

Georg Kreml  
Adrian Calma  
Bernhard Sick (Eds.)

Vincent Lemaire  
Andreas Holzinger

Daniel Kottke  
Robi Polikar

**IAL@ECML PKDD 2018**

**Workshop on  
Interactive Adaptive Learning**

Proceedings

The European Conference on Machine Learning and  
Principles and Practice of Knowledge Discovery in Databases  
(ECML PKDD 2018)

Dublin, Ireland, September 10, 2018

© 2018 for the individual papers by the papers' authors. Copying permitted for private and academic purposes. Re-publication of material from this volume requires permission by the copyright owners.

---

## Preface

Science, technology, and commerce increasingly recognize the importance of machine learning approaches for data-intensive, evidence-based decision making.

This is accompanied by increasing numbers of machine learning applications and volumes of data. Nevertheless, the capacities of processing systems or human supervisors or domain experts remain limited in real-world applications. Furthermore, many applications require fast reaction to new situations, which means that first predictive models need to be available even if little data is yet available. Therefore approaches are needed that optimize the whole learning process, including the interaction with human supervisors, processing systems, and data of various kind and at different timings: techniques for estimating the impact of additional resources (e.g. data) on the learning progress; techniques for the active selection of the information processed or queried; techniques for reusing knowledge across time, domains, or tasks, by identifying similarities and adaptation to changes between them; techniques for making use of different types of information, such as labeled or unlabeled data, constraints or domain knowledge. Such techniques are studied for example in the fields of adaptive, active, semi-supervised, and transfer learning. However, this is mostly done in separate lines of research, while combinations thereof in interactive and adaptive machine learning systems that are capable of operating under various constraints, and thereby address the immanent real-world challenges of volume, velocity and variability of data and data mining systems, are rarely reported.

Therefore, this workshop aims to bring together researchers and practitioners from these different areas, and to stimulate research in interactive and adaptive machine learning systems as a whole. This workshop aims at discussing techniques and approaches for optimizing the whole learning process, including the interaction with human supervisors, processing systems, and includes adaptive, active, semi-supervised, and transfer learning techniques, and combinations thereof in interactive and adaptive machine learning systems. Our objective is to bridge the communities researching and developing these techniques and systems in machine learning and data mining. Therefore we welcome contributions that present a novel problem setting, propose a novel approach, or report experience with the practical deployment of such a system and raise unsolved questions to the research community.

All in all, we accepted five regular papers (7 papers submitted) and six short papers (7 submitted) to be published in these workshop proceedings. The authors discuss approaches, identify challenges and gaps between active learning research and meaningful applications, as well as define new application-relevant research directions. We thank the authors for their submissions and the program committee for their hard work.

September 2018

Georg Kreml, Vincent Lemaire, Daniel Kottke  
Adrian Calma, Andreas Holzinger, Robi Polikar, Bernhard Sick

---

## **Organizing Committee**

Georg Kreml, Utrecht University  
Vincent Lemaire, Orange Labs France  
Daniel Kottke, University of Kassel  
Adrian Calma, University of Kassel  
Andreas Holzinger, Graz University of Technology  
Robi Polikar, Rowan University  
Bernhard Sick, University of Kassel

## **Program Committee**

Les Atlas, University of Washington  
Alexis Bondi, EDF R+D  
Lisheng Sun-Hosoya, LRI, University of Paris Saclay  
Marek Herde, University of Kassel  
Edwin Lughofer, University of Linz  
Rolf Würtz, University of Bochum  
Denis Huseljic, University of Kassel  
Luca Longo, Dublin Institute of Technology  
Freddy Lecue, Accenture Labs  
Jurek Stefanowski, Poznan University

## Table of Contents

|  |    |
|--|----|
| <b>Full Research Papers .....</b>  | 1  |
| On the Labeling Correctness in Computer Vision Datasets<br><i>Mohammed Al-Rawi and Dimosthenis Karatzas .....</i>  | 1  |
| Semi-supervised and Active Learning in Video Scene Classification<br>from Statistical Features<br><i>Tomáš Šabata, Petr Pulp and Martin Holena .....</i>                                       | 24 |
| Active Stream Learning with an Oracle of Unknown Availability for<br>Sentiment Prediction<br><i>Elson Serrao and Myra Spiliopoulou .....</i>   | 36 |
| ActivMetal: Algorithm Recommendation with Active Meta Learning<br><i>Lisheng Sun-Hosoya, Isabelle Guyon and Michele Sebag .....</i>  | 48 |
| Alignment-Based Topic Extraction Using Word Embedding<br><i>Tyler Newman and Paul Anderson .....</i>   | 60 |
| <b>Extended Abstracts .....</b>  | 73 |
| Human-Technology Relations in a Machine Learning Based Commuter<br>App<br><i>Lars Holmberg .....</i>   | 73 |
| An Interactive Learning Scenario for Real-time Environmental State<br>Estimation Based on Heterogeneous and Dynamic Sensor Systems<br><i>Agnes Tegen, Paul Davidsson and Jan Persson .....</i> | 77 |
| Adaptive Selection of Gaussian Process Model for Active Learning in<br>Expensive Optimization<br><i>Jakub Repický, Zbyněk Pitra and Martin Holena .....</i>                                    | 80 |
| Towards Interactive Feature Selection with Human-in-the-loop<br><i>Maialen Larranaga, Dimitra Gkorou, Thiago Guzella, Alexander Ypma, Faegheh Hasibi and Robert Jan van Wijk .....</i>         | 85 |
| Transfer of Knowledge for Surrogate Model Selection in Cost-Aware<br>Optimization<br><i>Zbyněk Pitra, Jakub Repický and Martin Holena .....</i>  | 89 |
| Embodiment Adaptation from Interactive Trajectory Preferences<br><i>Michael Walton, Ben Migliori and John Reeder .....</i>   | 95 |

---

# On the Labeling Correctness in Computer Vision Datasets

Mohammed Al-Rawi and Dimosthenis Karatzas

Computer Vision Center, Universidad Autonoma de Barcelona, Bellaterra, Spain  
al-rawi@cvc.uab.es

**Abstract.** Image datasets have heavily been used to build computer vision systems. These datasets are either manually or automatically labeled, which is a problem as both labeling methods are prone to errors. To investigate this problem, we use a majority voting ensemble that combines the results from several Convolutional Neural Networks (CNNs). Majority voting ensembles not only enhance the overall performance, but can also be used to estimate the confidence level of each sample. We also examined Softmax as another form to estimate posterior probability. We have designed various experiments with a range of different ensembles built from one or different, or temporal/snapshot CNNs, which have been trained multiple times stochastically. We analyzed CIFAR10, CIFAR100, EMNIST, and SVHN datasets and we found quite a few incorrect labels, both in the training and testing sets. We also present detailed confidence analysis on these datasets and we found that the ensemble is better than the Softmax when used estimate the per-sample confidence. This work thus proposes an approach that can be used to scrutinize and verify the labeling of computer vision datasets, which can later be applied to weakly/semi-supervised learning. We propose a measure, based on the Odds-Ratio, to quantify how many of these incorrectly classified labels are actually incorrectly labeled and how many of these are confusing. The proposed methods are easily scalable to larger datasets, like ImageNet, LSUN and SUN, as each CNN instance is trained for 60 epochs; or even faster, by implementing a temporal (snapshot) ensemble.

**Keywords:** Data annotation and labeling, ensembles, convolutional neural networks, semi-supervised learning

## 1 Introduction

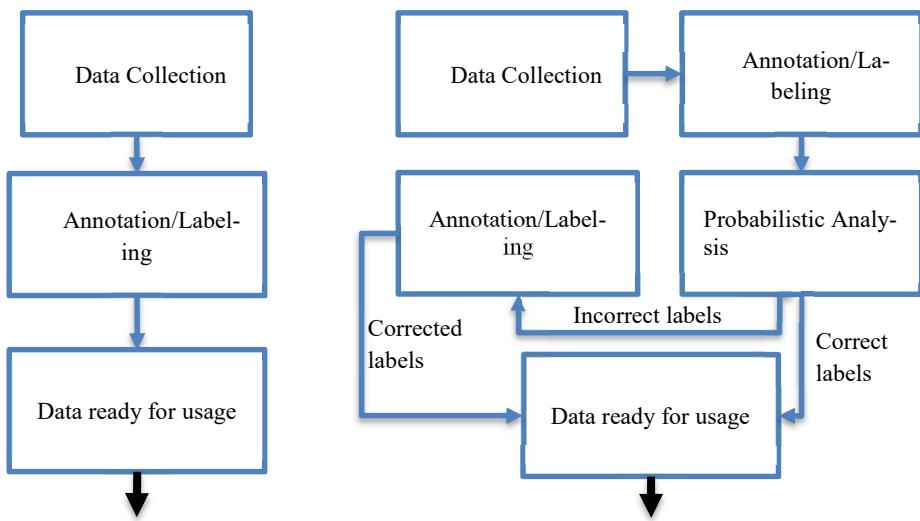
Recent developments in deep neural network approaches have greatly advanced the performance of visual recognition systems. Most research and development are based on standard computer vision datasets that have been annotated manually<sup>1</sup> or automatically. Moreover, the computer vision community is devoted to building larger datasets containing tens, or even hundreds, of millions of samples, for example the JFT-300M

---

<sup>1</sup> Amazon Mechanical Turk; Human intelligence through an API: <https://www.mturk.com/>

data [1]. Dataset annotation and/or labeling is a difficult, confusing and time consuming task; and even after labeling, it is difficult to assess a dataset for label correctness, whether manually or automatically. One way, however, to verify the labeling is by having a system that returns a confidence-level for each sample in the dataset, and not an overall system/classifier confidence, we illustrate the implementation of our ideas in **Fig. 1**.

Although state-of-the-art deep learning architectures can produce posterior probabilities, these probabilities may not be adequate to estimate the per-sample confidence-level value [2]. However, one promising approach that can be used to measure the per-sample confidence-level is by using ensemble classification methods. In ensemble learning, multiple classifiers can be combined to solve a specific classification task and they can be used to enhance the classification performance by compensating for the low performance of a poor classifier. Other important outcomes of ensemble learning include assigning a confidence-level, and/or posterior probability, to each sample in the testing set. Neural networks ensembles, nonetheless, have been investigated long before deep learning [3, 4]. After the deep learning boom in 2012, there has been quite a few works on ensembles built with deep nets deploying Convolutional Neural Networks (CNNs) [5-7]. Ensembles, in fact, can well be connected with deep learning frameworks and they are currently being used in many research and development aspects, including challenges and competitions [8].



**Fig. 1.** Data annotation that is normally used (left) and the proposed probabilistic analysis (right).

Ensembles' research work, however, have focused on improving the classification performance, on different applications and not only image understanding, compared to using a single learning model [5, 6, 9-11], and quite a few of them won computer vision

challenges, see for example [6]. While the ensemble performance-improvement hypothesis is effective and even supported by theoretical material, confidence analysis has not taken its expected share in the literature. Apart from this, and whenever compared to works tackling the overall confidence-level of the classifier, the confidence should go down to a low-level similar to humans' decision ability to be confident in their classification/decision for each sample/image. Such confidence analysis would highly be useful in weakly supervised learning, which was the goal of the work in [12], where the authors successfully implemented temporal ensembling. However, the authors of [12] have not examined the per-sample confidence levels to perceive how this could be useful in data cleaning, i.e. in a semi-supervised fashion. Other works that differentiate between expert and novice annotators, and between strongly and weakly annotated, in the so called active learning [13-15]. These works usually focus on uncertainty-based methods that usually ignore incorrectly labeled samples, and thus are sensitive to outliers [16, 17]. Furthermore, these works have not incorporated deep CNNs into active learning. The major aim therefore of this work is using deep CNNs to investigate the per-sample confidence level and compare it to the Softmax posterior probability, and to examine the possibility of using it to verify the labeling in computer vision datasets. The proposed approach can also find important application in weakly-supervised / active learning scenarios. We also aim to scrutinize the possibility of building ensemble classifier from one type of CNNs and compare the result to using different types of CNNs, including temporal ensembles, which will allow us to study the independence between the same kind of randomly trained CNN structures.

## 2 Methods

We tried different types of CNNs that have been trained with ImageNet (aka "Pre-Trained Models"). Generally speaking, a pre-trained model can learn the features from images faster than a model that starts from scratch (i.e. by randomly initializing its weights) [1]. In fact, some pre-trained models can reach an accuracy of 80% in three epochs on CIFAR10. For the ensemble classifier, we implemented voting schemes based on the predicted labels of the used classifiers. It has been proven that majority voting combination will always lead to a performance improvement for sufficiently large number of classifiers provided that the classifier outputs are independent [18]. To illustrate this further, consider a binary classifier and assuming that each classifier has a probability  $p$  of making a correct decision, the ensemble's probability ( $p_{ENS}$ ) of making a correct decision has a binomial distribution [19]:

$$p_{ENS} = \sum_{k=(M/2+1)}^M \binom{M}{k} p^k (1-p)^{M-k}, \quad (1)$$

where  $M$  is the number of classifiers used to build the ensemble. From the above, if  $p > 0.5$ ,  $p_{ENS} \rightarrow 1$  when  $M \rightarrow \infty$ . Note that  $p > 0.5$  (above chance-level) is almost present in most successfully trained binary classifiers. A similar argument can simply be conjectured for multiclass ensembles as combining binary classifiers for multi-class

classification is a very familiar approach [20]. The vital issue that can be of concern here is the independence of the output of different classifiers.

## 2.1 A measure to quantify the classified labels

In this work, we used majority voting ensemble based on the classifiers' output labels, and the ensemble chooses the category/class that receives the largest total vote. The higher the votes each sample gets, the higher the confidence and the lower the votes the lower the confidence. We then used the highest confidence as a key indicator to find any incorrectly labeled samples; which is the per-sample confidence level when the ensemble votes are equal to the number of classifiers used to build it. To make some inferences from the high confidence of the ensemble we make use of 1)  $N_{inco}$ , which is the number of incorrect samples that have been classified with high confidence (these are the false positives) with probability  $p_{inco} = (N_{inco}/N)$ , and 2)  $N_{corr}$ , which is the number of correct samples that have been classified with high confidence with probability  $p_{corr} = (N_{corr}/N)$ , where  $N$  is the number of testing samples. The value of  $p_{inco}$  is of most interest as it indicates that all classifiers of the ensemble agreed (with high confidence) to incorrectly classify a sample. The high-confidence incorrectly classified samples will further be investigated to verify their labels. It is also possible that these incorrectly classified samples contain some of the difficult / confusing details that deceived the ensemble, or the classifiers that were used to build the ensemble were not independent. To compare the performance of different ensembles, we will calculate the Odds Ratio (OR) using the formula [21]:

$$OR = p_{inco}(1 - p_{corr})/(p_{corr}(1 - p_{inco})). \quad (2)$$

The value of OR will be used to estimate the likeliness that the ensemble may produce false positives but with high confidence (on the assumption that all samples are correctly labeled/annotated), i.e. how likely the incorrect samples will be classified as correct ones with high confidence; hence, the lower the OR the better. An OR equals to one indicates that the classification of correct and incorrect samples with high confidence is equally likely to occur.

We used CIFAR10 and CIFAR100 [22] in all confidence analyses' experiments. CIFAR10 is a well-known dataset that has heavily been investigated in the computer vision literature. It essentially has 50K  $32 \times 32$  RGB image samples for training and 10K  $32 \times 32$  RGB image samples for testing, where each image belongs to one of ten classes; airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. Each class thus has 5000 images in the training set and 1000 images in the testing set. CIFAR100, on the other hand, has a similar image structure but it has 100 classes distributed on 600 samples and 500 samples in the training set and 100 samples in the testing set. To implement our algorithms, we used PyTorch [23] as our main deep learning framework. Further details on the used methods and experimental setting can be found in the supplemental material.

We chose the VGG CNN family [24] (we will refer to VGG Ensemble; 'VGG-E') as they require less training time than other CNNs, and they can reach higher accuracy than other CNNs, when trained up to 60 epochs. We chose 60 epochs for the following

reasons: 1) to see how fast and how well the ensemble classifier can learn with confidence 2) to reduce the execution time of ensembles, 3) following the Schapire’s idea on the strength of weak learnability [25], and 4) for the proposed methodology to be efficient and scalable when used on larger data sets. Some VGGs have Batch Normalization (BN) others do not, thus they have been postfixed with ‘BN’, VGG11BN thus denotes VGG11 with batch normalization. In most analysis, we used eight VGG CNNs, these are: VGG11, VGG11BN, VGG13, VGG13BN, VGG16, VGG16BN, VGG19, VGG19BN.

### 3 Results

#### 3.1 The Softmax Posterior Probability

It is widely known that the CNNs, and neural networks in general, yield Posterior Probabilities (PPs) as their outputs, when Softmax is used. It is not known, however, if these posterior probabilities can be used to estimate the per-sample confidence to an accurate degree. To investigate the confidence distribution of the correctly and incorrectly classified labels via Softmax outputs, we used CIFAR100 to train VGG19BN with the previously mentioned settings except that we increased the number of training epochs to 600. We will refer to the condition where Softmax posterior probability equals one as high confidence. The typical situation of the PP of the incorrectly classified samples is to have an exponential distribution or, in the worst case, a normal distribution. The results of the training and testing are demonstrated in **Table 1** and show that Softmax posterior probabilities of a single VGG have high OR values, and thus, may not be used as good estimates of the per-sample confidence level. This deduction is clearly depicted in **Fig. 2** that shows the posterior probability distributions, where the incorrectly classified samples have a peek at PP=1 (PP=1 denotes high confidence as the probability is 100%). We also perceive from **Fig. 2** that the distribution of the PP values is right-skewed for the incorrect labels, and this means using these PP values for the per-sample confidence level is not reliable. The presented Softmax results, in fact, copes with the neural networks posterior probabilities as being over-confidence estimates, as has been detailed in [2]. Our probability analysis provides further evidence of why adversarial attacks [26] are possible when using Softmax to state the confidence of the classified object/image.

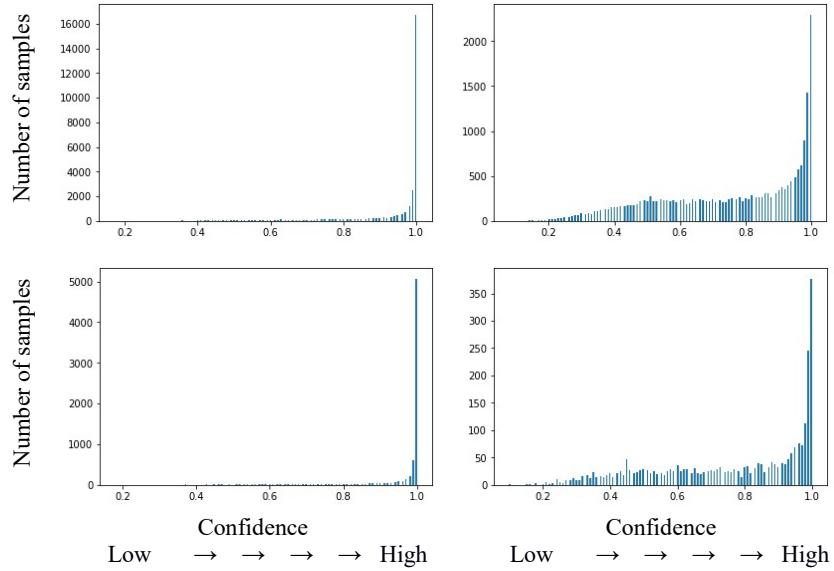
#### 3.2 Ensembles Built with Different VGG Types

Using CIFAR10, each VGG type was trained for up to 16 times, VGG-E thus has a total of 118 VGGs (Skipping chance-level local minima resulted sometimes in less than the planned  $16 \times 8 = 128$  VGGs). The results of the discovered images with incorrect labels in the testing set are presented in **Table 2**. Our tests showed that there are 9 incorrect samples with high-confidence (voting is equal to the number of VGGs used to build the ensemble). Investigating the 9 false positives, we found that most of them have incorrectly been labeled in the testing set of CIFAR10. We also present in the

supplementary material a few samples that have high per-sample confidence level values but were incorrectly classified. After examination, however, these samples appear to be confusing.

