental Algorithms for Data Selection

2.1 Generic Description

The generic principle of stream-based incremental algorithms for data selection is described in algorithm 1. The core of such an algorithm is the criterion for data selection. A first possibility consists in selecting the data before their labeling. In this case, an obvious idea would be to adapt active learning algorithms in the pool-based setting to the stream-based setting, by adding a threshold. In the pool-based setting, criteria try to optimize different measures. *Uncertainty sampling* selects the instance on which the current hypothesis has lowest certainty [12]. *Query-by-Committee* selects instances that maximize the disagreement among a committee of hypotheses [9, 10, 11]). Other works aim at reducing the size of the version space [3, 15]. None of these methods directly optimize the metric by which the learner will be ultimately evaluated, i.e. the learner's expected errors on future test sets, which can be estimated with the current learner and data sampling [7].

Algorithm 1: Stream-Based Incremental Algorithm for Data Selection.

```
Input: Let $A$ be the component learning algorithm
       Let $x_t \in X$ be the instance available at time $t$
       Let $y_t \in Y$ be the (unknown) class of $x_t$
       Let $S = [x_{1 \leq t \leq N}]$ be a stream of data
       Let $L_i = \{(x_t, y_t)\}$ be the learning set after the selection of $i$ labeled instances
       Let $h_i : X \rightarrow Y$ be the output hypothesis after running $A$ on $L_i$
Output: Final hypothesis $h_{final}$.
begin
    Label $t_0$ instances from $S$ and put them in $L_0$
    Run $A$ on $L_0$ with output hypothesis $h_0$
    $t = t_0 + 1; i = 0$
    repeat
        if $SelectionCriterion(x_t, h_i, L_i)$ satisfied then
            $i = i + 1$
            $L_i = L_{i-1} \cup \{(x_t, y_t)\}$ ($y_t$ is obtained in $SelectionCriterion$)
            Run $A$ on $L_i$ with output hypothesis $h_i$
        $t = t + 1$
    until $StoppingCriterion$
end
```

In the pool-based setting, the active learning algorithm will query the expert for the label of the instance(s) of the pool minimizing or maximizing one of these measures. It is straightforward to adapt these criteria in the stream-based setting with a threshold: for example, select the current instance x_t if the hypothesis confidence in its prediction is below a certain threshold. A major drawback of such adapted criteria is the tedious tuning of a threshold, and thus its lack of robustness for different tasks. Another possibility is to select the data once they are labeled. We present in the next part a criterion based on the misclassified instances.

Apart from the criterion for data selection, the stopping criterion and the choice of the final hypothesis are important components. These two elements are linked. If the quality of the hypotheses h_i learnt on the training set L_i is satisfying, the stopping criterion is not crucial and the final hypothesis is one of the learnt h_i. Otherwise, either a good stopping criterion, or a method to build a satisfying final hypothesis with the numerous hypotheses h_i. The present work is focused on the improvement of the quality of the final hypothesis by combining the h_i. This point is discussed later.

2.2 Criterion for Data Selection

First of all, the instances with fuzzy labels are not used in the work described in this paper. The expert is hesitating between two classes, and thus these instances bring less information than the instances that clearly belong to a class, in fact no information concerning the boundaries between these classes.

Our primary goal is to improve the learning performance for a fixed set of labeled instances, not to reduce the number of labeled instances. We choose to ask the expert for the labels of all incoming instances, then to select the instances that are misclassified by the current hypothesis (cf algorithme 2). Learning on misclassified instances has been studied in the supervised batch setting, e.g. with *Windowing* [16]. The author attributes the interest of the technique to the skewing of the true data distribution. *Boosting* techniques have the same goal [17, 18], resampling many times the data sets by focusing on the most difficult to learn. Our work follows the same principles as Boosting, and can be seen as an extension of Windowing to the stream-based setting. The idea has also been tried in pool-based active learning [8], except that the true labels were not known, but guessed by another classifier learnt on the labeled instances.

We select the misclassified instances in order to distort the real distribution of the data set towards difficult areas, areas where differently labeled instances can be very close and areas that have not been explored yet. A hypothesis learnt on a set in which the instances close to the boundaries between classes are over-represented should help specify these boundaries, in a better way than a hypothesis learnt the whole data set. This implies to use only a subset of the available data.

