ESANN 2018 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 25-27 April 2018, i6doc.com publ., ISBN 978-287587047-6. Available from http://www.i6doc.com/en/.

# Efficient Accuracy Estimation for Instance-Based Incremental Active Learning

Christian Limberg[1,2], Heiko Wersing[2] and Helge Ritter[1]

1- Bielefeld University, CoR-Lab
Universitätsstraße 25, 33615 Bielefeld - Germany

2- HONDA Research Institute Europe GmbH
Carl-Legien-Straße 30, 63065 Offenbach - Germany

**Abstract**. Estimating system's accuracy is crucial for applications of incremental learning. In this paper, we introduce the Distogram Estimation (DGE) approach to estimate the accuracy of instance-based classifiers. By calculating relative distances to samples it is possible to train an offline regression model, capable of predicting the classifier's accuracy on unseen data. Our approach requires only a few supervised samples for training and can instantaneously be applied on unseen data afterwards. We evaluate our method on five benchmark data sets and for a robot object recognition task. Our algorithm clearly outperforms two baseline methods both for random and active selection of incremental training examples.

## 1 Motivation

Estimating the classification accuracy of a machine learning model is particularly important in incremental learning tasks. This can be used to decide online whether further training is required for a predefined target performance. If the system accuracy does not improve with additional data the task may have to be delegated [1]. The estimated accuracy can also be used to optimize workload distribution in human machine interaction (HMI) scenarios where the system collaborates with a human to provide a joint target performance.

There is some related work focusing on accuracy estimation for unlabeled data. Platanios et al. [2] estimate classifier accuracy by considering the agreement rate of multiple different classifiers trained with independent features, possibly underestimating performance gains using all features. Another recent approach by Donmez et al. [3] is also applicable with a single classifier but requires the label distribution $p(y)$ for evaluating a maximum likelihood. This is applicable e.g. for medical diagnosis or handwriting recognition, where the marginal frequency of each class is known. Conformal prediction has been introduced for estimating a classifier's confidence [4] which is related to accuracy estimation.

While the mentioned accuracy estimation approaches have been applied in an offline learning setting, here we focus on incremental learning scenarios. Common choices to estimate the accuracy in incremental learning are to perform cross validation on a fraction of the data or to use the interleaved test/train error [5]. The latter requires a classification of each new example before using it for training. However, both approaches solely estimate the accuracy based on labeled instances. In most real world scenarios data is partially labeled since labeling information is usually expensive and difficult to obtain. Approaches including unsupervised information in their estimation should be able to deliver more precise results.

Our new approach is to train a regression model using the distance information of instance-based models to predict classifier accuracy with only a fraction of labeled data. We evaluate our method on five benchmark data sets and within an object recognition task, both with random sampling and with active querying. Our algorithm clearly outperforms the other methods in both cases.

## 2 Learning Vector Quantization

Instance-based classifiers have been shown to provide competitive approaches to incremental learning tasks [5, 6, 7]. We denote our target classifier as $C$ and choose as examples k-Nearest-Neighbor (kNN) and Generalized Learning Vector Quantization (GLVQ) ($C_{kNN}$ and $C_{GLVQ}$) for evaluation. Their representation of the data is interpretative, the number of classes does not have to be known beforehand, no complete retraining is necessary and efficient techniques for querying new samples in active learning were studied before [8].

LVQ is more memory efficient compared to kNN, because several samples can be represented by a single prototype. In LVQ, as originally proposed by Kohonen et al. [9], an n-dimensional training-set $X = \{(x_i, y_i) \subset \mathbb{R}^n \times \{1, ..., c\} | i \in \{1, ..., m\}\}$ is modeled through $p$ prototypes $W = \{w_1, ..., w_p\}$ and their assigned labels $L = \{\{l_1, ..., l_p\} | l_i \in \{1, ..., c\}\}$ for $c$ classes. Notice that $p$ can be varying during training. The receptive field or Voronoi region of a prototype $w_i$ is defined by $V^i = \{x \in X || x - w_i| \leq |x - w_j| \forall i \neq j]\}$ and an input sample can be classified by choosing the label of the nearest prototype, where the Euclidean distance is often used. During supervised training, the algorithm adjusts the position of the prototypes so they model the training set optimally. An efficient variant is generalized LVQ (GLVQ) [10]. The prototype positions are updated via the rules $\Delta w^+ = \lambda \frac{\Phi'(\mu(x)) d^-}{(d^+ + d^-)^2}(x - w^+)$ and $\Delta w^- = \lambda \frac{\Phi'(\mu(x)) d^+}{(d^+ + d^-)^2}(x - w^-)$ where $w^+$ is the prototype of the true class and $w^-$ is the next prototype of another class, $d^+$ and $d^-$ are their corresponding distances to $x$. Sato et al. [10] suggest to set $\Phi(x) = \frac{1}{1+e^{-x}}$ and to use a relative distance for $\mu$:

$$\mu(x) = \frac{d^+(x) - d^-(x)}{d^+(x) + d^-(x)} \qquad (1)$$

Performing a prototype position update like above is equivalent to a stochastic gradient descent. We will use $\mu$ later on for the accuracy estimation.