**Table 1.** VGG19BN output as posterior probability computed with Softmax, trained for 600 epochs.

|                 | Training set size | Testing set size | Classification accuracy % | #Incorrectly classified at PP=1 | #Correctly classified at PP=1 | OR    |
|-----------------|-------------------|------------------|---------------------------|---------------------------------|-------------------------------|-------|
| <b>CIFAR10</b>  | 50K               | 10K              | 93.0                      | 248                             | 8702                          | 0.004 |
|                 | 10K               | 50K              | 86.2                      | 2957                            | 39419                         | 0.017 |
| <b>CIFAR100</b> | 50K               | 10K              | 72.8                      | 377                             | 5072                          | 0.038 |
|                 | 10K               | 50K              | 58.8                      | 2295                            | 16733                         | 0.095 |



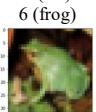
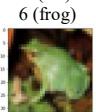
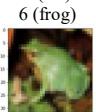
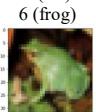
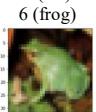
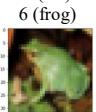
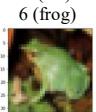
**Fig. 2.** VGG19BN Softmax posterior probability distribution of the correctly classified labels (left column) and the incorrectly classified labels (right column); training with 10K and testing with 50K samples of CIFAR100 (top row) and training with 50K and testing with 10K samples of CIFAR100 (bottom row).

To investigate correctness of the training set labels, we revert the training and testing datasets that are used to train the VGGs. In this case, we used CIFAR10 testing set (10K samples) for building the ensemble classifier and CIFAR10 training set (50K) for testing it. After training, the VGG-E has a total of 126 different VGGs. This, in principle,

is more challenging than the previous experiment, and could be useful in weakly supervised learning. The results of the discovered incorrect labels are presented in **Table 2**. By investigating the 81 false positives, we found that some have incorrectly been labeled in the training set of CIFAR10, but some images have confusing content. A few samples discovered by the VGG-E are not only be confusing CNNs, but also to the human observer, see the supplementary material. We repeated the same above experiments on Cifar100, which resulted in a VGG-E with 128 VGGs. The ensemble enhanced the accuracy by ~9% compared to the average of VGGs. A few incorrect labels in CIFAR100’s testing, as well in the training set, are demonstrated in supplemental material. The analysis of these ensembles are summarized in **Table 3**, and the per-sample confidence distributions are shown in **Fig. 3**.

We can see from **Table 2** that the frog (which has an index 2405 in the data) is labeled as a cat in CIFAR10 testing set, but the VGG-E managed to predict the correct label (more results are shown in the supplemental material). The amount of incorrect labels in CIFAR100 is higher, for example, a bottle (which has an index of 7762) is labeled as a cup, other images with incorrect labeling also exist. Nonetheless, by inspecting these images, one can admire the work that CNNs can achieve in classifying these CIFAR images, as most of the times the details are not clear even for the human observer, due to using the so called tiny images (as each image has a size of  $32 \times 32$ ). Thus, using CNNs ensembles would assist inspecting and verifying the labeling, as proposed in this work.

**Table 2.** Some Incorrect Labels Discovered by VGG-E in CIFAR10.

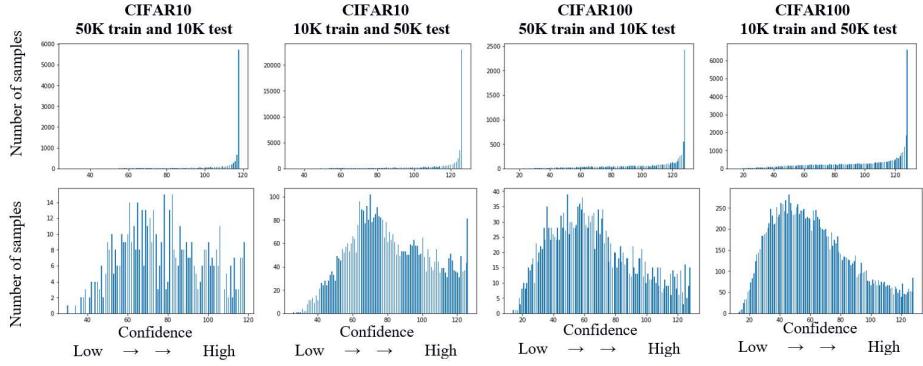
|  |  |  |  |  |   |   |
|--|--|--|--|--|---|---|
| <b>Testing set</b><br>2405*<br>3 (cat)<br>6 (frog)<br>  | 9227<br>1 (car)<br>9(truck)<br>   | 3309<br>4 (deer)<br>7(horse)<br>  | 3560<br>1 (car)<br>9(truck)<br> | 5141<br>6(frog)<br>3 (cat)<br>    | 9227<br>1(car)<br>9(truck)<br>   | 7311<br>1 (car)<br>9 (truck)<br> |
| <b>Training set</b><br>5747<br>1 (car)<br>9 (truck)<br> | 35103<br>6 (frog)<br>4 (deer)<br> | 30814<br>1 (car)<br>9 (truck)<br> | 6319<br>3 (cat)<br>4 (deer)<br> | 33079<br>5 (dog)<br>7 (horse)<br> | 7008<br>2 (bird)<br>4 (deer)<br> | 8803<br>3(cat)<br>5(dog)<br>     |

\* Index (The Index of the image in CIFAR10); Original label; Predicted label; and Image of Predicted label.

### 3.3 Experiments with the EMNIST dataset

The same experimental strategy used for CIFAR10&100 has been implemented on the EMNIST dataset [27]. The EMNIST dataset, which is derived from the NIST Special Database, has been compiled from a set of handwritten English characters and Arabic digits and has been suggested as a more challenging replacement to the MNIST dataset. The EMNIST ‘By Class’ split has 814,255 images distributed over 62 unbalanced classes. Pixel image format and dataset structure that directly matches the MNIST dataset,

each image is 28x28 gray-level. EMNIST is extremely challenging, on the labeling and testing levels, as it has upper and lower case confusion, in addition to numeral value one (1) versus letter (lower case L; l), O versus 0/zero, 9 versus q, etc. In fact, our analysis shows that this confusion has been present at the labeling/annotation stage.

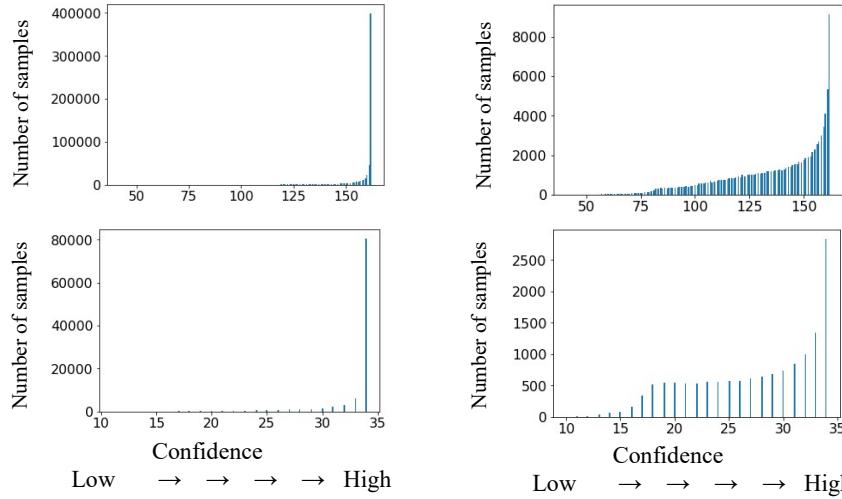


**Fig. 3.** Per-sample confidence distribution, correctly classified labels (top row) and incorrectly classified labels (bottom row) in CIFAR10 and CIFAR100.

**Table 3.** VGG-E using 16 classifier instances of each VGG type, a total of 128 VGGS to build the VGG-E.

|                 | Training set size | Testing set size | Average accuracy over VGGS % | Accuracy of VGG-E % | #Confidently classified, but incorrect | #Confidently classified and correct | OR     |
|-----------------|-------------------|------------------|------------------------------|---------------------|--|-------------------------------------|--------|
| <b>CIFAR10</b>  | 50K               | 10K              | $90.84 \pm 0.95$             | 94.2                | 9                                      | 5734                                | 0.0006 |
|                 | 10K               | 50K              | $86.10 \pm 1.07$             | 90.2                | 81                                     | 23024                               | 0.002  |
| <b>CIFAR100</b> | 50K               | 10K              | $69.5 \pm 0.14$              | 78.2                | 15                                     | 2446                                | 0.005  |
|                 | 10K               | 50K              | $57.06 \pm 1.5$              | 67.06               | 84                                     | 6687                                | 0.01   |

As for the results, the ensemble gave a classification accuracy 0.87 when trained using the training set. Furthermore, in the testing set, the incorrectly labeled samples that got recognized by the ensemble, with high confidence, is 2,837, while the correct samples that got recognized by the ensemble, with high confidence, is 83,467, and the quantitative measure OR is 0.0098. To inspect the labeling of the training set, we trained the ensemble with the testing, which gave classification accuracy of 0.85. The number of incorrect samples that got recognized by ensemble with confidence is 9,154, the correct samples that got recognized by the ensemble with high confidence is 408,588, and the quantitative measure OR is 0.0094. The confidence distributions are illustrated in **Fig. 4**. Due to space limitations, incorrect/confusing EMNIST images are demonstrated in the supplemental material.



**Fig. 4.** Per-sample confidence distribution training set (top row) and testing set (bottom row); correctly classified labels (left) and incorrectly classified labels (right) in EMNIST dataset.

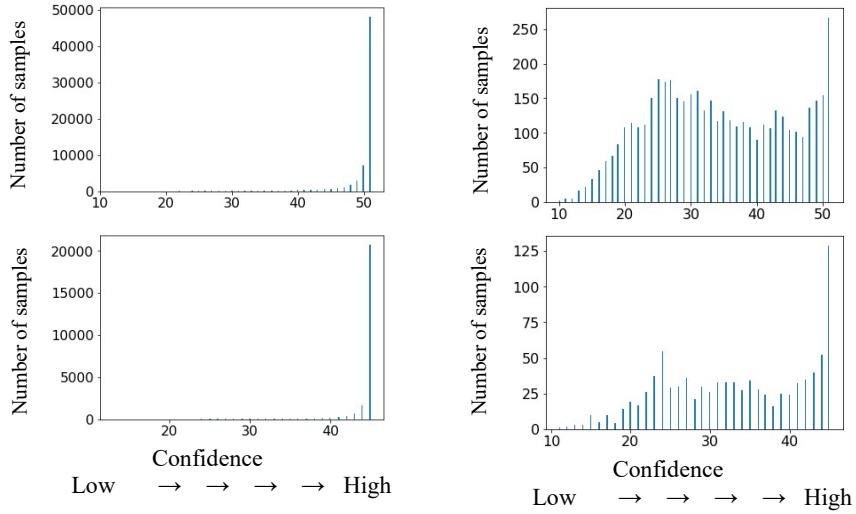
### 3.4 Experiments with the SVHN dataset

SVHN [28] is a real-world image digit dataset that has been inspired by MNIST structure (e.g., the images are of small cropped digits) but comes from a significantly harder, unsolved, real world problem (recognizing digits and numbers in natural scene images). SVHN, which contains 73257 digits for training and 26032 digits for testing, has been obtained from house numbers in Google Street View images. Using the training set for training, the classification accuracy of the ensemble is 95.14. The number of incorrect samples that got recognized by all the CNNs is 129 and the number of correctly-labeled samples that got recognized by the ensemble is 20887, yielding OR= 0.0012. Inspecting the label correctness in the training set showed that the number of incorrect labels that got recognized by the ensemble with high confidence is 267 with OR= 0.0018 (classification accuracy of the ensemble is 93.69). The confidence distributions are illustrated in **Fig. 5**. Due to space limitations, selected incorrect/confusing SVHN images that have been detected with our approach are demonstrated in the supplemental material.

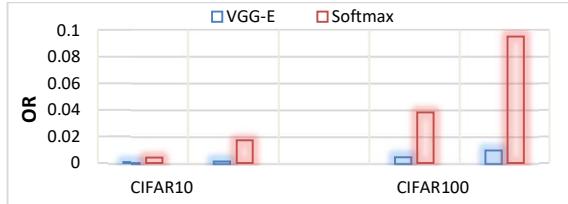
## 4 Conclusion

It is of high interest in computer vision to have a system that can conjecture with confidence what is wrong and what is right, i.e., to confidently guess which labels are correctly and/or incorrectly classified. This work is a step in that direction. This paper presents the use of CNN ensembles to detect incorrect labels in image classification datasets. Essentially, if the ensemble is confident on a result which is incorrect, either the sample is indeed visually confusing or it was incorrectly labelled. Probabilistic con-

fidence analyses showed that some images with incorrect labeling and confusing content exist. **Fig. 6** summarizes the results of CIFAR10 & 100 and illustrates that the OR values of Softmax posterior probabilities are higher than the OR values of the ensemble posterior probabilities; the lower the OR values the better. Hence, the posterior probability of a CNN, measured with Softmax, cannot be used to accurately estimate the per-sample confidence level. Furthermore, the proposed OR analysis provided a novel evidence that batch normalization increases the ensemble confidence, thus, could be related to improving generalization.



**Fig. 5.** Per-sample confidence distribution training set (top row) and testing set (bottom row); correctly classified labels (left) and incorrectly classified labels (right) in SVHN dataset



**Fig. 6.** VGG-E (ensemble) versus Softmax posterior probability.

Our analyses also agreed with previous ensemble works as the overall accuracy has been increased by around 5%, 9%, 2%, 5% for CIFAR10, CIFAR100, SVHN, and EMNIST respectively. Based on the proposed probabilistic methods and by making use of the snapshot ensemble (supplemental material), we are currently building a labeling verification tool to be implemented in PyTorch framework. This tool will be useful not only in labeling verification, but can also be used in semi-supervised and active learning applications. Evaluations on other datasets are left for future work.

## References

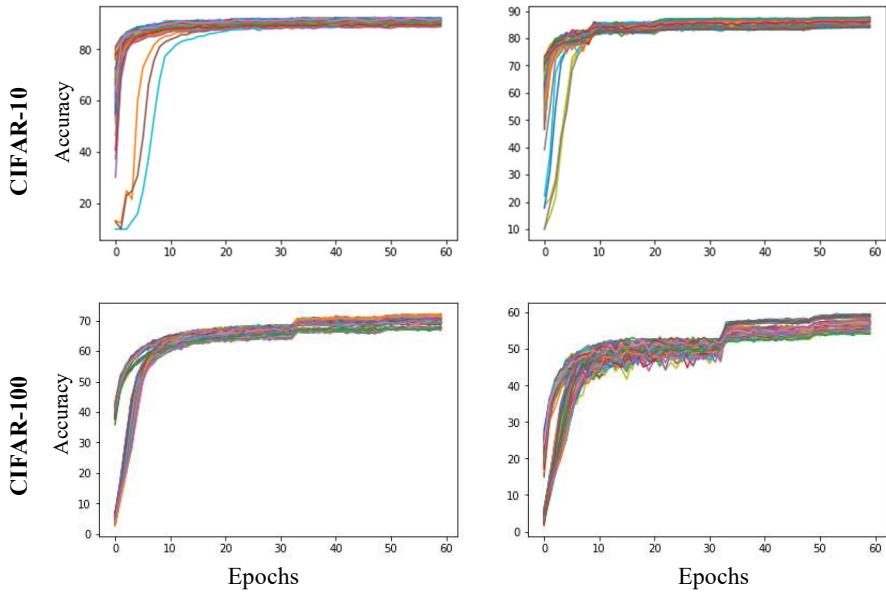
1. Sun, C., et al. *Revisiting Unreasonable Effectiveness of Data in Deep Learning Era.* in *16th IEEE International Conference on Computer Vision (ICCV).* 2017. Venice, ITALY.
2. Ju, C., A. Bibaut, and M. J. van der Laan, *The Relative Performance of Ensemble Methods with Deep Convolutional Neural Networks for Image Classification.* 2017, <http://arxiv.org/abs/1704.01664>.
3. Hansen, L.K. and P. Salamon, *Neural Network Ensembles.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 1990. **12**(10): p. 993-1001.
4. LeCun, Y., Y. Bengio, and G. Hinton, *Deep learning.* Nature, 2015. **521**(7553): p. 436-444.
5. Chen, J.L., et al. *An Ensemble of Convolutional Neural Networks for Image Classification Based on LSTM.* in *2017 International Conference on Green Informatics (ICGI).* 2017.
6. Ding, C.X. and D.C. Tao, *Trunk-Branch Ensemble Convolutional Neural Networks for Video-Based Face Recognition.* IEEE Transactions on Pattern Analysis and Machine Intelligence, 2018. **40**(4): p. 1002-1014.
7. Schmidhuber, J., *Deep learning in neural networks: An overview.* Neural Networks, 2015. **61**: p. 85-117.
8. Minetto, R., M. Pamplona Segundo, and S. Sarkar. *Hydra: an Ensemble of Convolutional Neural Networks for Geospatial Land Classification.* in <https://arxiv.org/abs/1802.03518>. 2018.
9. Chen, G.B., et al. *Ensemble Application of Convolutional and Recurrent Neural Networks for Multi-label Text Categorization.* in *International Joint Conference on Neural Networks (IJCNN).* 2017. Anchorage, AK.
10. Duan, M.X., K.L. Li, and K.Q. Li, *An Ensemble CNN2ELM for Age Estimation.* IEEE Transactions on Information Forensics and Security, 2018. **13**(3): p. 758-
11. Diaz-Vico, D., et al., *Deep Neural Networks for Wind and Solar Energy Prediction.* Neural Processing Letters, 2017. **46**(3): p. 829-844.
12. Laine, S. and T. Aila, *Temporal Ensembling for Semi-Supervised Learning,* in *International Conference on Learning Representations (ICLR).* 2017.
13. Yang, Y.Z. and M. Loog, *A variance maximization criterion for active learning.* Pattern Recognition, 2018. **78**: p. 358-370.
14. Reyes, O., A.H. Altalhi, and S. Ventura, *Statistical comparisons of active learning strategies over multiple datasets.* Knowledge-Based Systems, 2018. **145**: p. 274-288.
15. Wang, K.Z., et al., *Cost-Effective Active Learning for Deep Image Classification.* IEEE Transactions on Circuits and Systems for Video Technology, 2017. **27**(12): p. 2591-2600.
16. Freund, Y., et al., *Selective sampling using the query by committee algorithm.* Machine Learning, 1997. **28**(2-3): p. 133-168.
17. Burbridge, R., J.J. Rowland, and R.D. King, *Active learning for regression based on query by committee.* Intelligent Data Engineering and Automated Learning - Ideal 2007, 2007. **4881**: p. 209-218.
18. Bishop, C.M., *Pattern Recognition and Machine Learning (Information Science and Statistics).* 2006: Springer-Verlag New York, Inc. .