Algorithm 2: Criterion for data selection.

```
Input: Let $(x, y) \in X \times Y$ be the couple formed by the current instance and its class
       (unknown)
       Let $Y_{expert} = Y \cup \{fuzzy\ labels\}$ be the set of labels the expert can use for $x$
       Let $L = \{(x, y)\}$ be the current set of training labeled instances
       Let $h : X \to Y$ be the output hypothesis after running $A$ on $L$
Output: Selection of $x$.
begin
    Query the expert for the labeling of $x$: let $y_{expert}$ be this label
    if $y_{expert} \in Y$ and $h(x) \neq y_{expert}$ then
        Return true
    else
        Return false
end
```

2.3 Stopping Criterion and Final Hypothesis

Once a set of learning labeled instances is selected, how can we build the final hypothesis for use ? Most active learning algorithms in the pool-based setting use one of the hypotheses learnt in the active learning process. It is a reasonable choice as, the more labeled data there are, the better the performance. In this case, the stopping criterion is not crucial. In most works, the performance is just plotted with respect to the number of labeled data, and the reduction in the number of necessary labeled data highlighted. The stopping criterion and the methods to estimate the true accuracy of the hypothesis, that can be measured on an independent test set, are discussed in few papers. [7, 15] highlight the fact that the true accuracy of the hypothesis is not available. Even standard methods like cross-validation or leave-one-out fail to provide reliable true error estimates for an active learner, because the distribution of the learning set is biased by the active learning process towards difficult instances, and thus is different from the true distribution of the data. [7] introduces an indirect empirical estimator, the Classification Entropy Maximization, and [15] uses features specific to their SVM hypotheses, i.e. the number of new instances that become support vectors.

Algorithm 3: Incremental stream-based algorithm for the selection of learning data with the selection criterion of algorithm 2 and the notations of algorithm 1.

```
Input: Let $Vote_{i_1,i_2}(h_j)$ be the hypothesis obtained by taking majority votes over the
       predictions of $\{h_j \mid i_1 \leq j \leq i_2\}$
       Let $n$ be the number of hypotheses used in the $Vote$
Output: Final hypothesis $h_{final} = Vote_{max(0,i-n),i}(h_j)$.
begin
    Label $t_0$ instances from $S$ and put them in $L_0$
    Run $A$ on $L_0$ with output hypothesis $h_0$
    $t = t_0 + 1$; $i = 0$
    repeat
        if $SelectionCriterion(x_t, Vote_{max(0,i-n),i}(h_j), L_i)$ satisfied then
            $i = i + 1$
            $L_i = L_{i-1} \cup \{(x_t, y_t)\}$ ($y_t$ is obtained in $SelectionCriterion$)
            Run $A$ on $L_i$ with output hypothesis $h_i$
        $t = t + 1$
    until $StoppingCriterion$
end
```

However, if the performance doesn't constantly grow as a function of the number of selected learning data, a good stopping criterion has to be found. As we have seen, the evaluation of the performance of the current hypothesis in the active learning process is a difficult problem. It is possible to use an independent validation set. We focus on improving the quality of the learnt hypotheses. A solution is to build the final hypothesis as a combination of the hypotheses learnt during the learning process. [17] shows that such methods are more robust and accurate than its base learning method, especially when these base methods are unstable, i.e. sensitive to the learning sets. For example, *Bagging* and *Adaboost* are used in conjunction with query by committee in [9]. We suggest obtaining the final hypothesis by taking majority votes over the predictions of the set of the hypotheses built during the learning process, or a subset of it. We preferentially use the most recently learnt hypotheses because they are learnt on the biggest learning sets. For the same robustness reasons, the hypothesis used in the selection criterion is a combination of the previously learnt hypotheses. The new version of the algorithm is formally presented in algorithm 3. The tuning of the number n of hypotheses combined by *Vote* is discussed with respect to the application in the next part.

3 Evaluation on Real World Data

3.1 Experimental Data

We investigate stream-based algorithms for the selection of training data for a particular application, the automatic evaluation of the severity of vehicle interactions in an intersection. For this paper, we work on a specific severity indicator, a speed indicator, but our approach can be applied to any severity indicator.

The data come from an experiment run over a period of 8 months. The input data come from measurements of the spatial occupation of the intersection. The surface is divided into discrete units which indicate then the presence and dynamics of moving vehicles over a period of one second. These units can be in 6 different states (cf Fig. 1). The input vector of such units, i.e. the image of the intersection, have different sizes depending on the intersection. In this paper, we work with data coming from one intersection, and the input vectors are 80 units long. The expert identified 3 speed levels, minimum, medium and maximum (MIN, MED and MAX), and 2 progressive transitions between the MIN and MED classes, the MED and MAX classes, corresponding to the fuzzy labels ”MIN-MED” and ”MED-MAX”. The instances in the MED class are significantly more numerous than the instances of the two other classes and the instances of the MAX class are the most difficult to discriminate. This class holds the most severe interactions and therefore the most interesting for our road safety application.