## 3 Accuracy estimation of incrementally trained classifiers

In this contribution we want to answer the following question: "How can we best estimate current accuracy of an incremental learning classifier, taking into account previous learning sessions?" So we first use a standard incremental learning paradigm [7] to create an ensemble of incrementally improving classifiers. Then we train one regression model based on features computed on this ensemble.

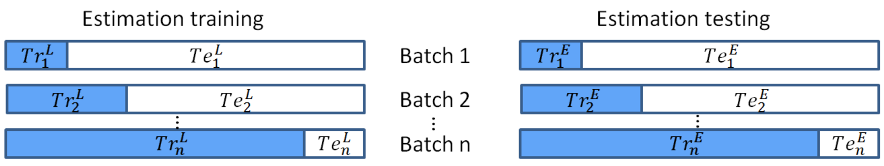

Fig. 1: Diagram describing the training and testing process for estimation.

Our approach is to estimate the accuracy by learning an offline model using statistics of the instance-based classifiers. In our evaluation we split up our data set $D$ into two equal sized sets $T^L$ for learning and $T^E$ for application of the estimation (Fig. 1). We first train a classifier $C^L$ on $T^L$ and then estimate the performance of a new classifier $C^E$ on $T^E$. The classifiers are trained in incremental batches with size $S_b$. While training $C^L$ the already trained samples are in $Tr^L$ and the yet unseen samples are in $Te^L$, where $T^L = Tr^L \cup Te^L$.

The idea behind our approach which we call Distogram Estimation (DGE) is to obtain a distribution of the unseen samples from $Te$ relative to the prototypes. We therefore redefine the relative distance from eq. 1 so that it is an unsupervised measure, where $d_+$ is the distance of a sample to the nearest prototype of any class and $d_-$ is the distance to the nearest prototype of another class. To use this information with a common regression approach, we generate a normalized histogram $h$ as statistics with $N_h$ representing the number of bins. The corresponding bin size can be calculated by $S_h = \frac{1}{N_h}$. The i'th bin $h_i$ is calculated by

$$h_i(X) = \sum_{x \in X} \theta(x,i) \quad where \quad \theta(x,i) = \begin{cases} 1 & if\ i * S_h \le |\mu(x)| < (i+1) * S_h \\ 0 & else \end{cases} \tag{2}$$

To compensate for data sets with different or increasing number of samples, we normalize the histogram so that $\sum_{i=1}^{N_h} h_i = 1$. The advantage of using the relative distance $\mu$ for building features is that its values lie between 0 and 1. Also we do not have to take into account the dimension of the feature space or the spread of the classes because the measure is only about relative distances of nearest winner and nearest prototype of another class. So we can compensate for classes of different sizes and distances.

We denote $C_k^L$ as the classifier trained after batch $k$ and our training set as $H = \{h^1, \ldots, h^n\}$ where $h^k = h(Tr_k^L)$. The accuracy of classifier $C$ on test set $Te$ is denoted as $a = Acc(C, Te)$ and respectively for each batch: $a_k = Acc(C_k, Te_k)$, where $Acc$ is the 0/1 accuracy.

After training the set of $C_k^L$, the accuracy estimator $E$ is trained with $(h_k, a_k^L)$. For $E$ we tested a neural net, Ridge Regression and Support Vector Regression. The neural net delivered best overall performance. We tested

$r \in \{1, 2, 3\}$ hidden layers with $s \in \{10, 30, 50\}$ neurons each. The best performing net on our data sets has architecture $r = 2$ and $s = 50, 30$. The output layer has one neuron which outputs the accuracy in a ratio from 0 to 1. The net is trained in 50 epochs. We used the Keras framework[1] with Tensorflow[2] backend to construct and train the neural nets.