19. Monteith, K., et al. *Turning Bayesian Model Averaging Into Bayesian Model Combination*. in *International Joint Conference on Neural Networks (IJCNN)*. 2011.
20. Shiraishi, Y. and K. Fukumizu, *Statistical approaches to combining binary classifiers for multi-class classification*. Neurocomputing, 2011. **74**(5): p. 680-688.
21. Rao, P.S., *Proportions, Odds Ratios and Relative Risks*, in *Statistical Methodologies with Medical Applications*. 2017, Wiley.
22. Krizhevsky, A., *Learning Multiple Layers of Features from Tiny Images*, *Technical Report*. 2009, <http://www.cs.toronto.edu/~kriz/cifar.html>: Canadian Institute for Advanced Research.
23. Paszke, A., et al. *Automatic differentiation in PyTorch*. in *NIPS 2017 Autodiff Workshop*. 2017.
24. Simonyan, K. and A. Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. in *International Conference on Learning Representations*. 2015.
25. Schapire, R.E., *The Strength of Weak Learnability*. Machine Learning, 1990. **5**(2): p. 197-227.
26. Li, X. and F.X. Li. *Adversarial Examples Detection in Deep Networks with Convolutional Filter Statistics*. in *16th IEEE International Conference on Computer Vision (ICCV)*. 2017. Venice, ITALY.
27. Cohen, G., et al., *EMNIST: an extension of MNIST to handwritten letters*. (2017): <https://www.nist.gov/itl/iad/image-group/emnist-dataset> <http://arxiv.org/abs/1702.05373>.
28. Netzer, Y., et al., *Reading Digits in Natural Images with Unsupervised Feature Learning*, in *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*. 2011: <http://ufldl.stanford.edu/housenumbers/>.

## Supplemental Material

### Experimental Setting

The following parameters have been used in all experiments: dropout probability is 0.2; maximum number of epochs is 60; learning rate is 0.01 (the learning rate is set to decrease by half according to the following milestones = {8, 20, 48}; unless mentioned otherwise); momentum=0.95; the seed was randomly pulled from the time function; weight-decay=0.0005; Nestrove momentum was used; SGD optimizer; 100 mini batches, random shuffling enabled, and Cross Entropy Loss. The run/training, however, was skipped if the CNN is stuck at a chance-level local minima (~10% for CIFAR10 and ~1% for CIFAR100), and a new training instance is launched with a new random seed. To demonstrate the possible variations in training each CNN of the ensemble, we present the training progress of various VGGs in Fig-Sup. 1.



**Fig-Sup. 1. Variations of the training progress of the different ensembles built from CIFAR10 and CIFAR100 data. Training with 50K and testing with 10K (left column) and training with 10K and testing with 50K (right column).**

In our preliminary analysis, we built ensembles using different CNN architectures; including, different types of ResNets\*, VGGs\* DualPathNets\* (DPNs), DenseNets\*, NasNetLarge, etc. However, we chose to build the ensembles via the VGG net family

as they require less training time than the other CNN types, they give similar performance to the ensemble built from different CNN architectures, and they result much higher accuracy than other CNNs, when trained up to 60 epochs. To give an example, NasNetLarge requires 9X times the training time of VGG11 and 5X times of VGG19BN. The classification accuracy of NasNetLarge gets to 75% compared to above 85% for all VGG types, when trained up to 60 epochs. To clarify further, for a maximum of 60 epochs, VGG11 reaches 85% accuracy in less than 6 minutes, while ResNet18 gets to 78% accuracy in 7 minutes, but NasNetLarge gets to 75% accuracy in 71 minutes. In general, the DPN, SqueezeNet, and ResNet (including Resnext\*) families are slower than VGGs and/or can get less than 80% accuracy in 60 epochs.

### Ensembles Built with a Single VGG Type

In this experiment, we used 16 classifier instances to see how do they perform compared to using different VGG classifiers. The training has been performed with the 10K testing set, and the testing has been performed using the 50 training set, as it is more challenging than using the sets in training the other way around. Table 4 summarizes the results. From Table 4 we notice that Batch-Normalization always leads to better confidence, when the same CNN is used, as the OR is less when using BN, that is it is less likely to have false positives with high confidence when BN is used.

**Table 4. VGG-E using 16 classifier instances of each VGG type; using 10K for training and 50K for testing of CIFAR10.**

| VGG-E          | Average accuracy over VGGs % | Accuracy of VGG-E % | #Confidently classified, but incorrect | #Confidently classified and correct | OR     |
|----------------|------------------------------|---------------------|--|-------------------------------------|--------|
| <b>VGG19BN</b> | $86.78 \pm 0.11$             | 90                  | 424                                    | 33073                               | 0.0043 |
| <b>VGG16BN</b> | $87.48 \pm 0.15$             | 91                  | 446                                    | 33892                               | 0.0042 |
| <b>VGG13BN</b> | $86.95 \pm 0.11$             | 90                  | 466                                    | 33559                               | 0.0046 |
| <b>VGG11BN</b> | $84.79 \pm 0.12$             | 88                  | 556                                    | 31506                               | 0.0066 |
| <b>VGG19</b>   | $86.27 \pm 0.33$             | 89                  | 607                                    | 33657                               | 0.0059 |
| <b>VGG16</b>   | $86.31 \pm 0.22$             | 89                  | 651                                    | 34120                               | 0.0061 |
| <b>VGG13</b>   | $86.09 \pm 0.13$             | 89                  | 706                                    | 34156                               | 0.0076 |
| <b>VGG11</b>   | $84.11 \pm 0.15$             | 87                  | 834                                    | 32379                               | 0.0092 |

From Table 4 we see that VGG11 has the weakest performance compared to other VGGs, thus, we took this experiment further to build one VGG-E using 128 VGG11s and another using 128 VGG13BN using CIFAR10 testing set for training. The VGG-E increased the performance by ~5%, but the confidence levels, as shown in the OR values, are better when using different VGG models than using only one VGG model, as

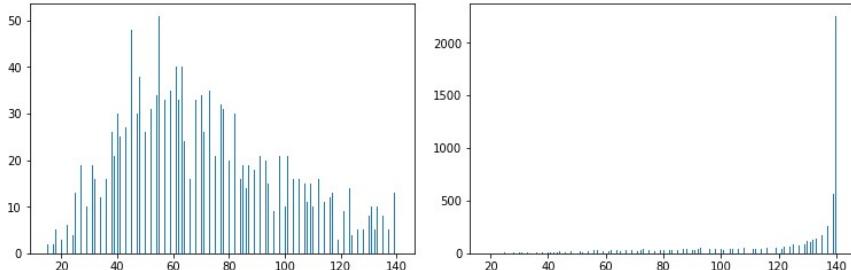
shown in Table 5. Thus, a VGG-E built using 128 VGGs results, given by the OR values, are not as good as a VGG-E built using different types of VGGs.

**Table 5. VGG-E with 126 VGGs of the same type (CIFAR10); using 10K for training and 50K for testing.**

|                | Mean accuracy over VGGs % | Accuracy of VGG-E % | #Confidently classified, but incorrect | #Confidently classified and correct | OR     |
|----------------|---------------------------|---------------------|--|-------------------------------------|--------|
| <b>VGG13BN</b> | $87.50 \pm 0.13$          | 90.9                | 116                                    | 26569                               | 0.0020 |
| <b>VGG11</b>   | $87.48 \pm 0.15$          | 87.3                | 229                                    | 25277                               | 0.0045 |

### Temporal (snapshot) Ensemble (VGG-ET)

We used VGG19BN to build a temporal ensemble for CIFAR100; training with 50K and testing with 10K. In this case, each epoch resulted a classifier. We used 150 epochs and neglected the results of the first ten epochs, as we opted for the training to reach a state of stability. Similar to VGG-E, VGG-ET was able to determine quite a few incorrect labels, and to produce descent per-sample confidence values. The VGG-ET reached an accuracy of 76.8% (slightly lower than VGG-E), and an OR (at high confidence) of 0.004. Thus, this snapshot/temporal ensemble could be used instead of an ensemble built from the different CNN architectures, which can be used to build a fast and efficient labeling verification tool, which is a future work we are trying. The confidence distributions are demonstrated in Fig-Sup. 2.



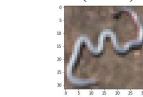
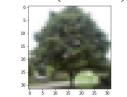
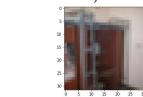
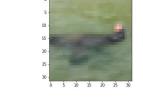
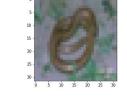
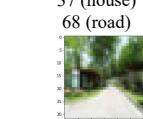
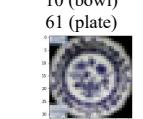
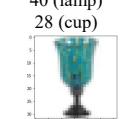
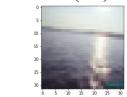
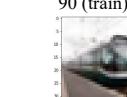
**Fig-Sup. 2. Per-sample confidence distribution, incorrectly classified labels (left) and correctly classified labels (right), using temporal (snapshot) ensemble VGG-ET (VGG19BN).**

### Extended Results

In the tables below, we demonstrate using a few samples *that we have selected from the incorrect labeled ones detected by the probability analysis*. The labels of the samples and the corresponding predicted labels, along with the corresponding image, are shown. To double check the incorrectness, by third parties, the readers of this article

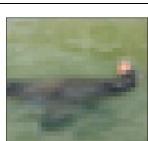
may use the index of the sample to examine it in the dataset, i.e. by loading the image and the corresponding label. The tables contain data from CIFAR10, CIFAR100, SVHN, and EMNIST datasets.

**Table 6. Selected incorrectly labeled images detected in CIFAR100 training set; predicted with high confidence**

|              |   |   |   |   |   |   |
|--------------|---|---|---|---|---|---|
| Testing set  | 7762<br>28 (cup)<br>9 (bottle)  | 5764<br>99 (worm)<br>78 (snake)   | 9601<br>33 (forest)<br>49 (mountain)  | 6927<br>92 (tulip)<br>54 (orchid)   | 8951<br>47 (maple tree)<br>52 (oak tree)  | 4460<br>59 (pine tree)<br>52 (oak tree)   |
|              |    |    |    |    |    |    |
|              | 1557<br>10 (bowl)<br>28 (cup)   | 8071<br>5 (bed)<br>94 (wardrobe)  | 1100*<br>72 (seal)<br>55 (otter)  | 2172<br>99 (worm)<br>78 (snake)   | 9298<br>17 (castle)<br>37 (house)   | 1357<br>96 (willow tree)<br>52 (oak tree)   |
|              |    |    |    |    |    |    |
|              | 687<br>59 (pine tree)<br>56 (palm tree)   | 780<br>37 (house)<br>68 (road)  | 7006<br>10 (bowl)<br>61 (plate)   | 12455<br>40 (lamp)<br>28 (cup)  | 10178<br>42 (leopard)<br>88 (tiger)   | 39441<br>11 (boy)<br>2 (baby)   |
|              |   |   |   |   |   |   |
| Training Set | 32814<br>90 (train)<br>81 (streetcar)   | 47936<br>12 (bridge)<br>76 (skyscraper)   | 35209<br>76 (sky-scraper)<br>69 (rocket)  | 23509<br>71 (sea)<br>60 (plain)   | 16305<br>60 (plane)<br>71 (sea)   | 22395<br>81 (street-car)<br>90 (train)  |
|              |  |  |  |  |  |  |

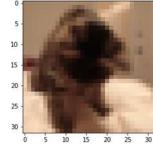
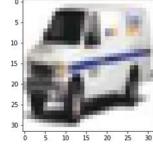
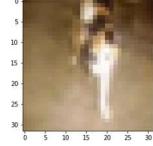
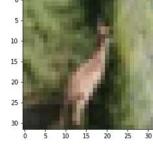
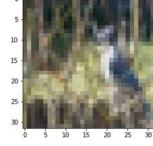
\* We found the same image in the training test with index 24083 but has the label 55 (otter). So not only the same image was included in both training and testing sets, but with an incorrect/opposite label.

**Table 7. Selected incorrectly labeled images detected in CIFAR100 testing set; predicted with high confidence**

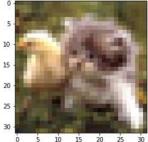
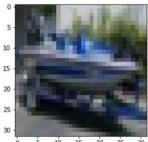
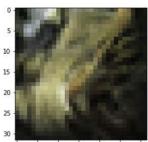
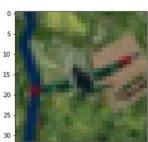
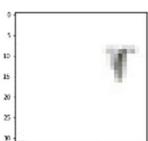
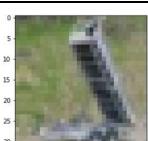
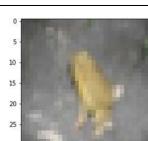
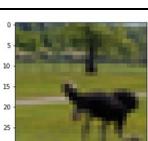
| Index | Original label (class) | Predicted label (class) | Image  | Remarks |
|-------|------------------------|-------------------------|--|---------|
| 1214  | 45(lobster)            | 26(crab)                |    |         |
| 5278  | 81(streetcar)          | 13(bus)                 |    |         |
| 6799  | 8(bicycle)             | 48(motorcycle)          |    |         |
| 6927  | 92(tulip)              | 54(orchid)              |   |         |
| 1100  | 72(seal)               | 55(otter)               |  |         |
| 7429  | 70(rose)               | 68(road)                |  |         |
| 7762  | 28(cup)                | 9(bottle)               |  |         |
| 5873  | 50(mouse)              | 74(shrew)               |  |         |

### Confusing Images Detected in CIFAR10

**Table 8.** Selected confusing images detected in CIFAR10 test set; predicted with high confidence

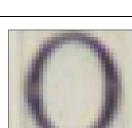
| Index | Original label<br>(class) | Predicted<br>label (class) | Image   | Remarks   |
|-------|---------------------------|----------------------------|---|---|
| 811   | 3 (cat)                   | 5 (dog)                    |    | Hard to tell what this is!  |
| 5416  | 9 (truck)                 | 1 (car)                    |    | This is a minivan, probably looks more like a car than a truck                          |
| 7099  | 3 (cat)                   | 5 (dog)                    |   | Probably the label is correct, but the tail is dominating the photo                     |
| 4794  | 4 (deer)                  | 2 (bird)                   |  | Difficult to infer which one is deer and which one is bird, even for the human observer |
| 9832  | 2 (bird)                  | 4 (deer)                   |  | Difficult to infer which one is deer and which one is bird, even for the human observer |
| 9503  | 2 (bird)                  | 4 (deer)                   |  | Difficult to infer which one is deer and which one is bird, even for the human observer |

**Table 9. Selected confusing images detected in CIFAR10 training set; high confidence pred**

| Index | Original label<br>(class) | Predicted la-<br>bel (class) | Image   | Remarks  |
|-------|---------------------------|------------------------------|---|--|
| 32085 | 3 (cat)                   | 5 (dog)                      |    | Bird and cat in one picture! Clas-<br>sified as dog                                  |
| 13694 | 8 (ship)                  | 1 (car)                      |    | Boat on top of a pull cart with<br>wheels identified as car                          |
| 43283 | 3 (cat)                   | 6 (frog)                     |    | The image is not clear, probably<br>of cat category, but classified as<br>frog       |
| 9119  | 0 (plane)                 | 4 (deer)                     |   | The image should be of category<br>plane, yet it is not clear; classified<br>as deer |
| 5867  | 2 (bird)                  | 0 (plane)                    |  | A very confusing object  |
| 36288 | 9 (truck)                 | 2 (bird)                     |  | A truck with a ladder is hard to<br>identify as a truck                              |
| 36788 | 6 (frog)                  | 3 (cat)                      |  | A frog that is hard to tell for the<br>human observer                                |
| 34305 | 2 (bird)                  | 4 (deer)                     |  | Something that does not look<br>very much like a bird has been<br>identified as deer |

### Experiments with SVHN

**Table 10.** Selected incorrectly labeled images detected in SVHN training set; predicted with high confidence

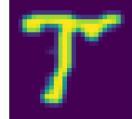
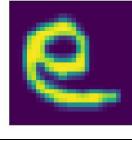
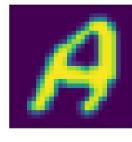
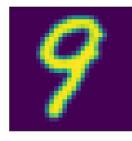
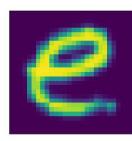
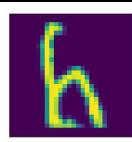
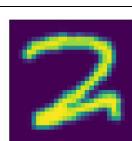
| Index | Original label (class) | Predicted label (class) | Image   | Remarks   |
|-------|------------------------|-------------------------|---|---|
| 3692  | 9                      | 8                       |    |   |
| 6290  | 9                      | 5                       |    |   |
| 6291  | 5                      | 9                       |    |   |
| 6875  | 1                      | 2                       |  |   |
| 6960  | 0                      | 7                       |  | This image has two digits, although there should only be one. It was labeled with zero, but the ensemble got the other digit correctly with high confidence (7) |
| 9502  | 2                      | 0                       |  |   |
| 9503  | 0                      | 1                       |  |   |
| 17666 | 3                      | 1                       |  | As each image should only have one digit, this image has been incorrectly segmented and labeled with 3, the ensemble labeled it as 1 with high confidence.      |

**Table 11.** Selected incorrectly labeled images detected in SVHN testing set; predicted with high confidence

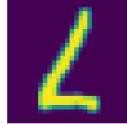
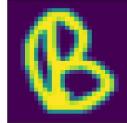
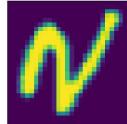
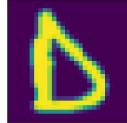
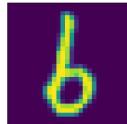
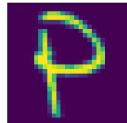
| Index | Original label<br>(class) | Predicted la-<br>bel (class) | Image   | Remarks   |
|-------|---------------------------|------------------------------|---|---|
| 1317  | 1                         | 5                            |    | The image is incorrectly segmented, as it should have only one digit.<br>Interestingly, the attention of the CNN is brought to the center |
| 3916  | 1                         | 6                            |    | Partially occluded with 5, but the ensemble got it correctly as 6. The original labels was incorrect with a value of 1.                   |
| 5397  | 0                         | 3                            |    |   |
| 6500  | 1                         | 2                            |   |   |
| 8250  | 1                         | 5                            |  |   |
| 11844 | 5                         | 1                            |  |   |
| 14678 | 7                         | 2                            |  |   |
| 14526 | 1                         | 8                            |  |   |

### Experiments with EMNIST

**Table 12.** Selected incorrectly labeled images detected in EMNIST training set; predicted with high confidence

| Index | Original label<br>(class) | Predicted<br>label (class) | Image   | Remarks |
|-------|---------------------------|----------------------------|---|---------|
| 104   | t                         | T                          |    |         |
| 13980 | F                         | e                          |    |         |
| 14283 | a                         | A                          |   |         |
| 15830 | g                         | 9                          |  |         |
| 18892 | r                         | e                          |  |         |
| 19781 | 6                         | h                          |  |         |
| 21248 | T                         | 7                          |  |         |
| 29785 | z                         | 2                          |  |         |

**Table 13. Selected incorrectly labeled images detected in EMNIST testing set; predicted with high confidence**

| Index | Original label (class) | Predicted la-<br>bel (class) | Image   | Remarks |
|-------|------------------------|------------------------------|---|---------|
| 2371  | (lower case L; l)      | L                            |    |         |
| 2202  | b                      | B                            |    |         |
| 2127  | n                      | N                            |    |         |
| 2618  | g                      | 9                            |   |         |
| 2640  | B                      | D                            |  |         |
| 2820  | b                      | h                            |  |         |
| 3369  | 6                      | b                            |  |         |
| 3515  | r                      | P                            |  |         |

---

# Semi-supervised and Active Learning in Video Scene Classification from Statistical Features

Tomáš Šabata<sup>1</sup>, Petr Pulc<sup>2</sup>, and Martin Holeňa<sup>2</sup>

<sup>1</sup> Faculty of Information Technology, Czech Technical University in Prague,  
Prague, Czech Republic

[tomas.sabata@fit.cvut.cz](mailto:tomas.sabata@fit.cvut.cz)

<sup>2</sup> Institute of Computer Science of the Czech Academy of Sciences,  
Prague, Czech republic  
[{pulc,martin}@cs.cas.cz](mailto:{pulc,martin}@cs.cas.cz)

**Abstract.** In multimedia classification, the background is usually considered an unwanted part of input data and is often modeled only to be removed in later processing. Contrary to that, we believe that a background model (i.e., the scene in which the picture or video shot is taken) should be included as an essential feature for both indexing and follow-up content processing. Information about image background, however, is not usually the main target in the labeling process and the number of annotated samples is very limited.