We made the following evaluation with 3 different sets. We first completely labeled a sequence of data S, of about 30 minutes, yielding 510 instances among which 103 instances have fuzzy labels and are not used for learning. We built a test set out of 4 sequences of 10 minutes, with different traffic conditions, 2 dense and 2 fluid, yielding 541 instances, among which 170 have fuzzy labels and are not used for testing. Last, we draw randomly from a third set for each active learning trial 3 instances, one for each class, for the initial learning set L_0. This constitutes a local learning with respect to the whole available database gathered over a period of 8 months. Other techniques will be necessary to apply this work to all the data, particularly to monitoring the performance.

As hypothesis, we take a naive Bayes classifier because its performance in classical batch learning is better than the other tested hypotheses, even though the assumption of attribute independence is violated. The learning can be incremental and is computationally inexpensive. We use the implementation of the Weka toolbox [19].

3.2 Experimental Results

We first investigate the tuning of n (cf algorithm 3). We observe that the performance is sensitive to n. We choose $n = 7$, and notice that the performance worsens for other values.

We compare the performance of the final hypothesis built by our algorithm 3 called MC (selection of *M*is*C*lassified instances) at each instant t with the performance of hypotheses learnt on the instances of the set $S_t = \{x_{t'} \in S | t' \leq t\}$, the set of incoming instances seen until t:

”algorithm” BATCH : the hypothesis is simply learnt on S_t,

”algorithm” BAGGING : the hypothesis is learnt by Bagging on S_t, based on n naive Bayes classifiers each learnt on a random subset of S_t, containing as many instances as L_i, the set of instances selected by MC,

”algorithm” BATCH-EQ : the hypothesis is learnt on a random subset of S_t with a balanced class distribution,

”algorithm” BATCH-EQ-MC : the hypothesis is learnt on a random subset of S_t, with the same class distribution as L_i.