During incremental training, the accuracy gets better and this effect becomes noticeable in the histograms. Please note that it is also important for estimation to include badly performing $C_k^L$ from the beginning. $C$ is trained $N_{ri}$ times with a shuffled $T^L$ for producing more training data for $E$. This improves the result because the initial state of the classifier is different, so we get more diverse classifiers. Next, we train $C^E$ on $T^E$ analog to $C^L$ to verify $E$. More precisely we train $C^E$ on $T^E$ and after each batch $k$ with size $S_b$ we calculate the absolute error $e_k^{abs} = abs(E(h_k) - a_k^E)$. After validation we calculate the mean absolute error with $mae = \frac{1}{n}\sum_k e_k^{abs}$ where $n$ is the number of batches. To compensate an inhomogeneous data distribution we repeat the whole process with a different $T^L$ and $T^E$ a number of times $N_{ro}$ and average the results.

| data set[3] | properties | | | mean absolute error (MAE) | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | GLVQ | | | kNN | | |
| | $n_s$ | $n_d$ | $n_c$ | *DGE* | *itt* | *cv* | *DGE* | *itt* | *cv* |
| MNIST(i) | 60,000 | 50 | 10 | 0.022 | 0.023 | **0.019** | **0.011** | 0.026 | 0.018 |
| MNIST(ii) | 60,000 | 50 | 10 | **0.015** | 0.214 | 0.324 | **0.008** | 0.137 | 0.263 |
| IRIS(i) | 150 | 4 | 3 | **0.066** | 0.154 | 0.095 | **0.065** | 0.158 | 0.097 |
| IRIS(ii) | 150 | 4 | 3 | **0.030** | 0.282 | 0.194 | **0.041** | 0.296 | 0.182 |
| CALTECH(i) | 8,677 | 4,096 | 101 | **0.024** | 0.038 | 0.043 | **0.016** | 0.038 | 0.044 |
| CALTECH(ii) | 8,677 | 4,096 | 101 | **0.020** | 0.199 | 0.319 | **0.016** | 0.220 | 0.326 |
| WDBC(i) | 569 | 9 | 2 | **0.018** | 0.053 | 0.028 | **0.019** | 0.055 | 0.032 |
| WDBC(ii) | 569 | 9 | 2 | **0.010** | 0.118 | 0.098 | **0.011** | 0.122 | 0.084 |
| OUTDOOR(i) | 5,000 | 4,096 | 50 | **0.022** | 0.048 | 0.041 | **0.012** | 0.047 | 0.072 |
| OUTDOOR(ii) | 5,000 | 4,096 | 50 | **0.020** | 0.192 | 0.281 | **0.008** | 0.127 | 0.193 |

Table 1: Evaluation of DGE approach compared to baseline models. Accuracy estimation was done after each batch. The table shows mean absolute error (MAE) compared to ground truth for i) random selection and ii) active learning.

## 4 Evaluation

We used five data sets for evaluation (Table 1). For WDBC and IRIS we did no preprocessing at all. For MNIST we used PCA for feature extraction. For CALTECH and OUTDOOR we used the VGG19 deep convolutional neural net [11] without the last softmax layer and with the weights from the image net competition as a feature representation. Also we compare two different strategies for selecting the samples from *Te* to train: i) random selection ii) active learning [12] by primarily selecting examples close to the decision boundary (where $d_-$ and $d_+$ are most similar).

The parameters for the experiments were $N_{ro} = 10$, $N_{ri} = 5$, $N_h = 15$, $S_b = 20$ and we denote the number of prototypes per class (in GLVQ experiments) as

[1]python deep learning library, URL https://keras.io/

[2]open-source software library for machine intelligence, URL https://tensorflow.org/

[3]data sets are taken from archive.ics.uci.edu, yann.lecun.com/exdb/mnist, and [5]

$N_w = 5$. The interleaved test/train error (itt) baseline is calculated over the last $itt^{ws}$ approaches, where $itt^{ws}$ is tuned for best performance on each data set. As a second baseline we choose a just 2 fold cross validation (cv) to compensate a small $Tr^E$ at early stages in training.

DGE has a smaller mean absolute error (MAE) compared to the baseline models in most cases. The results from GLVQ compared to kNN do not differ much. DGEs advantage becomes more noticeable in active querying experiments where the baseline methods are significantly worse.

## 5 Application Example

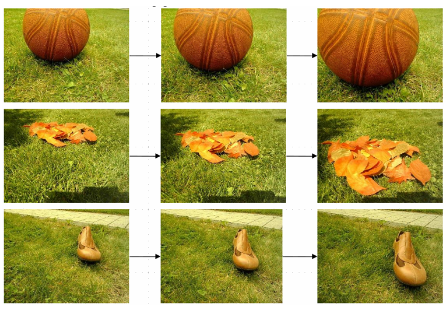

Fig. 2: Three objects from the outdoor data set with three images each. The first, fifth and last image of one approach is displayed. In total there are 50 different objects in the data set. Each object has 10 approaches from different object sides and light conditions. Each approach has 10 sequential images [5].