Therefore, we propose to use a combination of semi-supervised and active learning to improve the performance of our scene classifier, specifically a combination of self-training with uncertainty sampling. As a result, we utilize a combination of statistical features extractor, a feed-forward neural network and support vector machine classifier, which consistently achieves higher accuracy on less diverse data. With the proposed approach, we are currently able to achieve precision over 80% on a dataset trained on a single series of a popular TV show.

**Keywords:** video data, scene classification, semi-supervised learning, active learning, colour statistics, feedforward neural networks

## 1 Introduction

Automatic multimedia content labeling is still a comparatively difficult domain for machine learning. High input data dimensionality requires large training data sets, especially for approaches that are designed without prior assumptions on the data properties.

Moreover, the increasing resolution of image sensors brings higher detail (and thus, at least in theory, more information), but poses a significant issue for training phases of virtually all machine learning algorithms.

Many approaches, therefore, have to introduce a trade-off concerning the number of involved parameters, the number of distinct output labels (classes) [26] and the resolution of the input imagery [7]. Alternatively, they have to use only the statistical properties of the input data (as [3] and many others).

We also need to tackle the limitation on the amount of labeled training data.

Recent trends in video content processing include a task usually called Video to Text. The primary objective of such processing is to take multimedia content and describe its main features in a human-comprehensible text. Such representation may contain gathered information on the scene, actors, objects and actions in which they are involved. Such as the single image description “baseball player is throwing ball in game,” as presented in [12].

Current approaches, however, commonly omit the information concerning the visual appearance of the background in complex multimedia content – even though such information might provide substantial contextual information for the object detection and event description itself. Approaches that use neural networks are mostly data-driven and require large amounts of data to adapt to each selected class. This requirement is, however, seldom met in smaller multimedia collections, such as home video, university lecture recordings, movie studios or corporate media databases.

We also want to reflect that a particular scene can be recalled by a human from a couple of static frames. Therefore, manual scene labeling is a relatively easy task as opposed to event labeling that may need the full video sequence or object labeling that commonly requires drawing a bounding box around the annotated object.

To use the limited human involvement in scene labeling as efficiently as possible, we employ semi-supervised learning to allow making use of unlabeled data, which are substantially easier to obtain, whereas simultaneously selecting the data for annotation using active learning methods.

The rest of this paper is organized as follows: In Section 2, we briefly summarize the state of the art in scene classification in the context of single images without significant obstruction by foreground objects, as well as the state of the art in combining semi-supervised learning (SL) and active learning (AL). Section 3 describes our approach to scene recognition in video content. In Section 4, we compare the accuracy of our method for different approaches to feature selection and different classifiers.

## 2 State of the Art

Scene recognition is rather simple from the human perspective. Whether the scene is the same as one previously visited is recognized by the overall layout of the space, presence, and distribution of distinct objects, their texture, and color. Other sensory organs can provide even more information and allow faster recall. Scenes not visited beforehand may fall after a thorough exploration into one of broader categories based on similarity of such features.

Multimedia content, however, does not allow such space exploration directly. It is constrained to the color information of individual pixels at a rather small resolution. Video content resolves this issue only partially with a motion of the camera, which, on the other hand, introduces more degrees of freedom in background modeling and increases its complexity.

## 2.1 Single Image Scene Classifiers Based on Colour Statistics

The early scene classifiers, including the Indoor/Outdoor problem [22,27], and also the more recent approaches mentioned below are directly based on the overall color information contained in the picture. The vital decision in this particular case is the selection of color space and the granularity of the considered histograms.

RGB (red, green and blue components) is the primary color space of multimedia acquisition and processing. However, it does not directly encode the quality of the color perceived by a human. By qualities of color, we primarily mean the color shade (hue). In HSV encoding (hue, saturation and value of the black/white range components, the last of them related to the overall lightness of the color), hue is commonly sampled with finer precision (narrower bins in histogram approaches) than saturation and lightness [5,8].

Mainly because of memory consumption and model size, statistical features of the individual images are commonly used for image processing, including basic scene classification. Other approaches are based on object detection [11,15], on interest point description [3,2], or in recent years they use deep convolutional neural networks [26,29,32].

## 2.2 Multi-label Extension

Often, a single image contains multiple semantic features – such as sea, beach and mountains. A crisp classification into only one class would, however, have to take only the dominant class, which might be different from the selection of the annotator. A somewhat possible extension is to create a new crisp class for each encountered combination of the labels, but this would have a substantial impact in the areas where the amount of labeled content is not sufficient to enable proper training on such sub-classes.

Another possibility is to organize the labels into a hierarchical structure. If the described scenery shares multiple features, the parent label may be preferred for content description. When the scene classifier detects only a specific part of the scenery, we should not consider it a full miss.

**Statistical approach** One of common assumptions in scene classification is that, during a single shot, the background will be visible for a more extended period than the foreground object. Therefore, we may process each frame in a single shot by a scene recognition algorithm and vote among the proposed labels. The statistical approach to background modeling applies if we assume a static camera shot. When such an assumption is met, all frames are perfectly aligned, and the background model can be extracted from the long-term pixel averages.

## 2.3 Semi-supervised Learning and Active Learning

Semi-supervised learning (cf. the survey [33]) is a technique that benefits from making use of easily obtainable unlabeled data for training. In this paper, we

mainly focus on the self-training approach to semi-supervised learning [10]. It is a simple and efficient method, in which we add samples with the most confidently predicted labels (pseudo-labels) to the training dataset. This can be done so the model is retrained in each iteration. Other approaches to semi-supervised learning are co-training [1] and multiview training [9] that benefit from agreement among multiple learners.

Active learning (cf. the survey [23]) is related to semi-supervised learning through being also used in machine learning problems where obtaining unlabeled data is cheap and manual labeling is expensive but possible. Its goal is to spend a given annotation budget only on the most informative instances of the unlabeled data. Most commonly, it is performed as *pool-based sampling* [14], assuming a small set of labeled data and a large set of unlabeled data. Samples that were found to be the most informative, are given to an annotator and are moved into the labeled set. The considered machine learning model (e.g., a classifier) is retrained and the algorithm iterates until the budget is exhausted or the performance of the model is satisfactory.

Pool based sampling needs to evaluate an *utility function* that estimates some kind of usefulness of knowing the label of a particular sample. There are various ways of defining the utility function: for example, as a measure of uncertainty in *uncertainty sampling* [13], as a number of disagreements within an ensemble of diverse models in a method called *query-by-committee* [25], as the expected model change [24], the expected error [20] or only the variance part of the model error [6].

Semi-supervised and active learning can be quite naturally combined since they address unlabeled data set from opposite ends. For example, self-training uses the most certain samples to be turned to labeled samples and uncertainty sampling queries the most uncertain samples and obtains its label from an annotator. Such a combination was used for various problems [16,21,31]. Successful combinations with active learning exist also for multiview training [17,18,30].

### **3 Multimedia Histogram Processing with Feed-Forward Neural Network using SVM**

In the reported research, our main concern is to enable an automatic annotation of small datasets with a generally small variation within the individual classes. For example, we are not particularly interested in recognition of a broader scenery concept (such as a living room), but we aim at the classification that the video shot was captured in one specific living room.

One of the possible applications, on which we will demonstrate our approach in the next section, is the classification of individual scenes in long-running shows and sit-coms. However, our approach is designed to be versatile and enable, for example, disambiguation of individual television news studios or well-known sites.

Another concern of us is that the training of the classifier should require a minimal amount of resources to enable connection into more complex systems

of multimedia content description as a simple high-level scene disambiguation module.

Therefore, we revise the traditional approaches in scene classification and propose the use of color histograms, possibly with partial spatial awareness. To demonstrate our reasoning behind this step, we refer to Figure 1.

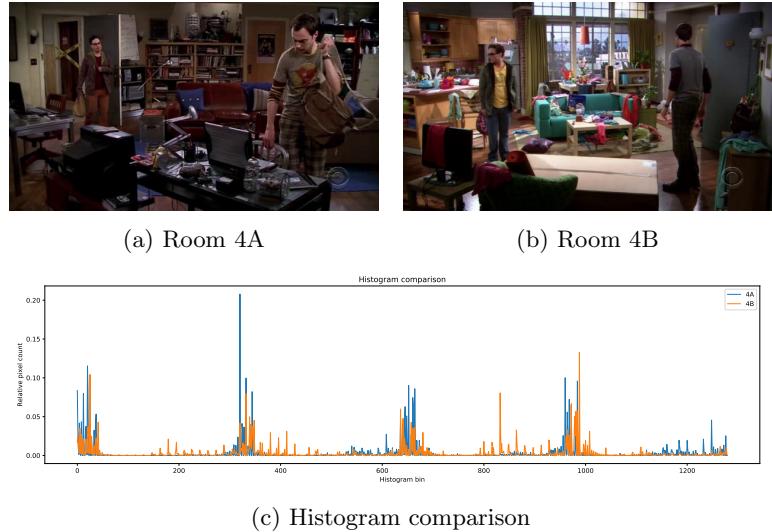


Fig. 1: Representative frames from two distinct living rooms and comparison of the proposed histograms. Although both of these pictures depict a living room, the distribution of colours is different. Source images courtesy of CBS Entertainment.

We choose a feed-forward neural network as the base classifier. In particular, we use a network with two hidden layers of 100 and 50 neurons and logistic sigmoid as activation function. The output layer uses the softmax activation function. The network is trained using backpropagation with a negative log-likelihood loss function and a stochastic gradient descent optimizer. The network topology, activation function and optimizer were found through a simple grid search, in which we considered also other activation functions such as ReLU or hyperbolic tangent, and another optimizer, based on an adaptive estimate of first and second moments of the gradients [?].

For the scene classification task, we can use the trained neural network directly. However, we introduce an improvement inspired by transfer learning. Transfer learning is usually used in deep convolutional neural nets where the convergence of all parameters is slower [28]. However, we would like to demonstrate, that the transfer learning can bring a substantial benefit also in shallow neu-

ral networks. Especially in combination with a support vector machine (SVM) classifier.

In our scenario, we freeze the parameters of first layers and use the network as a feature extractor. For the classification stage, the original softmax layer is then replaced with a linear support vector machine. This brings us a rather small but consistent improvement in the final accuracy.

For an overall structure of our proposed network, please refer to Figure 2. In the figure, red arrows represent the first learning phase in which parameters of the net are found using a backpropagation. Blue arrows represent the second learning phase – transfer learning. In the second phase, the first two layers of the already trained neural net are used for training dataset generation. After that, a linear SVM classifier is trained. Green arrows represent the prediction of new samples.

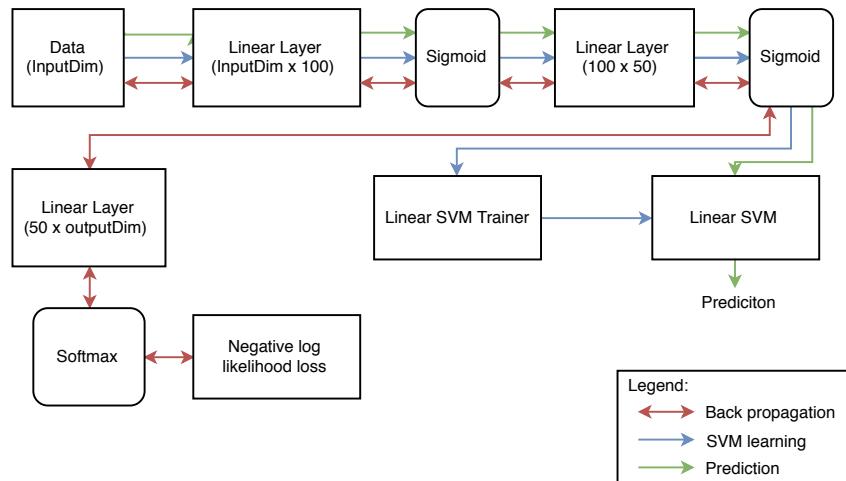


Fig. 2: The architecture of the proposed neural net. Red arrows represent the first learning phase; blue arrows represent a second learning phase with SVM and green arrows represent the prediction phase.

Finally, the model performance was improved by using a combination SL+AL. We have chosen a combination of uncertainty sampling with pseudo-labeling through self-training. In the experimental evaluation, the utility functions least uncertain (eq. 1), margin (eq. 2) and entropy (eq. 3) were included.

$$\phi_{LC}(x) = P_\theta(y_1^*|x), \quad (1)$$

$$\phi_M(x) = P_\theta(y_1^*|x) - P_\theta(y_2^*|x)), \quad (2)$$

$$\phi_E(x) = - \sum_i^N P_\theta(y_i|x) \log P_\theta(y_i|x), \quad (3)$$

In each iteration,  $n$  samples with the lowest utility function were queried to be annotated. At the same time, samples with the utility function higher than a threshold were predicted using the current version of the model, and these predictions were then used to train the next version of the model. Utility functions were calculated from the output of softmax layer of the neural net. The number of samples  $n$  was chosen to be 5 in each iteration. The threshold value was tuned to keep the number of wrong labels getting into training data as low as possible.

### 3.1 Weighted accuracy

The scene description in our experiment is constructed hierarchically so there are three different levels of the label. The first level describes building name, the second level describes a room, and the last level describes detail in the room. For instance, if the camera shot captures the whole living room of the flat “4A” in the “main” building, we use a label such as `main.4a`. If only a specific portion of the room is shown, we use a more detail level of the label such as `main.4a.couch`.

To take into account the label hierarchy, we introduce weighted accuracy of a classifier  $F$  predicting  $\hat{y}_1, \dots, \hat{y}_n$  for training data  $(x_1, y_1), \dots, (x_n, y_n)$ :

$$\text{WA}(F) = \frac{1}{n} \sum_{i=1}^n f(y_i, \hat{y}_i),$$

$$f(y_i, \hat{y}_i) = \begin{cases} 1 & \text{if } \mathbb{W}(y_i = \hat{y}_i, 3) \\ 0.5 & \text{if } \mathbb{W}(y_i = \hat{y}_i, 2), \\ 0 & \text{otherwise.} \end{cases}$$

where  $\mathbb{W}(y_i = \hat{y}_i, k)$  is the truth function of equality of all components of  $y_i$  and  $\hat{y}_i$  on the  $k$ -th or a higher level of the component hierarchy.

## 4 Experimental evaluation

For the evaluation of all the following approaches, we prepared our dataset [19] from the first series of a sit-com The Big Bang Theory. This particular show uses only a couple of scenes and by 2018 new series are still being produced. The dataset is chosen for the proof of concept experiment and new datasets should follow in future experiments. The multimedia content was automatically

segmented into individual camera shots by PySceneDetect [4] using the content detector.

A middle frame from the detected shot was stored as a reference for human annotation and convolutional neural network processing. Due to the copyright protection, these stored frames are not contained in the dataset. They were divided into 80% training and 20% test data along the time axis.

For statistical approach experiments, the following histograms averaged by the respective frame area and shot duration were obtained: *RGB 8x8x8* (flattened histogram over  $8 \times 8 \times 8$  bins), *H* (hue histogram with 180 bins), *HSV* (concatenation of 180 bins H, 256 bins S and 256 bins V histograms) and *HSV 20x4x4 2\*2* (flattened histogram over  $20 \times 4 \times 4$  bins in each of 4 parts of the frame introduced by its prior division in  $2 \times 2$  grid).

#### 4.1 Combinations of histograms and classifiers

We have compared combinations of the above described histograms with the following classifiers: linear SVM,  $k$  nearest neighbours ( $k$ -NN), naive Bayes (NB) and the feedforward neural nets (FNNs) described in section 3, i.e., FNN alone and FNN+SVM. A full comparison of the unweighted accuracy of all 16 combinations is carried out in Table 1.

Table 1: Accuracy of combining the considered four kinds of histograms with the following classifiers: linear SVM,  $k$ -NN, NB, FNN and FNN+SVM. For each classifier, the highest accuracy with respect to the different kinds of histograms is in *italics*, and the highest accuracy with respect to different classifiers is in **bold**

| Accuracy [%]          | Linear SVM  | <i>k</i> -NN | NB          | FNN  | FNN+SVM     |
|-----------------------|-------------|--------------|-------------|------|-------------|
| RGB 8x8x8             | 18.1        | 32.42        | 26.1        | 54.5 | <b>60.0</b> |
| <i>H</i>              | 12.4        | 30.7         | 26.1        | 56.0 | <b>58.9</b> |
| HSV                   | 14.1        | 32.6         | 32.4        | 63.3 | <b>65.7</b> |
| <i>HSV 20x4x4 2*2</i> | <i>46.0</i> | <i>45.4</i>  | <i>33.9</i> | 77.2 | <b>78.8</b> |

It is noticeable that HSV 20x4x4 2\*2 feature dominates over all other variants. Therefore, we were using HSV 20x4x4 2\*2 in the subsequent experiments. On the other hand, adding an SVM as the last layer of the FNN brings only a smaller improvement.

#### 4.2 Comparison with an inception style neural network

State-of-the-art approaches in image scene classification usually use the residual deep convolutional neural networks with inception-style layers. They are typically combined with multi-scale processing of the input imagery.

With these key features in mind, we used the winner of the 2016 LSUN challenge [29] as the reference method for scene classification on our dataset.

The results are, however, worse than expected. The accuracy progress (see Figure 3) shows that the network training is very unstable. The testing accuracy achieves a maximum of 32.4% in the 801st epoch.

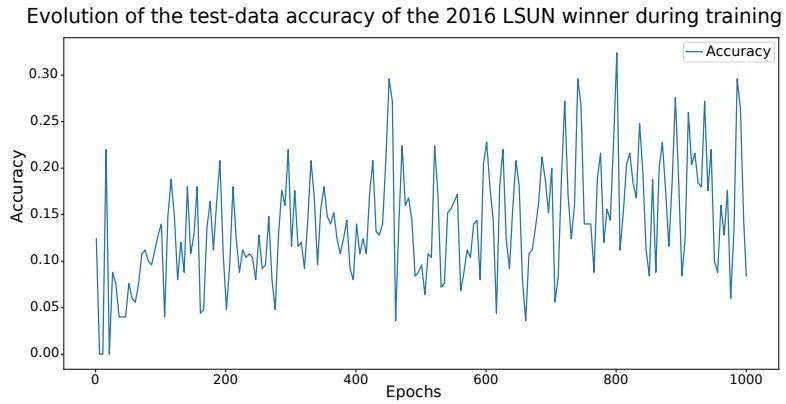


Fig. 3: Accuracy of the inception-style winner of the LSUN challenge [29] on the testing set

As we are unable to interpret the inner state of the neural network directly, we may only assume that the main issue with using the multi-resolution convolutional neural network is the small dataset size. However, this is exactly the issue we need to mitigate.

#### 4.3 Including supervised and active learning

As was shown in Subsection 4.1, the use of feed-forward neural network itself brings a substantial increase in classification metrics. As Table 2 indicates, the SVM layer provides an additional improvement as well as using part of the unlabeled dataset with SL+AL. Although the improvement is not high, we believe that using the more sophisticated combination of SL+AL could bring us even further.

The initial labeled dataset contained 5315 samples. An unlabeled dataset with 26528 samples was used for both active and semi-supervised learning. A human annotator was asked five queries at each of ten iterations.