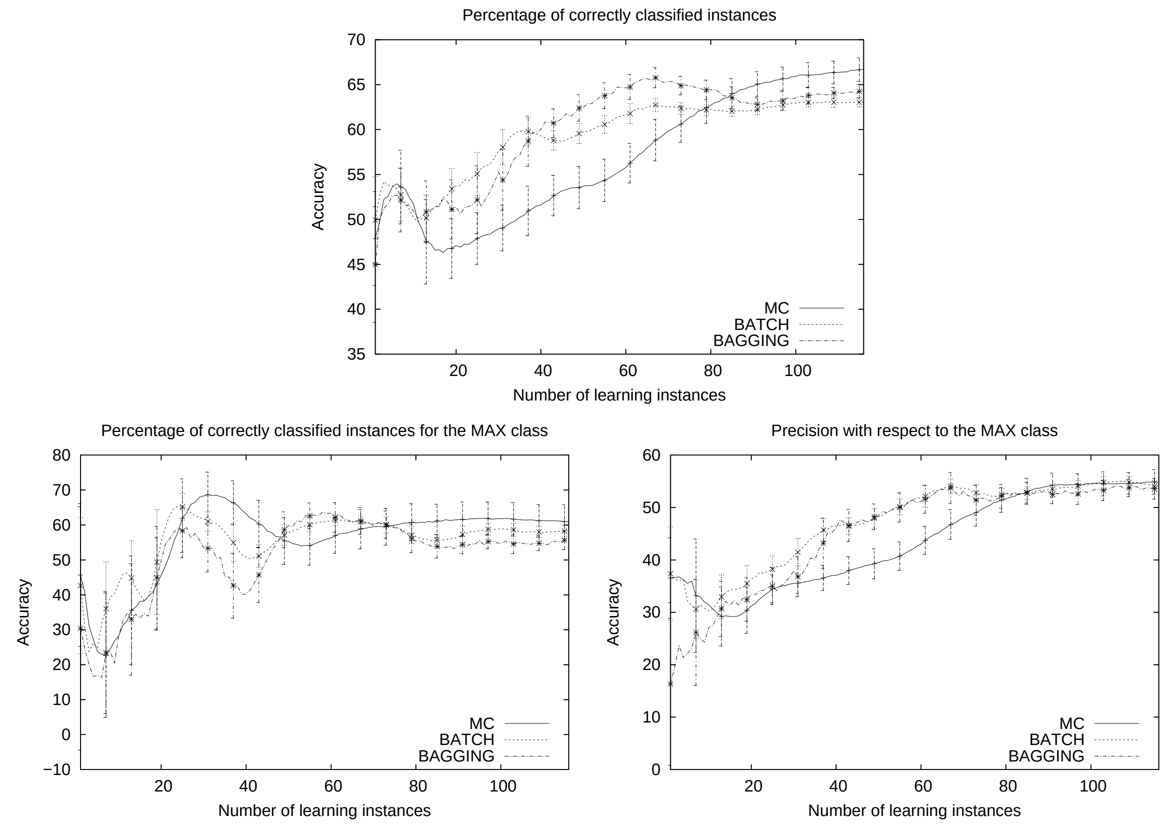

Figure 5: Learning curves of the algorithms MC, BATCH and BAGGING, as a function of the number of instances selected by MC; **Top**: Percentage of correctly classified instances for the 3 algorithms; **Bottom left**: Percentage of correctly classified instances belonging to the class MAX for the 3 algorithms; **Bottom right**: Precision with respect to the class MAX for the 3 algorithms. All values are estimated on the test set, averaged over 50 trials, with error bars (diluted do avoid clutter).

We compare BATCH and MC to show that it is possible to improve the performance by selecting a subset of the instances for learning. BAGGING combines hypothesis in the same way as MC for the final hypothesis. We thus want to show that MC actually selects instances that are more informative than those obtained by random sub-sampling. We also want to analyse the way MC works, in particular check if it is linked to a readjustment of the class distribution. For that purpose, we compare MC to BATCH-EQ and BATCH-EQ-MC, which is not a challenging algorithm since it requires the class distribution of the learning set selected by MC.

We plot the learning curves of MC and its challengers BATCH and BAGGING in Figure 5. Our algorithm performs best among the tested algorithms for the last third of the learning process (after 80 selected instances). However the performance is worse among the tested algorithms for the beginning of the learning. It is interesting to notice the quasi-regular increase in the performance of MC after random starts. Since we want to improve the results independently for all classes, we also show the results for the MAX class in Figure 5. The accuracy of the hypotheses produced by MC is better than that of the hypotheses built by the other tested algorithms, for the last third of the learning. The difference in precision[1] among the tested algorithms are not significant. Our algorithm performs well after the selection of a minimum number of instances.

	BATCH	BAGGING	BATCH-EQ	BATCH-EQ-MC	MC
CC	63,3 $\pm$ 0,4	65,9 $\pm$ 0,7	58,5 $\pm$ 2,2	63,3 $\pm$ 1,1	**67,3** $\pm$ **1,6**
CC (MIN)	83,1 $\pm$ 0,8	82,5 $\pm$ 1,0	**84,2** $\pm$ **1,7**	82,8 $\pm$ 1,7	74,9 $\pm$ 2,5
Pr (MIN)	51,2 $\pm$ 0,7	56,6 $\pm$ 1,5	47,3 $\pm$ 2,2	53,0 $\pm$ 1,9	**64,9** $\pm$ **3,3**
CC (MED)	59,5 $\pm$ 0,4	65,4 $\pm$ 1,0	48,0 $\pm$ 4,5	56,7 $\pm$ 1,8	**69,0** $\pm$ **2,4**
Pr (MED)	76,4 $\pm$ 0,4	75,2 $\pm$ 0,8	**78,0** $\pm$ **2,9**	78,0 $\pm$ 1,1	75,5 $\pm$ 1,8
CC (MAX)	58,9 $\pm$ 1,1	56,5 $\pm$ 1,9	64,0 $\pm$ 4,9	**64,7** $\pm$ **2,2**	59,2 $\pm$ 4,3
Pr (MAX)	55,2 $\pm$ 0,5	**57,6** $\pm$ **1,3**	49,4 $\pm$ 2,9	53,7 $\pm$ 1,5	55,1 $\pm$ 2,8

Table 1: Percentage of Correctly Classified instances (CC) and Precision (Pr) at the end of the stream S, global and independently per class (averaged over 50 trials with random initialization of L_0). For each line, the best performance appears in boldface. **Top**: Comparison of MC with BATCH and BAGGING; **Bottom**: Comparison of MC with BATCH-EQ and BATCH-EQ-MC

We present in table 1 the performance of the final hypothesis built by the 5 algorithms. We let MC run until the end of S is reached. The average number of selected instances is 136 ± 5, among 407 instances. In both tables, MC has the best global accuracy, and performs best for 2 class results. It is close to the best in 2 other cases, and significantly worse in 2 cases, the percentage of correctly classified instances of the class MIN and MAX. However, the other algorithms are better at the expense of the precision for the same classes which is much lower than that of MC. Although this is a difficult task, we observe that our algorithm MC has constant gains over the other algorithms. In particular, we show that selecting a subset of the learning instances can improve the performance. Random selection, a simple readjustment or the random skewing of the class distribution are not enough. Our algorithm selects informative instances that improve the quality of the learnt hypotheses.