Let us demonstrate the applicability of the DGE approach for a challenging real-world incremental object recognition task as investigated in [5]. The target scenario is an incremental training of a mobile robot in a garden environment to enable recognition-dependent behavior. The training and testing data consists of 50 different objects, each covered by 10 robot approaches (10 images each) from different sides and light conditions (Fig. 2). The goal is the estimation of the expected recognition performance on unseen test data, based on the current training state.

Fig. 3 shows for random and active learning the comparison between the ground truth accuracy measured on unseen test data, the distogram estimate (DGE), cross-validation estimation using the training data (cv), and interleaved train/test error (itt). DGE clearly outperforms the baseline methods with an error of appr. 3%. DGE is especially better in the early stage of training. The baseline approaches need significantly longer for converging to the ground truth. Because mainly unconfident samples are requested while querying in active learning, the baseline approaches are generally too pessimistic in that case. DGE can adapt to the different placement of the prototypes.

## 6 Conclusion

We showed that our DGE method improves prediction of the accuracy of an incremental trained instance-based classifier for both random and active learning. It seems plausible that data distributions need to be similar for training and test conditions, excluding strong drift scenarios. It may be possible to extend

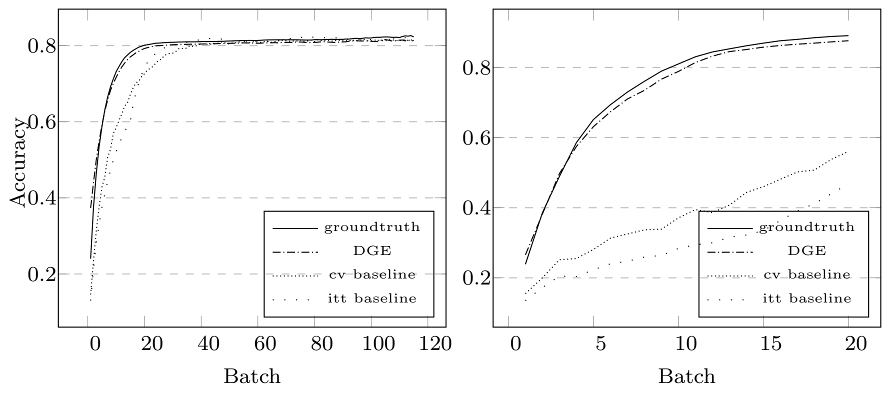

Fig. 3: Evaluation of incremental (left plot) and active sample selection (right plot). Distogram Estimation outperforms cv and itt.

this idea to other classifiers like support vector machines or neural nets. Our proposed DGE approach may be very useful in shared human-machine tasks, where the machine competence can be estimated to control work distribution between human and machine [13].

## References

[1] B. Wu, B. Hu, and H. Lin. A learning based optimal human robot collaboration with linear temporal logic constraints. *proc. CoRR*, abs/1706.00007, 2017.

[2] E. Platanios, A. Blum, and T. Mitchell. Estimating accuracy from unlabeled data. In *proc. Uncertainty in Artif. Intell. UAI*, pages 682–691, 2014.

[3] P. Donmez, G. Lebanon, and K. Balasubramanian. Unsupervised supervised learning I: estimating classification and regression errors without labels. *JMLR*, 11:1323–1351, 2010.

[4] G. Shafer and V. Vovk. A tutorial on conformal prediction. *Journal of Machine Learning Research*, 9:371–421, 2008.

[5] V. Losing, B. Hammer, and H. Wersing. Interactive online learning for obstacle classification on a mobile robot. In *proc. IJCNN*, pages 1–8, 2015.

[6] S. Kirstein, H. Wersing, and E. Körner. Rapid online learning of objects in a biologically motivated recognition architecture. In *proc. DAGM*, pages 301–308, 2005.

[7] V. Losing, B. Hammer, and H. Wersing. Incremental on-line learning: A review and comparison of state of the art algorithms. *Elsevier Neurocomputing*, 275:1261–1274, 2018.

[8] F. Schleif, B. Hammer, and T. Villmann. Margin-based active learning for LVQ networks. *Elsevier Neurocomputing*, 70:1215–1224, 2007.

[9] T. Kohonen. Improved versions of learning vector quantization. In *proc. IJCNN*, pages 545–550, 1990.

[10] A. Sato and K. Yamada. Generalized learning vector quantization. In *proc. NIPS*, pages 423–429, 1995.

[11] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *proc. CoRR*, 2014.

[12] B. Settles. Active learning literature survey. *University of Wisconsin–Madison*, 2009.

[13] M. Krüger, C. Wiebel, and H. Wersing. From tools towards cooperative assistants. In *proc. Conference on Human Agent Interaction, HAI*, pages 287–294, 2017.