Table 2: Final achieved accuracy, weighted accuracy, precision, Recall and F1 score with the HSV 20x4x4 2\*2 histogram. For each of these classifier performance measures, the highest value among the considered classifiers is in bold

|                    | Acc           | Weighted acc  | Precision     | Recall        | F1            |
|--------------------|---------------|---------------|---------------|---------------|---------------|
| FNN                | 0.7723        | 0.8518        | 0.7813        | 0.7590        | 0.7578        |
| FNN-SVM            | 0.7883        | <b>0.8626</b> | 0.8026        | 0.7837        | 0.7842        |
| FNN-SVM with SL+AL | <b>0.7895</b> | 0.8617        | <b>0.8037</b> | <b>0.8022</b> | <b>0.7978</b> |

## 5 Conclusions and Future Work

In this paper, we sketched how semi-supervised learning combined with active learning can be applied to scene recognition. In addition, we propose to use neural networks for further feature enhancement.

The resulting features extracted from the proposed neural network provide a substantial improvement over the engineered features on input. Especially, if the extracted features are used as a data embedding for a linear SVM classifier.

This allows us to achieve an accuracy of almost 79% on a small dataset that is significantly higher than reference method (32.4%).

Several descriptors are, however, still hard to recognize even for a human annotator (e.g. staircase floor number). In these situations, one may benefit from the context of the previous and following shot and consequently improve the classification accuracy. Therefore, we would like to try context-based classifiers, such as HMM, CRF or BI-LSTM-CRF as a next step of our research.

Last but not least, we would like to use transductive SVM in the top layer of the final classifier and provide further experiments in the combination with semi-supervised and active learning, primarily with active multiview training.

## Acknowledgements

The reported research has been supported by the grant 18-18080S of the Czech Science Foundation (GAČR).

## References

1. Blum, A., Mitchell, T.: Combining labeled and unlabeled data with co-training. In: Proceedings of the eleventh annual conference on Computational learning theory. pp. 92–100. ACM (1998)
2. Bosch, A., Zisserman, A., Muñoz, X.: Scene classification via plsa. In: European conference on computer vision. pp. 517–530. Springer (2006)
3. Bosch, A., Zisserman, A., Muñoz, X.: Scene classification using a hybrid generative/discriminative approach. IEEE transactions on pattern analysis and machine intelligence **30**(4), 712–727 (2008)
4. Castellano, B.: Pyscenedetect. <https://github.com/Breakthrough/PySceneDetect> (2017)

5. Chen, L.H., Lai, Y.C., Liao, H.Y.M.: Movie scene segmentation using background information. *Pattern Recognition* **41**(3), 1056–1065 (2008)
6. Cohn, D., Atlas, L., Ladner, R.: Improving generalization with active learning. *Machine learning* **15**(2), 201–221 (1994)
7. Deng, J., Dong, W., Socher, R., Li, L.J., Li, K., Fei-Fei, L.: Imagenet: A large-scale hierarchical image database. In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. pp. 248–255. IEEE (2009)
8. Fan, J., Elmagarmid, A.K., Zhu, X., Aref, W.G., Wu, L.: Classview: hierarchical video shot classification, indexing, and accessing. *IEEE Transactions on Multimedia* **6**(1), 70–86 (2004)
9. Farquhar, J., Hardoon, D., Meng, H., Shawe-taylor, J.S., Szemek, S.: Two view learning: Svm-2k, theory and practice. In: *Advances in neural information processing systems*. pp. 355–362 (2006)
10. Grandvalet, Y., Bengio, Y.: Semi-supervised learning by entropy minimization. In: *Advances in neural information processing systems*. pp. 529–536 (2005)
11. Han, S., Kim, J.: Video scene change detection using convolution neural network. In: *Proceedings of the 2017 International Conference on Information Technology*. pp. 116–119. ACM (2017)
12. Karpathy, A., Fei-Fei, L.: Deep visual-semantic alignments for generating image descriptions. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. pp. 3128–3137 (2015)
13. Lewis, D.D., Catlett, J.: Heterogeneous uncertainty sampling for supervised learning. In: *Machine Learning Proceedings 1994*, pp. 148–156. Elsevier (1994)
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*. pp. 3–12. Springer-Verlag New York, Inc. (1994)
15. Li, L.J., Su, H., Fei-Fei, L., Xing, E.P.: Object bank: A high-level image representation for scene classification & semantic feature sparsification. In: *Advances in neural information processing systems*. pp. 1378–1386 (2010)
16. Liu, A., Jun, G., Ghosh, J.: A self-training approach to cost sensitive uncertainty sampling. *Machine Learning* **76**(2), 257–270 (Sep 2009). <https://doi.org/10.1007/s10994-009-5131-9>, <https://doi.org/10.1007/s10994-009-5131-9>
17. Mao, C.H., Lee, H.M., Parikh, D., Chen, T., Huang, S.Y.: Semi-supervised co-training and active learning based approach for multi-view intrusion detection. In: *Proceedings of the 2009 ACM Symposium on Applied Computing*. pp. 2042–2048. SAC '09, ACM, New York, NY, USA (2009). <https://doi.org/10.1145/1529282.1529735>, <http://doi.acm.org/10.1145/1529282.1529735>
18. Muslea, I., Minton, S., Knoblock, C.A.: Active + semi-supervised learning = robust multi-view learning. In: *Proceedings of the Nineteenth International Conference on Machine Learning*. pp. 435–442. ICML '02, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2002), <http://dl.acm.org/citation.cfm?id=645531.655845>
19. Pulc, P.: Replication data for: Feed-forward neural networks for video scene classification from statistical features (2018). <https://doi.org/10.7910/DVN/MPZGWO>, <https://doi.org/10.7910/DVN/MPZGWO>
20. Roy, N., McCallum, A.: Toward optimal active learning through monte carlo estimation of error reduction. *ICML*, Williamstown pp. 441–448 (2001)

21. Sabata, T., Borovicka, T., Holena, M.: K-best viterbi semi-supervised active learning in sequence labelling (2017)
22. Serrano, N., Savakis, A., Luo, A.: A computationally efficient approach to indoor/outdoor scene classification. In: Pattern Recognition, 2002. Proceedings. 16th International Conference on. vol. 4, pp. 146–149. IEEE (2002)
23. Settles, B.: Active learning. Synthesis Lectures on Artificial Intelligence and Machine Learning **6**(1), 1–114 (2012)
24. Settles, B., Craven, M., Ray, S.: Multiple-instance active learning. In: Advances in neural information processing systems. pp. 1289–1296 (2008)
25. Seung, H.S., Opper, M., Sompolinsky, H.: Query by committee. In: Proceedings of the Fifth Annual Workshop on Computational Learning Theory. pp. 287–294. COLT '92, ACM, New York, NY, USA (1992). <https://doi.org/10.1145/130385.130417>, <http://doi.acm.org/10.1145/130385.130417>
26. Song, F.Y.Y.Z.S., Xiao, A.S.J.: Construction of a large-scale image dataset using deep learning with humans in the loop. arXiv preprint arXiv:1506.03365 (2015)
27. Szummer, M., Picard, R.W.: Indoor-outdoor image classification. In: Content-Based Access of Image and Video Database, 1998. Proceedings., 1998 IEEE International Workshop on. pp. 42–51. IEEE (1998)
28. Tang, Y.: Deep learning using linear support vector machines. arXiv preprint arXiv:1306.0239 (2013)
29. Wang, L., Guo, S., Huang, W., Xiong, Y., Qiao, Y.: Knowledge guided disambiguation for large-scale scene classification with multi-resolution cnns. IEEE Transactions on Image Processing **26**(4), 2055–2068 (2017)
30. Wang, W., Zhou, Z.H.: On multi-view active learning and the combination with semi-supervised learning. In: Proceedings of the 25th international conference on Machine learning. pp. 1152–1159. ACM (2008)
31. Yao, L., Sun, C., Wang, X., Wang, X.: Combining self learning and active learning for chinese named entity recognition. Journal of software **5**(5), 530–537 (2010)
32. Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., Torralba, A.: Places: A 10 million image database for scene recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence (2017)
33. Zhu, X.: Semi-supervised learning literature survey (2005)

---

# Active Stream Learning with an Oracle of Unknown Availability for Sentiment Prediction

Elson Serrao and Myra Spiliopoulou

Otto-von-Guericke-University Magdeburg, Germany  
elson.serrao@gmail.com, myra@ovgu.de

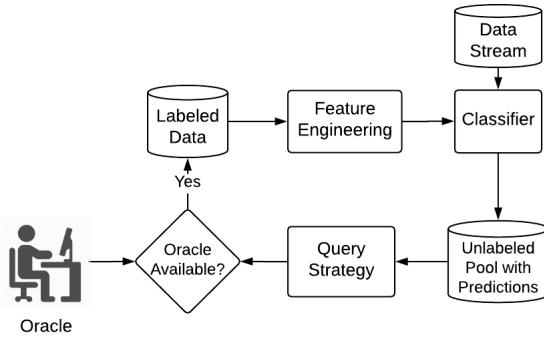
**Abstract.** Active learning holds the promise of learning models from the data with minimal expert input. However, it assumes that the expert is always available or only at the beginning. We waive this assumption and investigate to what extent active learning is effective in practice. We focus on sentiment classification over real streams of opinions. We show that at least for the two real streams we have analyzed, the random strategy is very competitive, and querying the expert in an intelligent way does not bring many advantages, at least when the expert is irregularly available.

**Keywords:** active learning, oracle availability, polarity model learning, opinion stream mining

## 1 Introduction

The objective of active learning is to obtain better or comparable performance to a fully supervised learner with fewer labels if the learner is given the opportunity to select the instances for which it requires labels [1]. Active learning is thus very suitable in those scenarios where there is an abundance of unlabeled data and obtaining new labels is rather expensive. Labels are obtained using an oracle who can, for example, be a domain expert or a human annotator from a crowd-sourcing platform. However, it is often assumed that there is a single oracle that is always correct, always available and inexpensive to query. While there are surveys [1], [2], and [3] and studies [4], [5] that provide insights to the above mentioned challenges in active learning, only a few studies focus on the availability of the oracle for streaming data [6].

In this paper, we consider a stream of opinionated documents and try to predict the sentiment of the document as being either positive, negative or neutral. Over time drift may be observed in the stream due to evolving topics, data and vocabulary, requiring the classifier to adapt to the opinionated stream. For this we use active learning to obtain new labels from the data stream. Instances to be labeled by the oracle are sampled using an appropriate query strategy. However, we assume that the oracle is available irregularly i.e. according to a pattern unknown to the learner. This implies that the oracle may be queried at each moment, but will respond by delivering the label only if it is available.



**Fig. 1.** Interaction of the stream learner and an irregularly available oracle  
If the oracle is unavailable, the instance is not used. This workflow is shown in Figure 1.

The remaining of the paper is organized as follows. Section 2 discusses the related work. In Section 3 we detail our framework and the active learning query strategies used. Section 4 describes the setup for our experiments. While in Section 5 we discuss the results of those experiments. We present our concluding remarks in Section 6.

## 2 Related Work

Most of the algorithms for active learning on streams either assume infinite verification latency, whereupon they invoke semi-supervised learning [7] or they assume that the Oracle is always available to provide labels [8], [9], [10], [11], [12].

Shickel and Rashidi in [6] propose a framework that is aware of the oracle's availability for data streams. Their framework considers multiple oracles and focuses on querying first those oracles that have a higher availability. They try to achieve a cost-benefit tradeoff by using a dynamic labeling budget proportional to the oracle's availability with the cost of labeling an instance inversely proportional to the oracle's availability. However, such a cost-benefit tradeoff seems unrealistic in real-world scenarios where the cost is likely to be regulated by the difficulty in obtaining the label and other factors [5]. No experiments were conducted with oracles of varying expertise, which is often seen in active learning literature considering multiple oracles.

## 3 Active Learning on an Opinionated Stream

We consider a data stream  $\mathcal{D}$  of opinionated documents observed at distinct timepoints  $t_0, t_1, \dots, t_i, \dots$  where at each timepoint  $t_i$  we receive a batch of documents. We define the timepoint on a temporal level where, for example, the timepoint could be a week. Consequently, all the documents arriving during the time period from  $t_{i-1}$  to  $t_i$  would comprise the batch of documents for  $t_i$ .

Our framework encompasses an algorithmic core described in Section 3.1 and the query strategies in Section 3.2. We link this framework with a simulator of the oracle’s availability, described in Section 4.1, and with an algorithm for the preparation of the opinionated data stream, described in Section 4.2.

### 3.1 Modeling Oracle Unavailability during Querying

We consider the beginning of the stream at  $t_0$  to be characterized by the availability of an initial set of labeled documents  $L_0$ . The initially labeled documents  $L_0$  are used to initialize the classifier  $\Delta$ . At subsequent timepoints i.e.  $t_1$  onwards, we receive unlabeled data  $U_t$ . If the budget  $\mathcal{B}$  is not exceeded, for every unlabeled document  $x$ , we use the trained classifier  $\Delta$  to predict the probability  $P(\hat{y}_c|x) \forall c \in C$ , where  $c$  represents the sentiment of the document, namely, positive, negative or neutral. Our method calculates the confidence of the learner’s prediction  $\mathcal{I}$  using metric  $\phi$ , and launches a request for the true label  $y$  when necessary. The oracle provides the true label  $y$  only if it is available and the document  $x$  is added to the labeled data for the next iteration. In the event that the oracle is unavailable,  $x$  is not used to adapt the learner. An overview of our framework is shown in Algorithm 1.

We assume the cost of labeling is the same for every document at any time  $t$ . If  $n_t$  is the number of queries sent to the oracle at  $t$ , then the utilized budget at  $t$  is given by

$$\frac{n_t}{|U_t|} < \mathcal{B} \quad (1)$$

To adapt to the evolving data stream, we utilize a sliding window  $\mathcal{W}$  [13]. At every timepoint  $t$ , we add the labeled documents  $L_t$  to the window. Once the window is full, the documents from the oldest timepoint within the window are forgotten. Depending on the chosen classifier  $\Delta$ , for every iteration, there may be a need to retrain  $\Delta$  with the documents in the window.

Although the framework is capable of using any confidence metric such as maximum posterior probability or the maximum margin between the first and second most probable class, we propose calculating the confidence of a prediction using entropy as,

$$\phi_H = 1 - \left[ - \sum_{c \in C} P(\hat{y}_c|x) \log_{|C|} P(\hat{y}_c|x) \right] \quad (2)$$

### 3.2 Query Strategies

We use uncertainty based query strategies for the active learner. We also use the variable uncertainty and variable randomized uncertainty strategies introduced by Žliobaitė et al. in [9], [10] and [11]. The difference in our implementation is that we have generalized the strategies to allow the use of any confidence metric while determining if an instance needs to be sampled.

**Random Strategy:** This strategy randomly selects an instance to be labeled by the oracle with a probability given by the budget  $\mathcal{B}$ . In this sense, it is very naive and is used as the baseline strategy.

**Algorithm 1:** Active Learning with an Irregularly Available Oracle

---

**Input:**  $\Delta$  - classifier with relevant parameters;  $size$  - window size;  $\mathcal{B}$  - budget;  $o$  - oracle;  $\phi$  - confidence metric;  $\text{queryStrategy}$  - query strategy with parameters;

**Initialize:**  $t \leftarrow 0$ ;  $\mathcal{W} \leftarrow \text{SlidingWindow}(size)$

- 1 Receive labeled data  $L_t$
- 2  $\mathcal{W} \leftarrow \text{addToWindow}(L_t)$
- 3 **for**  $t = 1, 2, \dots$  **do**
- 4     Train classifier  $\Delta$  with instances in  $\mathcal{W}$
- 5     Receive unlabeled data  $U_t$
- 6      $n_t \leftarrow 0$
- 7      $L_t \leftarrow \emptyset$
- 8     **for** each instance  $x \in U_t$  **do**
- 9          $P_c \leftarrow P_{\Delta}(\hat{y}_c|x) \forall c \in C$  // predict the probability
- 10          $\hat{y} \leftarrow \arg \max_c(P_c)$  // predicted label
- 11         **if**  $(n_t / |U_t|) < \mathcal{B}$  **then**
- 12              $I \leftarrow \phi(P_c)$  // compute the confidence
- 13             **if**  $\text{queryStrategy}(\mathcal{B}, I, \dots) = True$  **then**
- 14                  $n_t \leftarrow n_t + 1$
- 15                 **if**  $\text{isAvailable}(o)$  **then**
- 16                      $y \leftarrow \text{get true label of } x \text{ from the oracle } o$
- $L_t \leftarrow L_t \cup (x, y)$
- 16      $\mathcal{W} \leftarrow \text{addToWindow}(L_t)$

---

**Fixed Uncertainty Strategy:** This strategy samples those instances which the learner are least confident of by comparing the confidence of prediction of an instance to a fixed threshold  $\theta$ .

**Variable Uncertainty Strategy:** For a learner to adapt to an evolving data stream, obtaining labels for the least confident instances within each time-point would be more beneficial. The variable uncertainty strategy described in Algorithm 2 provides a variable threshold that adjusts itself to the incoming data stream. When the data stream speeds up, the confidence of the learner decreases. In such cases, it decreases its threshold so that least confident instances are queried first. On the other hand, at times when the learner is confident of its prediction, the threshold is increased to capture the most uncertain instances.

**Algorithm 2:** VariableUncertainty( $\mathcal{I}, s$ )

---

**Input:**  $\mathcal{I}$  - confidence score;  $s \in (0, 1]$  - threshold adjustment step

**Output:** *True* if true label is required else *False*

**Initialize:** labeling threshold  $\theta \leftarrow 1$

- 1 **if**  $\mathcal{I} < \theta$  **then**
- 2      $\theta \leftarrow \theta(1 - s)$ // uncertain instance: decrease the threshold
- 3     **return** *True*
- 4 **else**
- 5      $\theta \leftarrow \theta(1 + s)$ // confident instance: increase the threshold
- 6     **return** *False*

---

**Variable Randomized Uncertainty Strategy:** Uncertainty based active learning query strategies generally focus on sampling those instances that are close to the decision boundary of the learner. In evolving data streams changes may occur anywhere in the instance space. So as to not miss the change that may occur elsewhere, the variable randomized uncertainty strategy occasionally samples those instances that the learner is confident of. As shown in Algorithm 3, the threshold is multiplied by a normally distributed random variable to sample the confident instances.

---

**Algorithm 3: VariableRandomizedUncertainty( $\mathcal{I}, s, \delta$ )**

---

**Input:**  $\mathcal{I}$  - confidence score;  $s \in (0, 1]$  - threshold adjustment step  
 $\delta$  - variance of threshold randomization

**Output:** *True* if true label is required else *False*

**Initialize:** labeling threshold  $\theta \leftarrow 1$

```

1  $\theta_{randomized} \leftarrow \theta \times \eta$ , where  $\eta \in \mathcal{N}(1, \delta)$  is a random multiplier
2 if  $\mathcal{I} < \theta_{randomized}$  then
3    $\theta \leftarrow \theta(1 - s)$  // uncertain instance: decrease the threshold
4   return True
5 else
6    $\theta \leftarrow \theta(1 + s)$  // confident instance: increase the threshold
7   return False
```

---

## 4 Experiment Setup

The goal of our experiments is to study how the performance of the learner is affected when the availability of the oracle changes. The following sub-sections describe the oracle availability simulator, the datasets used, and the evaluation criteria and strategy.

### 4.1 Simulator of Oracle Availability

At timepoint  $t$ , we consider the oracle to be available with a probability of  $\alpha_t$ . In our experiments, at any timepoint  $t$ , we set the oracle's availability  $\alpha_t = \alpha$  and is considered to be independent of the availability at  $t - 1$ . Algorithm 4 provides a more formal definition of our simulator.