[1]The precision of a hypothesis on a test set for a class is defined as the number of correctly classified instances of this class, over the number of instances classified by the hypothesis in this class. This criterion and the ration of correctly classified instances for the same class are complementary, since the latter can be very high, but the precision poor is the hypothesis classifies all instances in the class.

4 Conclusion

In this paper, we investigated stream-based algorithms for the selection of learning instances. We give the context, and present a solution based on misclassified instances and combination by vote of hypotheses for robustness.

We test our algorithm in a real world application, the evaluation of the severity of vehicle interactions in intersections. The results are promising. We show that our algorithm improves the global performance, and for some classes.

In the future, we plan to investigate the importance of the initialisation of the learning set. We would like to explore the possibilities offered by the fuzzy labels and the metrics of the target concept. We are also interested in other ensemble methods like boosting, which has similarities with our algorithm.

References

[1] Saunier, N., Midenet, S., Grumbach, A.: Automatic detection of vehicle interactions in a signalized intersection. In: 16th International Cooperation on Theories and Concepts in Traffic Safety Workshop, Soesterberg, Holland (2003)

[2] van der Horst, R.: A time-based analysis of road user behavior in normal and critical encounter. PhD thesis, Delft University of Technology (1990)

[3] Tong, S.: Active Learning: Theory and Applications. PhD thesis, Department of Computer Science of Stanford University (2001)

[4] Yan, R., Yang, J., Hauptmann, A.: Automatically labelling video data using multi-class active learning. In: Proceedings. Ninth IEEE International Conference on Computer Vision, 2003. (2003) 516–523

[5] Duda, R.O., Hart, P.E.: Pattern Classification. Wiley-Interscience (2000)

[6] Baram, Y., El-Yaniv, R., Luz, K.: Online choice of active learning algorithms. In: Proc. of The Twentieth International Conference on Machine Learning. (2003) 19–26

[7] Roy, N., McCallum, A.: Toward optimal active learning through sampling estimation of error reduction. In: Proc. 18th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA (2001) 441–448

[8] Iyengar, V.S., Apte, C., Zhang, T.: Active learning using adaptive resampling. In: Sixth ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining. (2000) 92–98

[9] Abe, N., Mamitsuka, H.: Query learning strategies using boosting and bagging. In: Proceedings of the Fifteenth International Conference on Machine Learning. (1998)

[10] Freund, Y., Seung, H.S., Shamir, E., Tishby, N.: Selective sampling using the query by committee algorithm. Machine Learning **28** (1997) 133–168

[11] Seung, H.S., Opper, M., Sompolinsky, H.: Query by committee. In: Computational Learning Theory. (1992) 287–294

[12] Lewis, D.D., Catlett, J.: Heterogeneous uncertainty sampling for supervised learning. In Cohen, W.W., Hirsh, H., eds.: Proceedings of ICML-94, 11th International Conference on Machine Learning, New Brunswick, US, Morgan Kaufmann Publishers, San Francisco, US (1994) 148–156

[13] Cesa-Bianchi, N., Conconi, A., Gentile, C.: Learning probabilistic linear-threshold classifiers via selective sampling. Lecture Notes in Computer Science **2777** (2003) 373–387

[14] Lewis, D.D., Gale, W.A.: A sequential algorithm for training text classifiers. In: Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, Springer-Verlag New York, Inc. (1994) 3–12

[15] Schohn, G., Cohn, D.: Less is more: Active learning with support vector machines. In: Proc. 17th International Conf. on Machine Learning, Morgan Kaufmann, San Francisco, CA (2000) 839–846

[16] Fürnkranz, J.: Integrative windowing. Journal of Artificial Intelligence Research **8** (1998) 129–164

[17] Schapire, R.E., Freund, Y., Bartlett, P., Lee, W.S.: Boosting the margin: a new explanation for the effectiveness of voting methods. In: Proc. 14th International Conference on Machine Learning, Morgan Kaufmann (1997) 322–330

[18] Schapire, R.E.: The strength of weak learnability. Machine Learning **5** (1990) 197–227

[19] Witten, I.H., Frank, E.: Data Mining: Practical machine learning tools with Java implementations. Morgan Kaufmann, San Francisco (2000)