---

**Algorithm 4: Oracle Availability Simulator**

---

**Input:**  $\alpha \in (0, 1]$  - availability of the oracle;  
**Output:** *True* if the oracle is available else *False*

```

1 return uniform(0, 1)  $\leq \alpha$ 
```

---

### 4.2 Datasets and Feature Engineering

**Yelp:** The Yelp Dataset <sup>1</sup> contains about 5.2 million reviews of various businesses over a period of 13 years from 11 metropolitan areas across 4 countries. We

---

<sup>1</sup> <https://www.yelp.com/dataset/challenge>

filtered the dataset for English reviews and additionally removed those reviews whose length was less than 15 words. For our data stream, we considered the reviews from 2009 onwards.

**Amazon:** The Amazon dataset introduced in [14], contains reviews of several product categories. Using the 5-core datasets of the product categories, we build a dataset with an intention of introducing concept drift. For this purpose, we randomly selected three product categories every three months from among the following nine categories: *Home and Kitchen*, *Kindle Store*, *Health and Personal Care*, *Cell Phones and Accessories*, *Apps for Android*, *Electronics*, *Clothing*, *Shoes and Jewelry*, *CDs and Vinyl*, and *Beauty*. We removed the duplicate reviews arising from the product having multiple categories.

Both the Yelp and Amazon employ a 5-star rating scheme, where 5-stars is the highest rating while 1-star is the lowest. We considered the 1 and 2-star rating to be negative, 3-star rating neutral and 4 and 5-star rating positive.

**Feature Engineering:** We preprocessed the reviews by replacing URLs, negations and currencies with the tokens URL, NEGATION and CURRENCY respectively and some emoticons with tokens like SMILE and HEART. We further suppressed repeated letters, expanded contractions (e.g. "I'm" into "I am"), removed stopwords and replaced words by their lemmas.

After preprocessing, we extracted features from the reviews using word  $n$ -grams with  $n = 3$  along with its corresponding frequency of occurrence. We selected the most relevant 15000 features using the chi-square test. Our feature vectors were then constructed using the TF-IDF weighting scheme.<sup>2</sup>

### 4.3 Evaluation Strategy and Evaluation Criteria

We define the duration of a timepoint to be a week and maintain a sliding window of five weeks. We perform prequential evaluation [13]: we first test on all documents for the incoming week and then adapt by using only the sampled documents whose labels have been provided by the oracle.

As we use entropy to calculate our confidence measure, we evaluate on log loss decrease [2]. The log loss  $l_t$  at timepoint  $t$  is given by,

$$l_t = -\frac{1}{|U_t|} \sum_{i \in U_t} \sum_{j \in C} b_{ij} \log p_{ij} \quad (3)$$

where  $b_{ij}$  is a binary indicator of whether or not label  $j$  is the correct classification for instance  $i$ , and  $p_{ij}$  is the model probability of assigning label  $j$  to instance  $i$ .

## 5 Experimental Evaluation

As the Stochastic Gradient Descent classifier was found to be effective for sentiment analysis in [15], we used the same with hinge loss,  $l2$  penalty and alpha value of 0.0001 to optimize the objective function of a linear support vector machine as our *base learner*. The base learner was calibrated using Platt Scaling to obtain probabilistic outputs.

---

<sup>2</sup> Code and supplementary material available at <https://github.com/elrasp/osm>

For each timepoint  $t$ , we set a fixed budget  $\mathcal{B} = 0.1$ . For the query strategies, we set the fixed threshold  $\theta = 0.9$ , and the suggested values of 0.01 and 1 for the threshold adjustment step  $s$  and the variance of the normally distributed random number generator  $\delta$  respectively [11].

In Section 5.1, we describe the underlying class distribution of the data stream for these datasets and in Section 5.2, we analyze the influence of the oracle’s availability on the performance of the learner.

### 5.1 Distribution of Data

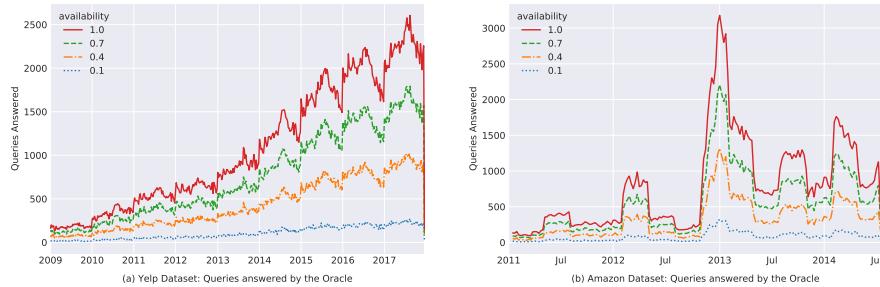
The weekly underlying class distribution of data in the stream of opinionated documents for the Yelp and Amazon datasets are shown in Figure 3(a) and Figure 3(b) respectively.

**Yelp:** For the Yelp dataset, we observe a gradual increase in the number of reviews obtained over time. The proportion of neutral reviews received remain almost constant for the entire data stream. In comparison, the positive and negative reviews are always increasing. Also, we find that the positive reviews dominate the class distribution accounting for more than 50% of the reviews in any week.

**Amazon:** Unlike the Yelp dataset, the amazon dataset exhibits sudden bursts in the volume of reviews received. This occurs as some chosen product categories are more popular than others and receive more reviews. We also observe that mostly there is a burst in the positive reviews received as compared to the negative and neutral reviews. Similar to the Yelp dataset, the positive reviews dominate the class distribution.

### 5.2 Impact of the Oracle’s Availability on Learning

Figure 2 shows how the oracle availability, simulated by the method of Section 4.1, affects the number of queries answered. We vary the availability between 1.0 (all queries answered) and 0.1 (only 10% of the queries per batch are answered) in steps of 0.1.



**Fig. 2.** Queries answered by the Oracle for varying availabilities

In Figure 3 we show the results of evaluation for the Yelp (on the left) and Amazon (on the right) datasets. We aggregated the log loss results for all the timepoints every two months and plot the mean (lines) and standard deviation (colored area around the line). For the Yelp dataset, in general, we observe that the error in performance gradually reduces as the stream progresses irrespective of the oracles availability and the query strategy used. On the other hand, the learner finds it much more difficult to adapt to the evolving data stream of the Amazon dataset.

In the early stages of the stream, where the data volume in the stream is low, we observe the learner performing better for lower oracle availabilities. As more and more data is accumulated, the need to have the oracle for the Yelp dataset drops but remains for the Amazon dataset as it exhibits more drift.

We further conducted experiments with oracle availabilities varying between 0.01 and 0.1 in steps of 0.01 and compared the different query strategies at varying availabilities with the non-parametric Friedman’s test followed by Nemenyi post-hoc [16]. Friedman’s test proceeds by ranking the models under consideration. The best performing model is given the rank 1, the second best 2 and so on. If two or more models have identical performance they are given an average rank. The null hypothesis of Friedman’s test states that all the models perform the same and thus, will have the same average rank. If the test rejects the null hypothesis, we proceed with the Nemenyi post-hoc test that makes pair-wise comparisons of the different models. It identifies statistically significant models if the difference between their average rank is more than the critical distance.

Figure 4(a) and Figure 4(b) shows the critical distance diagram of the Yelp and Amazon dataset respectively. In these diagrams the better performing models are ranked higher and are placed to the right. The model name corresponds to a combination of the query strategy and the value of the oracle availability, whereupon ”rand” in the name refers to the random strategy. The models whose difference in average rank is less than the calculated critical distance (CD) are connected to each other by a horizontal line, indicating that these models are statistically indifferent to each other.

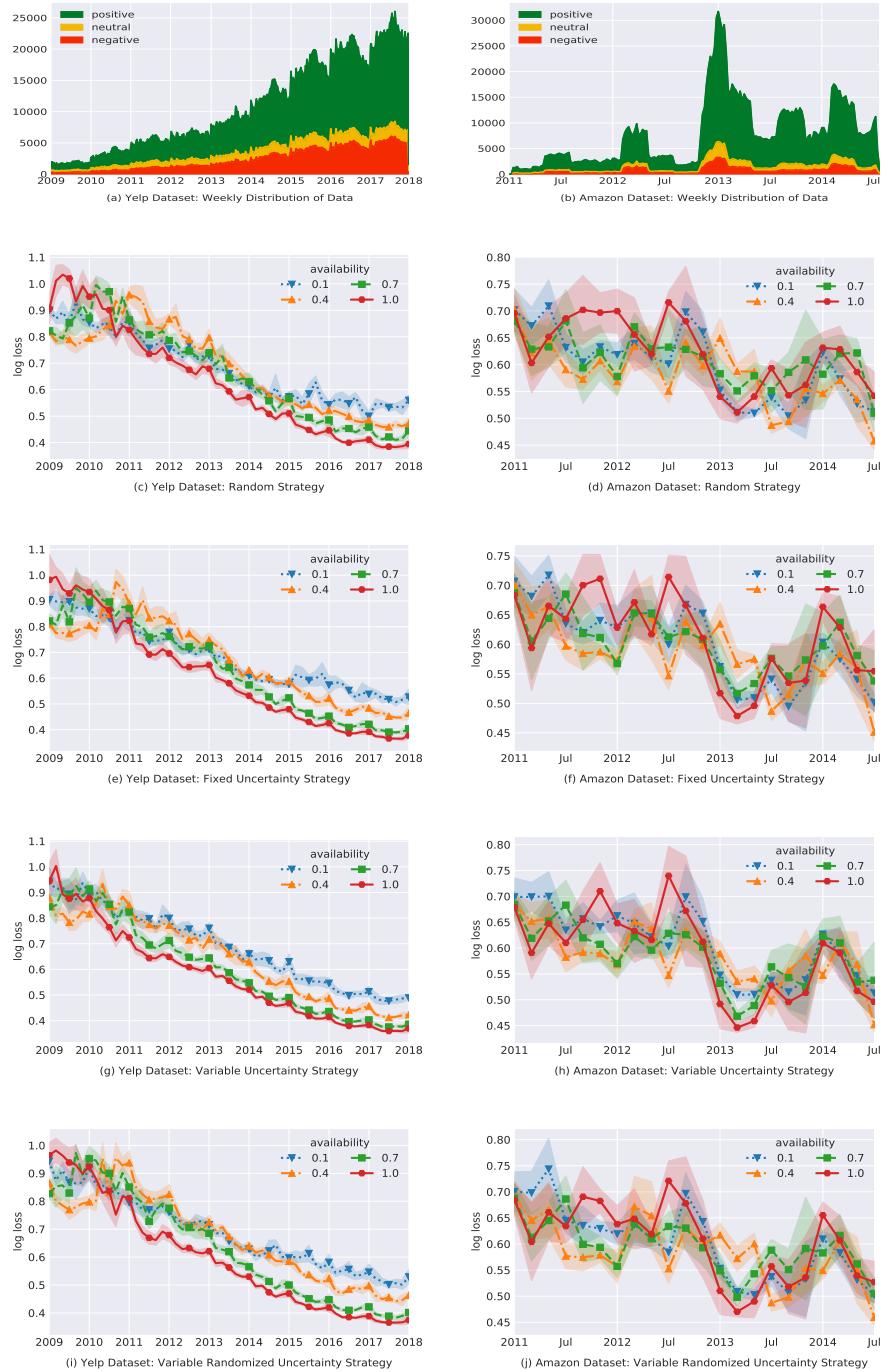
As we can see in these figures, there are strategies that which perform significantly better when the oracle availability changes. At very low oracle availabilities we find that there is no strategy that performs significantly better than the others. For the Amazon dataset, even at higher oracle availabilities there is no difference in the performance of the strategies. This could mainly be attributed to the nature of the drift exhibited in both the datasets.

## 6 Conclusions

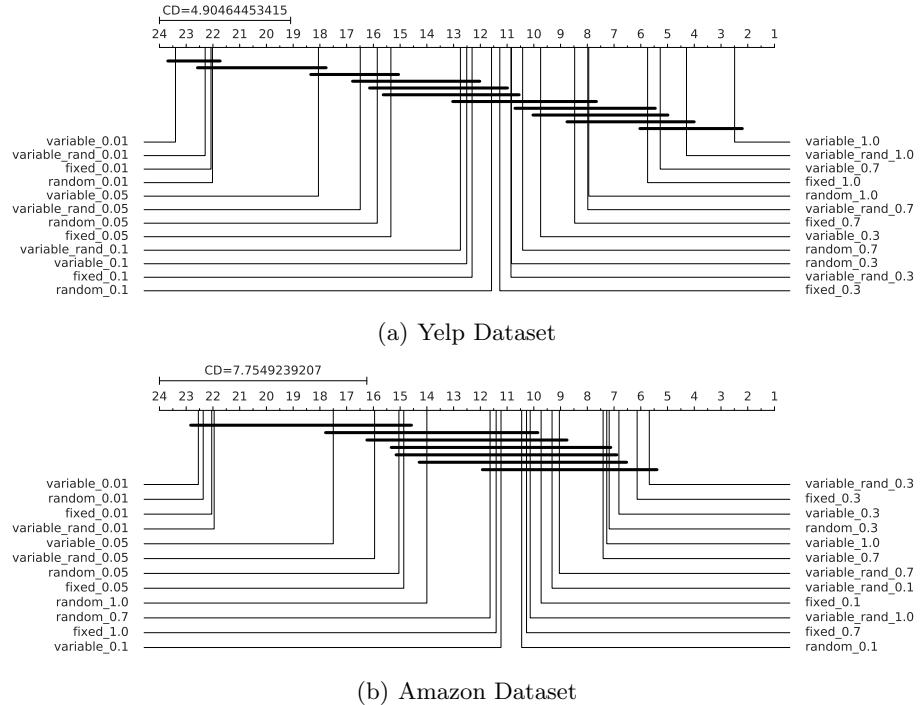
The need for an oracle depends on the overall variability of the dataset. If there is convergence over time as in the case of the Yelp dataset, the need for an oracle is limited, because the learner can predict the labels by itself. Hence, low availability of the oracle is only relevant if there is drift.

### Active Stream Learning with an Oracle of Unknown Availability

---



**Fig. 3.** Evaluation results of the Yelp (on the left) and Amazon (on the right) datasets



**Fig. 4.** Critical Distance Diagram. The models can be identified by the query strategy and oracle availability. The best performing models are shown to the right and models that are not statistically different to each other are connected by the horizontal line.

If an oracle is available, the random strategy is a good choice, as it shows the same tendency as the other strategies, is easy to implement and is fast. However, more experiments are needed to check whether stronger active learning strategies can beat the random sampler in this setting, by capitalizing more effectively on the few available labels. Experiments are also required for different domains.

To improve model quality, we intend to consider more elaborate querying strategies [17], and to investigate whether instance-based active learning might deliver better results than the block-based active learning paradigm we currently use.

To compensate for oracle inavailability, we also intend to combine active learning with semi-supervised learning. Semi-supervised methods are used to propagate labels to the arriving instances, cf. [18], [19]. In that case, we want to investigate how disagreement between oracle and self-learner can be alleviated in a seamless way.

*Acknowledgements.* This work was inspired and partially conducted (last author) within the project OSCAR Opinion Stream Classification with Ensembles and Active Learners, funded by the German Research Foundation.

## References

1. Settles, B.: Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison (2009)
2. Fu, Y., Zhu, X., Li, B.: A survey on instance selection for active learning. *Knowledge and Information Systems* **35**(2) (May 2013) 249–283
3. Lugofer, E.: On-line active learning: A new paradigm to improve practical useability of data stream modeling methods. *Information Sciences* **415-416** (nov 2017) 356–376
4. Wu, W., Liu, Y., Guo, M., Wang, C., Liu, X.: A probabilistic model of active learning with multiple noisy oracles. *Neurocomputing* **118** (2013) 253 – 262
5. Dommez, P., Carbonell, J.G.: Proactive learning: Cost-sensitive active learning with multiple imperfect oracles. In: Proceedings of the 17th ACM Conference on Information and Knowledge Management. CIKM ’08, New York, NY, USA, ACM (2008) 619–628
6. Shickel, B., Rashidi, P.: ART: an availability-aware active learning framework for data streams. In Markov, Z., Russell, I., eds.: Proceedings of the Twenty-Ninth International Florida Artificial Intelligence Research Society Conference, FLAIRS 2016, Key Largo, Florida, May 16-18, 2016., AAAI Press (2016) 92–97
7. Zimmermann, M., Ntoutsi, E., Spiliopoulou, M.: Adaptive semi supervised opinion classifier with forgetting mechanism. In: Proceedings of the 29th Annual ACM Symposium on Applied Computing. SAC ’14, New York, NY, USA, ACM (2014) 805–812
8. Zimmermann, M., Ntoutsi, E., Spiliopoulou, M.: Incremental active opinion learning over a stream of opinionated documents. WISDOM’15 (Workshop on Issues of Sentiment Discovery and Opinion Mining) 2015 at Knowledge Discovery and Data Mining, KDD’15 Workshops 2015, Sydney, Australia, August 10, 2015 (2015)
9. Žliobaitė, I., Bifet, A., Pfahringer, B., Holmes, G.: Active learning with evolving streaming data. In Gunopulos, D., Hofmann, T., Malerba, D., Vazirgiannis, M., eds.: Machine Learning and Knowledge Discovery in Databases, Berlin, Heidelberg, Springer Berlin Heidelberg (2011) 597–612
10. Žliobaitė, I., Bifet, A., Holmes, G., Pfahringer, B.: Moa concept drift active learning strategies for streaming data. In Diethe, T., Balcazar, J., Shawe-Taylor, J., Tirnauca, C., eds.: Proceedings of the Second Workshop on Applications of Pattern Analysis. Volume 17 of Proceedings of Machine Learning Research., CIEM, Castro Urdiales, Spain, PMLR (19–21 Oct 2011) 48–55
11. Žliobaitė, I., Bifet, A., Pfahringer, B., Holmes, G.: Active learning with drifting streaming data. *IEEE Transactions on Neural Networks and Learning Systems* **25**(1) (Jan 2014) 27–39
12. Smailović, J., Grčar, M., Lavrač, N., Žnidaršić, M.: Stream-based active learning for sentiment analysis in the financial domain. *Inf. Sci.* **285**(C) (November 2014) 181–203
13. Gama, J.a., Žliobaitė, I., Bifet, A., Pechenizkiy, M., Bouchachia, A.: A survey on concept drift adaptation. *ACM Comput. Surv.* **46**(4) (March 2014) 44:1–44:37
14. McAuley, J., Targett, C., Shi, Q., van den Hengel, A.: Image-based recommendations on styles and substitutes. In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval - SIGIR 2015, ACM Press (2015)
15. Bifet, A., Frank, E.: Sentiment knowledge discovery in twitter streaming data. In: Proceedings of the 13th International Conference on Discovery Science. DS’10, Berlin, Heidelberg, Springer-Verlag (2010) 1–15

16. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.* **7** (December 2006) 1–30
17. Kottke, D., Krempel, G., Spiliopoulou, M.: Probabilistic active learning in data-streams. In Fromont, E., De Bie, T., van Leeuwen, M., eds.: *Advances in Intelligent Data Analysis XIV*, Cham, Springer International Publishing (2015) 145–157
18. Dyer, K.B., Capo, R., Polikar, R.: COMPOSE: A semisupervised learning framework for initially labeled nonstationary streaming data. *IEEE Transactions on Neural Networks and Learning Systems* **25**(1) (jan 2014) 12–26 Journal Article Research Support, U.S. Gov’t, Non-P.H.S.
19. Souza, V.M.A., Silva, D.F., Gama, J., Batista, G.E.A.P.A.: Data stream classification guided by clustering on nonstationary environments and extreme verification latency. In Venkatasubramanian, S., Ye, J., eds.: *Proceedings of the 2015 SIAM International Conference on Data Mining*. [Society for Industrial and Applied Mathematics] (jun 2015) 873–881

---

# ACTIVMETAL: Algorithm Recommendation with Active Meta Learning

Lisheng Sun-Hosoya<sup>1</sup>, Isabelle Guyon<sup>1,2</sup>, and Michèle Sebag<sup>1</sup>

<sup>1</sup> UPSud/CNRS/INRIA, Univ. Paris-Saclay. <sup>2</sup> ChaLearn

**Abstract.** We present an *active meta learning* approach to model selection or algorithm recommendation. We adopt the point of view “collaborative filtering” recommender systems in which the problem is brought back to a missing data problem: given a sparsely populated matrix of performances of algorithms on given tasks, predict missing performances; more particularly, predict which algorithm will perform best on a new dataset (empty row). In this work, we propose and study an active learning version of the recommender algorithm CofiRank algorithm and compare it with baseline methods. Our benchmark involves three real-world datasets (from StatLog, OpenML, and AutoML) and artificial data. Our results indicate that CofiRank rapidly finds well performing algorithms on new datasets at reasonable computational cost.

**Keywords:** Model Selection · Recommender · Active Meta Learning.

## 1 Introduction

While Machine Learning and Artificial Intelligence are taking momentum in many application areas ranging from computer vision to chat bots, selecting the best algorithm applicable to a novel task still requires human intelligence. The field of AutoML (Automatic Machine Learning), aiming at automatically selecting best suited algorithms and hyper-parameters for a given task, is currently drawing a lot of attention. Progress in AutoML has been stimulated by the organization of challenges such as the AutoML challenge series<sup>1</sup>. Among the winning AutoML approaches are AUTOWEKA and AUTO-SKLEARN, developed by the Freiburg team [6,5,7] (more in Section 2). These approaches, taking inspiration from Bayesian optimization [4], alternatively learn an inexpensive estimate of model performance on the current dataset, and use this estimate to reduce the number of model candidates to be trained and tested using the usual expensive cross-validation procedure. A novel ingredient of AUTO-SKLEARN, referred to as “meta-learning”, takes in charge the initialization of the Bayesian optimization process, with a predictor using “meta-features” describing the datasets. Meta-learning reportedly yields significant improvements over random initializations.

---

<sup>1</sup> <http://automl.chalearn.org>

Another approach targeting AutoML is based on recommender systems (RS), popularized by the Netflix challenge [2]. RS approaches seek the item best suited to a given user, based on historical user-item interactions and user preferences. By analogy [15] proposed first to treat algorithm selection as a recommender problem in which datasets “prefer” algorithms solving their task with better performance. Along this line, the “Algorithm Recommender System” ALORS [9], combines a recommender system and an estimate of model performance based on predefined meta-features, to achieve AutoML (more in Section 2).

In this paper, we propose an *active meta-learning* approach inspired by AUTO\_SKLEARN and ALORS. Formally (Section 3), given a matrix of historical algorithm performance on datasets, we aim at finding *as fast as possible* the model with best performance on a new dataset. The originality compared to the former approaches lies in the coupled search for the meta-features describing the dataset, the model performance based on these meta-features, and the selection of a candidate model to be trained and tested on the dataset.

This paper is organized as follows: After briefly reviewing the SOTA in Section 2, we formalize our problem setting in Section 3. We then describe the benchmark data in Section 4 and provide an empirical validation of the approach in Section 5. While the validation considers only the “classical” machine learning settings, it must be emphasized that the proposed approach does not preclude of any type of tasks or algorithms, hence is applicable to a broader range of problems.

## 2 State of the art

It is notorious that the success of model search techniques can be dramatically improved by a careful initialization. In AUTO\_SKLEARN, the search is improved by a sophisticated initialization using a form of transfer learning [10] called “meta-learning”. The meta-data samples include all the datasets of openml.org [12] (a platform which allows to systematically run algorithms on datasets). Systematically launching AUTO\_SKLEARN on each dataset yields the best (or near best) models associated with each dataset.

Independently, each dataset is described using so-called meta-features. Meta-features are generally of two kinds: i) simple statistics of the dataset such as number of training examples, number of features, fraction of missing values, presence of categorical variables, etc.; ii) performance on the current dataset of “landmark algorithms”, namely a well-chosen set of algorithms that can be trained and tested with moderate computational effort such as one nearest neighbor (1NN) or decision trees.

When considering a new dataset, AUTO\_SKLEARN first determines its nearest neighbors in the meta-feature space, and initializes the search using the best models associated with these neighbors. Other meta-learning formalisms, not considered further in this paper, are based on learning an estimate of the model performance from meta-features [11], or learning to predict the best performing algorithm, as a multi-class classification problem [17].

The delicate issue is to control the cost of the initialization step: considering many landmark algorithms comes with an expected benefit (a better initialization of the search), and with a cost (the computational cost of running the landmarks).

As said, recommender systems (RS) aim at selecting the item best suited to a particular user, given a community of users, a set of items and some historical data of past interactions of the users with the items, referred to as “collaborative matrix” [16,3], denoted  $S$  (for “score”) in this paper. As first noted by [15], algorithm selection can be formalized as a recommender problem, by considering that a dataset “likes better” the algorithms with best performances on this dataset. Along this line, one proceeds by i) estimating all algorithm performances on this dataset (without actually evaluating them by training and testing); and ii) recommending the algorithm(s) with best estimated performance on this dataset.

The merits of RS approaches regarding algorithm selection are twofold. Firstly, **RS approaches are frugal** (like other methods, e.g. co-clustering). RS proceeds by estimating the value associated with each (user, item) pair – here, the performance associated with each (algorithm, dataset) – from a tiny fraction of the (user, item) ratings, under the assumption that the collaborative matrix is of low rank  $k$ . More precisely the (usually sparse) matrix  $S$  of dimensions  $(p, N)$  is approximated by  $UV'$ , with  $U$  a  $(p, k)$  matrix and  $V$  a  $(N, k)$  matrix, such that  $\langle U_{i,:}, V_{j,:} \rangle$  is close to  $S_{i,j}$  for all pairs  $i, j$  (e.g. using maximum margin matrix factorization in [14]).  $U$  (respectively  $V$ ) is referred to as latent representation of the users (resp. the items). In the model selection context, RS approaches are thus frugal: they can operate even when the performance of a model on a dataset is known on a tiny fraction of the (model, dataset) pairs. Secondly, most-recent **RS approaches are ranking methods**. Estimating algorithm performance is a harder problem than ranking them in order of merit. A second benefit of RS is that they can rank items conditionally to a given user. The CofiRank algorithm [18] accordingly considers the rank matrix (replacing  $S_{i,j}$  with the rank of item  $j$  among all items user  $i$  has rated) and minimizes the Normalized Discounted Cumulative Gain (NDCG) in which correctness in higher ranked items is more important. As optimizing NDCG is non-convex, CofiRank thus instead optimizes a convex upper-bound of NDCG.

In counterpart for these merits, mainstream RS is not directly applicable to AutoML, as it focuses on recommending items to known users (warm-start recommendation). Quite the contrary, AutoML is concerned with recommending items (models) to new users (new datasets), a problem referred to as cold-start recommendation [13,8]. This drawback is addressed in the general purpose ALORS system [9], where external meta-features are used to estimate the latent representation  $\hat{U}$  of the current dataset; this estimated latent representation is used together with the latent representation of any model to estimate the model performance (as  $\langle \hat{U}_{:,j}, V_j \rangle$ ) and select the model with best estimated performance. The novel active meta-learning approach presented in this paper proposes a different approach to warm start, not requiring external meta-features: Previously evaluated algorithm scores are themselves used as meta-features (see Section 3).

### 3 Problem setting and algorithms

We define the **active meta-learning problem** in a **collaborative filtering recommender** setting as follows:

**GIVEN:**

- An ensemble of **datasets** (or tasks)  $\mathcal{D}$  of elements  $d$  (not necessarily finite);
- A finite ensemble of  $n$  **algorithms** (or machine learning models)  $\mathcal{A}$  of elements  $a_j, j = 1, \dots, N$ ;
- A **scoring program**  $\mathcal{S}(d, a)$  calculating the performance (score) of algorithm  $a$  on dataset  $d$  (*e.g.* by cross-validation). Without loss of generality we will assume that **the larger  $\mathcal{S}(d, a)$ , the better**. The evaluation of  $\mathcal{S}(d, a)$  can be computationally expensive, hence we want to limit the number of times  $S$  is invoked.
- A **training matrix**  $S$ , consisting of  $p$  lines (corresponding to example datasets  $d_i, i = 1, \dots, p$  drawn from  $\mathcal{D}$ ) and  $n$  columns (corresponding to all algorithms in  $\mathcal{A}$ ), whose elements are calculated as  $S_{ij} = \mathcal{S}(d_i, a_j)$ , but may contain missing values (denoted as NaN).
- A **new test dataset**  $d_t \in \mathcal{D}$ , NOT part of training matrix  $S$ . This setting can easily be generalized to test matrices with more than one line.

**GOAL:** Find “as quickly as possible”  $j_* = \text{argmax}_j(S(d_t, a_j))$ .

For the purpose of this paper “as quickly as possible” shall mean by evaluating as few values of  $S(d_t, a_j), j = 1, \dots, n$  as possible. More generally, it could mean minimizing the total computational time, if there is a variance in execution time of  $S(d_t, a_j)$  depending on datasets and algorithms. However, because we rely in our experimental section on archival data without information of execution time, we reserve this refinement for future studies. Additionally, we assume that the computational cost of our meta-learning algorithm (excluding the evaluations of  $\mathcal{S}$ ) is negligible compared to the evaluations of  $\mathcal{S}$ , which has been verified in practice.

In our setting, we reach our goal iteratively, in an **Active Meta Learning** manner (ACTIVMETAL), see Algorithm 1. The variants that we compare differ in the choices of  $\text{INITIALIZATIONSCHEME}(S)$  and  $\text{SELECTNEXT}(S, \mathbf{t})$ , as described in Algorithms 2-5: Given a new dataset (an empty line), we need to initialize it with one or more algorithm performances, this initialization is done by  $\text{INITIALIZATIONSCHEME}(S)$  and is indispensable to fire CofiRank. Algorithms 2-5 show 2 initialization methods: `randperm` in Algorithm 2 (the first algorithm is selected at random) and `median` in Algorithms 3-5 (the algorithms are sorted by their median over all datasets in training matrix  $S$  and the one with highest median is selected as the first algorithm to evaluate). Once we have evaluated the first algorithm, the next algorithms can be chosen with or without active learning, this is done by  $\text{SELECTNEXT}(S, \mathbf{t})$ : Algorithm 2-3 without active meta learning select next algorithms at random or according to median over training datasets, *i.e.* the knowledge from evaluated algorithms on the new dataset is not taken into account; Algorithm 4-5 run CofiRank for active meta learning, which, initialized with performances of evaluated algorithm, returns a ranking of algorithms on the new dataset. The difference is that in Alg. 4 we run CofiRank for

each selection of next algorithm, i.e. CofiRank is initialized with more and more known values. In Alg. 5 CofiRank is run only once at the beginning, initialized with 3 landmark values.

---

**Algorithm 1** ACTIVMETAL

---

```

1: procedure ACTIVMETAL( $\mathcal{A}$ ,  $\mathcal{S}$ ,  $S$ ,  $d_t$ ,  $n_{max}$ )
2:    $n \leftarrow \text{size}(S, 2)$                                  $\triangleright$  Number of algorithms to be evaluated on  $d_t$ 
3:    $\mathbf{t} \leftarrow \text{NaNvector}(n)$      $\triangleright$  Algorithm scores on  $d_t$  are initialized w. missing values
4:    $j_+ \leftarrow \text{INITIALIZATIONSCHEME}(S)$            $\triangleright$  Initial algorithm  $a_{j_+} \in \mathcal{A}$  is selected
5:   while  $n < n_{max}$  do
6:      $\mathbf{t}[j_+] \leftarrow \mathcal{S}(d_t, a_{j_+})$   $\triangleright$  Complete  $\mathbf{t}$  w. one more prediction score of  $a_{j_+}$  on  $d_t$ 
7:      $j_+ = \text{SELECTNEXT}(S, \mathbf{t})$ 
8:      $n \leftarrow \text{length}(\text{notNaN}(\mathbf{t}))$             $\triangleright$  number of algorithms evaluated on  $d_t$ 
9:   return  $j_+$ 

```

---



---

**Algorithm 2** Random

---

```

1: procedure INITIALIZATIONSCHEME( $S$ )
2:    $\mathbf{r} \leftarrow \text{randperm}(\text{size}(S, 2))$        $\triangleright$  Replaced by something more clever elsewhere
3:   return  $j_+ \leftarrow \text{argmax}(\mathbf{r})$ 
4: procedure SELECTNEXT( $S, \mathbf{t}$ )
5:   evaluated  $\leftarrow \text{notNaN}(\mathbf{t})$ 
6:    $\mathbf{r} \leftarrow \text{randperm}(\text{size}(S, 2))$        $\triangleright$  Replaced by something more clever elsewhere
7:    $\mathbf{r}(\text{evaluated}) \leftarrow -\text{Inf}$ 
8:   return  $j_+ \leftarrow \text{argmax}(\mathbf{r})$ 

```

---



---

**Algorithm 3** SimpleRankMedian

---

```

1: procedure INITIALIZATIONSCHEME( $S$ )
2:    $\mathbf{r} \leftarrow \text{median}(S, 2)$                        $\triangleright$  Column-wise median
3:   return  $j_+ \leftarrow \text{argmax}(\mathbf{r})$ 
4: procedure SELECTNEXT( $S, \mathbf{t}$ )
5:   evaluated  $\leftarrow \text{notNaN}(\mathbf{t})$ 
6:    $\mathbf{r} \leftarrow \text{median}(S, 2)$                        $\triangleright$  Column-wise median
7:    $\mathbf{r}(\text{evaluated}) \leftarrow -\text{Inf}$ 
8:   return  $j_+ \leftarrow \text{argmax}(\mathbf{r})$ 

```

---



---

**Algorithm 4** ActiveMetaLearningCofiRank

---

```

1: procedure INITIALIZATIONSCHEME( $S$ )
2:    $\mathbf{r} \leftarrow \text{median}(S, 2)$                        $\triangleright$  Column-wise median
3:   return  $j_+ \leftarrow \text{argmax}(\mathbf{r})$ 
4: procedure SELECTNEXT( $S, \mathbf{t}$ )
5:   evaluated  $\leftarrow \text{notNaN}(\mathbf{t})$ 
6:    $\mathbf{r} \leftarrow \text{CofiRank}(S, \mathbf{t})$            $\triangleright$  Collaborative filtering on  $[S; \mathbf{t}]$  returning last line
7:    $\mathbf{r}(\text{evaluated}) \leftarrow -\text{Inf}$ 
8:   return  $j_+ \leftarrow \text{argmax}(\mathbf{r})$ 

```

---

**Algorithm 5** MedianLandmarks1CofiRank

---

```

1: procedure INITIALIZATIONSCHEME( $S$ )
2:    $r \leftarrow \text{median}(S, 2)$                                       $\triangleright$  Column-wise median
3:   return  $j_+ \leftarrow \text{argmax}(r)$ 
4: procedure SELECTNEXT( $S, t$ )
5:    $\text{evaluated} \leftarrow \text{notNaN}(t)$ 
6:   if  $\text{length}(\text{evaluated}) < \text{num\_landmarks}$  then
7:      $r \leftarrow \text{median}(S, 2)$                                       $\triangleright$  Column-wise median
8:   else if  $\text{length}(\text{evaluated}) == \text{num\_landmarks}$  then
9:     static  $r \leftarrow \text{CofiRank}(S, t)$        $\triangleright$  Keep the CofiRank predictions thereafter
10:     $r(\text{evaluated}) \leftarrow -\text{Inf}$ 
11:   return  $j_+ \leftarrow \text{argmax}(r)$ 

```

---

## 4 Benchmark data

To benchmark our proposed method, we gathered datasets from various sources (Table 1). Each dataset consists of a matrix  $S$  of performances of algorithms (or models) on tasks (or datasets). Datasets are in lines and algorithms in columns. The performances were evaluated with a single training/test split or by cross-validation. The tasks were classification or regression tasks and the metrics quasi-homogeneous for each  $S$  matrix (*e.g.* Balanced Accuracy a.k.a. BAC for classification and  $R^2$  for regression). We excluded data sources for which metrics were heterogeneous (a harder problem that we are leaving for further studies). Although ACTIVMETAL lends itself to using sparse matrices  $S$  (with a large fraction of missing values), these benchmarks include only full matrices  $S$ .

The artificial dataset was constructed from a matrix factorization to create a simple benchmark we understand well, allowing to easily vary the problem difficulty. Matrix  $S$  is simply obtained as a product of three matrices  $U\Sigma V$ ,  $U$  and  $V$  being orthogonal matrices and  $\Sigma$  a diagonal matrix of “singular values”, whose spectrum was chosen to be exponentially decreasing, with  $\Sigma_{ii} = \exp(-\beta i)$ ,  $\beta = 100$  in our experiments. The other benchmarks were gathered from the Internet or the literature and represent the performances of real algorithms on real datasets. We brought back all metrics to scores that are “the larger the better”. In one instance (StatLog), we took the square root of the performances to equalize the distribution of scores (avoid a very long distribution tail). For AutoML, many algorithms were aborted due to execution time constraints. We set the corresponding performance to 0. To facilitate score comparisons between benchmark datasets, all  $S$  matrices were globally standardized (*i.e.* we subtracted the global mean and divided by the global standard deviation). This scaling does not affect the results.

We conducted various exploratory data analyses on the benchmark data matrices, including two-way hierarchical clustering, to visualize whether there were enough similarities between lines and columns to perform meta-learning. See our supplemental material referenced at the end of this paper.

**Table 1: Statistics of benchmark datasets used.** #Datasets=number of datasets, #Algo=number of algorithms, Rank=rank of the performance matrix.

|               | Artificial           | Statlog                         | OpenML            | AutoML                            |
|---------------|----------------------|---------------------------------|-------------------|-----------------------------------|
| #Dataset      | 50                   | 21                              | 76                | 30                                |
| #Algo         | 20                   | 24                              | 292               | 17                                |
| Rank          | 20                   | 21                              | 76                | 17                                |
| Metric        | None                 | Error rate                      | Accuracy          | BAC or $R^2$                      |
| Preprocessing | None                 | Take square root                | None              | Scores for aborted algo. set to 0 |
| Source        | Generated by authors | Statlog Dataset in UCI database | Alors [9] website | AutoML1 (2015-2016)               |

## 5 Results

**Table 2: Results of meta-learning methods for all 4 meta-datasets.** Performances of meta-learning algorithms are measured as the area under the meta learning curve (AUMLC) normalized by the area of the best achievable curve. Active\_Meta\_Learning w. CofiRank (our proposed method) performs always best, although not significantly considering the 1-sigma error bars of the leave-one-dataset-out procedure.

|   | Artificial                          | Statlog                               | OpenML                              | AutoML                              |
|---|-------------------------------------|---------------------------------------|-------------------------------------|-------------------------------------|
| <b>Active_Meta_Learning w. CofiRank</b> | <b>0.91 (<math>\pm 0.03</math>)</b> | <b>0.802 (<math>\pm 0.117</math>)</b> | <b>0.96 (<math>\pm 0.04</math>)</b> | <b>0.84 (<math>\pm 0.11</math>)</b> |
| Random                                  | 0.81 ( $\pm 0.05$ )                 | 0.77 ( $\pm 0.05$ )                   | 0.95 ( $\pm 0.03$ )                 | 0.79 ( $\pm 0.07$ )                 |
| SimpleRank w. median                    | 0.7 ( $\pm 0.2$ )                   | 0.798 ( $\pm 0.102$ )                 | 0.95 ( $\pm 0.04$ )                 | 0.82 ( $\pm 0.12$ )                 |
| Median_LandMarks w. 1-CofiRank          | 0.88 ( $\pm 0.04$ )                 | 0.795 ( $\pm 0.099$ )                 | 0.92 ( $\pm 0.08$ )                 | 0.83 ( $\pm 0.11$ )                 |

In this section, we analyze the experimental results of Table 2 and Figure 1. The graphs represent meta-learning curves, that is the performance of the best algorithm found so far as a function of the number of algorithms tried.<sup>2</sup> The ground truth of algorithm performance is provided by the values of the benchmark matrices (see Section 4).

We remind the reader that in a meta-learning problem, each sample is a dataset. To evaluate meta-learning we use the equivalent of a leave-one-out estimator, *i.e.* leave-one-dataset-out. Hence, we use as meta-learning training data

<sup>2</sup> In the future, when we have meta-learning datasets for which the computational run time of algorithms is recorded, we shall tackle the harder and more interesting problem of meta-learning performance as a function of “total” computational time rather than number of algorithms tried.

all datasets but one, then create the learning curve for the left-out dataset. Thus, given a benchmark data matrix, we generate meta-learning curves using as matrix  $S$  a sub-matrix with one line left out (held out), which serves as target vector  $t$  for the dataset tested. Subsequently, we average all meta-learning curves, step-by-step. Thus the result shown in Figure 1 are the averaged learning curves obtained with the leave-one-dataset-out scheme, *i.e.* averaged over all datasets, for a given benchmark dataset.

To evaluate the significance of the efficiency of our proposed method, we ran 1000 times the Random search algorithm, in which algorithms are ran in a random sequence. We drew the curves of median performance (blue curves) and showed as blue shadings various quantiles. The fact that the red curves, corresponding to the proposed algorithm Active Meta Learning w. CofiRank is generally above the blue curve comforts us that the method is actually effective. It is not always significantly better than the median of Random search. However, this is a very hard benchmark to beat. Indeed, the median of Random search is not a practical method, it is the average behavior of random search over many runs. Thus, performing at least as good as the median of Random search is actually pretty good.

We also compared our method with two other baselines. (1) The SimpleRank w. median (green curves) uses the median performance of algorithms on all but the left-out dataset. Thus it does not perform any *active* meta-learning. (2) The Median Landmark w. 1 CofiRank (pink curves) makes only one call of CofiRank to reduce computational expense, based on the performance of only 3 Landmark algorithms, here simply picked based on median ranking.

The first benchmark using artificial data (Figure 1(a)) a relative position of curves that we intuitively expected: SimpleRank w. median (in green) does not perform well and Active Meta Learning w. CofiRank (in red) is way up in the upper quantiles of the random distribution, close to the ideal curve that goes straight up at the first time step (selects right away the best algorithm). Median Landmark w. 1 CofiRank (in pink) quickly catches up with the red curve: this is promising and shows that few calls to CofiRank might be needed, should this become a computational bottleneck.

However, the analysis of the results on real data reveals a variety of regimes. The first benchmark using the datasets of the AutoML challenge (Figure 1(b)) gives results rather similar to artificial data in which Active Meta Learning w. CofiRank still dominates, though SimpleRank w. median performs surprisingly well. More surprisingly, Active Meta Learning w. CofiRank does not beat SimpleRank w. median on the StatLog benchmark and beats it with difficulty (after more than 10% of the algorithms have been trained/tested) on the OpenML benchmark. Also, the cheap algorithm calling CofiRank just once (Median Landmark w. 1 CofiRank, performing no active learning) which looked promising on other benchmark datasets, performs poorly on the OpenML dataset. This is unfortunate since this is the largest dataset, on which running active-learning is most computationally costly. We provide a discussion of computational considerations in Section 6.

Table 2 sums up the results in terms of area under the meta-learning curves (AUMLC). Active Meta Learning w. CofiRank consistently outperforms other methods, although not significantly according to the error bars.

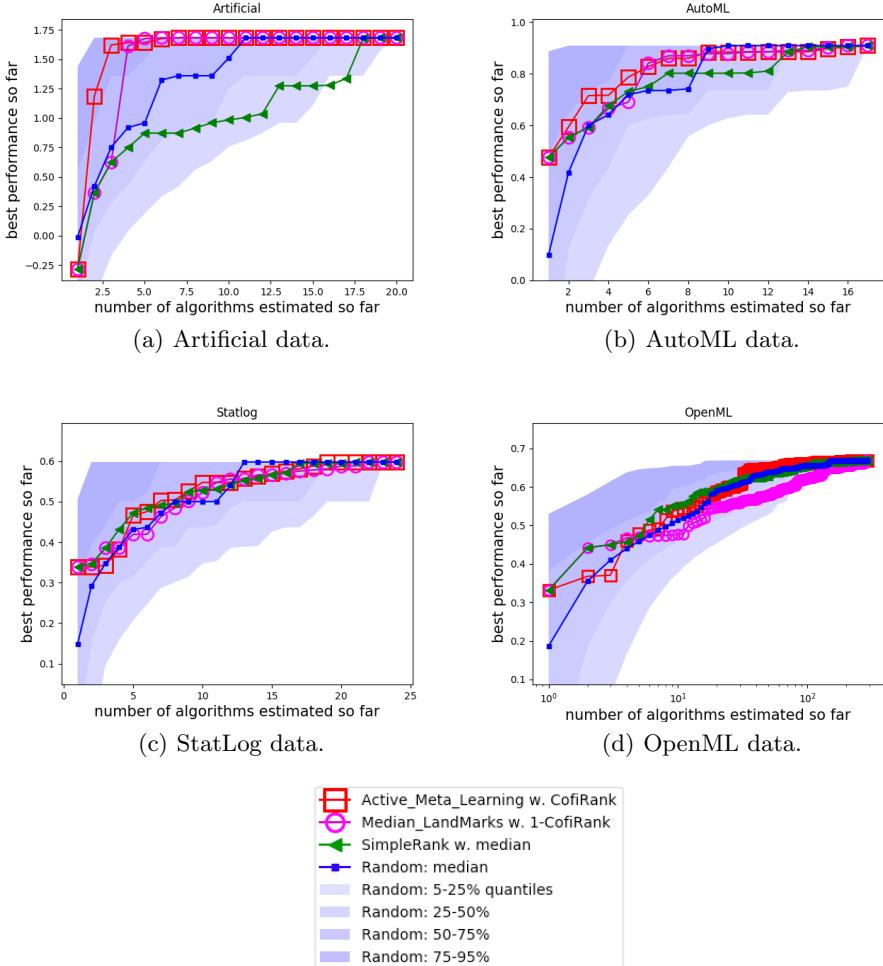
## 6 Discussion and conclusion

We have presented an approach to algorithm recommendation (or model selection) based on meta-learning, capitalizing on previous runs of algorithms on a variety of datasets to rank candidate algorithms and rapidly find which one will perform best on a new dataset. The originality of the paper lies in its active learning approach based on a collaborative-filtering algorithm: CofiRank. Collaborative filtering is a technique to fill in missing data in a collaborative matrix of scores, which in our case represents performances of algorithms on datasets. Starting from the evaluation of a single algorithm on a new dataset of interest, the CofiRank method ranks all remaining algorithms by completing the missing scores in the collaborative matrix for that new dataset. The next most promising algorithm is then evaluated and the corresponding score added to the collaborative matrix. The process is iterated until all missing scores are filled in, by trying all algorithms, or until the allotted time is exhausted.

We demonstrated that Active Meta Learning w. CofiRank performs well on a variety of benchmark datasets. Active Meta Learning w. CofiRank does not always beat the naive SimpleRank w. median baseline method, but it consistently outperforms the “hard-to-beat” median of Random ranking, while SimpleRank w. median does not.

We also investigated whether the (meta-) active learning aspect is essential or can be replaced by running CofiRank a single time after filling in a few scores for Landmark algorithms. This technique (called Median Landmark w. 1 CofiRank) seemed promising on the smallest benchmark datasets, but gives significantly worse results than Active Meta Learning w. CofiRank on the largest benchmark dataset on which it would help most (computationally). One avenue of future research would be to put more effort in the selection of better Landmarks.

Further work also includes accounting for the computational expense of model search in a more refined way. In this work, we neglected the cost of performing meta-learning compared to training and testing the algorithms. This is justified by the fact that their run time is a function of the volume of training data, which is considerably smaller for the collaborative matrix (of dimension usually  $\simeq 100$  datasets times  $\simeq 100$  algorithms) compared to modern-times “big data” datasets (tens of thousands of samples times thousands of features). However, as we acquire larger meta learning datasets, this cost may become significant. Secondly, we assumed that all algorithms had a comparable computational time (to be able to use meta-learning datasets for which this information was not recorded). In the future, we would like to take into account the duration of each algorithm to better trade-off accuracy and computation. It is also worth noting that ACTIVMETAL does not optimize the exploration/exploitation trade-off. It is more geared toward exploitation than exploration since the next best algorithm



**Fig. 1: Meta-learning curves.** We show results of 4 methods on 4 meta-learning datasets, using the leave-one-dataset-out estimator. The learning curves represent performance of the best model trained/tested so far, as a function of the number of models tried. The curves have been averaged over all datasets held-out. The method Active Meta Learning w. CofiRank (red curve) generally dominates other methods. It always performs at least as well as the median of random model selection (blue curve), a hard-to-beat benchmark. The more computationally economical Median Landmark w. 1 CofiRank consisting in training/testing only 3 models (Landmarks) to rank methods using only 1 call to CofiRank (pink curve) generally performs well, except on OpenML data for which it would be most interesting to use it, since this is the largest meta learning datasets. Thus active learning cannot easily be replaced by the use of Landmarks, lest more work is put into Landmark selection. The method SimpleRank w. median that ranks algorithm with their median performance (green curve) is surprisingly a strong contender to Active Meta Learning w. CofiRank for the StatLog and OpenML datasets, which are cases in which algorithms perform similarly on all datasets.

is chosen at every step. Further work may include incorporating monitoring the exploration/exploitation trade-off. In particular, as said, so far we have not taken into account the computational cost of running algorithms. When we have a total time budget to respect, exploring first using faster algorithms then selecting slower (but better) algorithms may be a strategy that ActivMetal could adopt (thus privileging first exploration, then exploitation).

At last, the experiments performed in this paper assumed that, except to the new dataset being tested, there were no other missing values in the collaborative matrix. One of the advantages of collaborative filtering techniques is that they can handle matrices sparsely populated. This deserves further investigation.

## Supplemental material, data and code

For full reproducibility of our results, datasets and code are available on [Github](#). To run it, CofiRank must be installed. We recommend using the Docker [1] image we built for this purpose. Please refer to the Github repository for all instructions. Our repository also includes a Jupyter-notebook with additional graphs referred to in the text.

## References

1. Docker. <https://www.docker.com/>
2. Bennett, J., Lanning, S.: Netflix: The Netflix prize. KDD Cup and Workshop in conjunction with ACM SIGKDD p. 201–206 (2007)
3. Bobadilla, J., Ortega, F., Hernando, A., Gutiérrez, A.: Recommender systems survey. *Knowledge-Based Systems* **46**, 109–132 (2013)
4. Eggensperger, K., Feurer, M., Hutter, F., Bergstra, J., Snoek, J., Hoos, H., Leyton-Brown, K.: Towards an empirical foundation for assessing bayesian optimization of hyperparameters. In: NIPS workshop on Bayesian Optimization in Theory and Practice (2013)
5. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., Hutter, F.: Efficient and robust automated machine learning. In: Proceedings of the Neural Information Processing Systems, pp. 2962–2970 (2015), <https://github.com/automl/auto-sklearn>
6. Feurer, M., Klein, A., Eggensperger, K., Springenberg, J., Blum, M., Hutter, F.: Methods for improving bayesian optimization for automl. In: Proceedings of the International Conference on Machine Learning 2015, Workshop on Automatic Machine Learning (2015)
7. Feurer, M., Springenberg, J., Hutter, F.: Initializing bayesian hyperparameter optimization via meta-learning. In: Proceedings of the AAAI Conference on Artificial Intelligence. pp. 1128–1135 (2015)
8. Gunawardana, A., Meek, C.: Tied boltzmann machines for cold start recommendations. In: Proceedings of the 2008 ACM conference on Recommender systems. pp. 19–26. ACM (2008)
9. Misir, M., Sebag, M.: Alors: An algorithm recommender system. *Artificial Intelligence* **244**, 291–314 (2017)

10. Pan, S.J., Yang, Q.: A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* **22**(10), 1345–1359 (2010)
11. Rice, J.: The algorithm selection problem. *Advances in computers* **15**, 65–118 (1976)
12. van Rijn, J., Bischl, B., Torgo, L., Gao, B., Umaashankar, V., Fischer, S., Winter, P., Wiswedel, B., Berthold, M., Vanschoren, J.: OpenML: A collaborative science platform. In: Blockeel, H., Kersting, K., Nijssen, S., Železný, F. (eds.) *Proceedings of the Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD Part III, LNCS*, vol. 8190, pp. 645–649. Springer (2013)
13. Schein, A.I., Popescul, A., Ungar, L.H., Pennock, D.M.: Methods and metrics for cold-start recommendations. In: *Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*. pp. 253–260. ACM (2002)
14. Srebro, N., Rennie, J., Jaakkola, T.: Maximum-margin matrix factorization. *Advances in neural information processing systems* **17**(5), 1329–1336 (2005)
15. Stern, D., Herbrich, R., Graepel, T., Samulowitz, H., Pulina, L., Tacchella, A.: Collaborative expert portfolio management. In: *AAAI*. pp. 179–184 (2010)
16. Su, X., Khoshgoftaar, T.M.: A survey of collaborative filtering techniques. *Advances in artificial intelligence* **2009**, 4 (2009)
17. Sun-Hosoya, L., Guyon, I., Sebag, M.: Lessons learned from the automl challenge. In: *Conférence sur l’Apprentissage Automatique 2018*. Rouen, France (June 2018)
18. Weimer, M., Karatzoglou, A., Le, Q., Smola, A.: CofiRank-maximum margin matrix factorization for collaborative ranking. In: *Proceedings of the 21st Annual Conference on Neural Information Processing Systems (NIPS)*. pp. 222–230 (2007)

---

# Alignment-Based Topic Extraction Using Word Embedding

Tyler Newman<sup>[0000–0003–3108–8407]</sup> and Paul Anderson<sup>[0000–0002–8408–3944]</sup>

College of Charleston, Charleston SC 29424, USA

[newmantp@g.cofc.edu](mailto:newmantp@g.cofc.edu)

[andersonpe2@cofc.edu](mailto:andersonpe2@cofc.edu)

**Abstract.** Being able to extract targeted topics from text can be a useful tool for understanding the large amount of textual data that exists in various domains. Many methods have surfaced for building frameworks that can successfully extract this topic data. However, it is often the case that a large number of training samples must be labeled properly, which requires both time and domain knowledge. This paper introduces new alignment-based methods for predicting topics within textual data that minimizes the dependence upon a large, properly-labeled training set. Leveraging Word2Vec word embeddings trained using unlabeled data in a semi-supervised approach, we are able to reduce the amount of labeled data necessary during the text annotation process. This allows for higher prediction levels to be attained in a more time-efficient manner with a smaller sample size. Our method is evaluated on both a publicly available Twitter sentiment classification dataset and on a real estate text classification dataset with 30 topics.

**Keywords:** Topic extraction · Text annotation · Text classification · Word vectors · Text tagging

## 1 Introduction

Finding specific topics within textual data is an important task for many domains. A multitude of approaches for achieving this task have appeared in recent years [14], no doubt due to the ever growing amount of textual data available to organizations and researchers. In the most straightforward case, topic labels are known *a priori* and non-domain experts can be trained to manually label examples using systems such as Mechanical Turk [4]. In a second case, topic labels are not known prior to annotation and only after a domain expert has defined them can non-experts be used to label examples. Conversely, once these topic labels have been defined, many domains require an expert throughout the entire annotation process [1, 13]. The fourth case, and the one that motivated our work specifically, is the case in which a domain expert must concurrently evolve the set of topic labels through manual annotation of examples.

In all of these cases, once an appropriate number of training samples have been acquired, many different machine learning algorithms have successfully

been used to automatically identify topics in new examples [14]. Further, reducing the number of training samples needed to produce an accurate predictive model is beneficial in all instances, although the fourth case in particular benefits from a flexible predictive modeling approach that can quickly be retrained on a small number of examples. This is because the domain expert may revise or modify the current topic labels while exploring their problem domain. An example of this scenario arose during a data science project in the real estate domain where a sentence-level topic annotator was desired. Experts in this domain could not specify a final or draft set of topic labels, preventing us from utilizing other approaches that are built to expect a stable set of topics. Further, domain experts required a lightweight, web-based interface where topics could be easily defined, modified, and applied at the sentence level while minimizing the number of training examples needed to produce an automated prediction due to the large number of topics in their domain.

This paper describes a novel method and application for predicting topics at the sentence level in order to reach a high level of accuracy with a limited number of training samples. We evaluate our algorithm on its ability to predict over 30 different sentence-level topic labels within real estate data. We also provide an evaluation of our algorithm on a standard and publicly available twitter sentiment-based prediction dataset. In the real-estate domain, expert knowledge is required for the annotation of the important topics of interest. The topics were not available or known *a priori*, further limiting the number of possible expert annotators. We show that our algorithm achieved a higher prediction accuracy with a very small number of examples, resulting in a significant improvement over standard methods for topic identification with small sample sizes. Finally, our method maintains its ability to predict well in small sample sizes, even in the absence of negative training examples which would further reduce the burden on the domain expert. The rest of this paper is structured as follows: in Section 2 we present a review of related work; in Section 3 we provide descriptions of relevant machine learning algorithms and introduce our architecture and novel methods; and we conclude with Section 6 where we present our experimental results and discussion.

## 2 Related Work

Kim *et al.* proposes a method for categorizing text at the sentence and document level using word vector distances [6]. Their algorithm helps deal with sparsity issues in word data, specifically caused by words looking very different when actually meaning the same thing. These sparsity issues are often caused by short documents or a small amount of training data. One of the datasets that they used was the SemEval 2013 Task B dataset (Twitter), which contains 12,348 tweets that are labeled as positive, negative, or neutral. They found that their algorithm produced good results quickly and with a relatively small number of training samples (requiring at least sample sizes in the hundreds).

Topic and feature extraction is popular in the world of data mining, and there has been a lot of work for making this process more efficient and more accurate. [18] proposes an algorithm that combines binary classifiers, Conditional Random Fields (CRFs), and Expectation Maximization in order to extract information from business-to-consumer (B2C) emails. Different pieces of the extracted information are annotated as specific features. This is done by utilizing templates that have been predefined based on other similar documents. Their approach is fully unsupervised and requires no manual corrections or fixes to the data. Instead of using word vectors for matching, they run low-accuracy annotators trained on weak features which then feed into their CRF model.

Nguyen *et al.* introduces an algorithm that combines preprocessing, pattern recognition, iterative model development, and active learning to annotate and classify features found within clinical text records [13]. In their algorithm, the textual data is first standardized and normalized to perform tasks such as correcting spelling, expanding abbreviations, and converting to a standard layout. The data then moves to the iterative model development process, where models are trained and evaluated using Support Vector Machines (SVMs) and CRFs. The model is refined using a visual annotator that allows for some manual correction, along with active learning that lets the learner select the most informative data to retrain the model. Their approach requires a relatively large number of training samples. In one of their tests, they ran various active learning algorithms on 100 batches of radiology reports with 10 reports per batch. The F-scores for the active learning algorithms, on average, did not surpass 75% until around 7-10 batches (i.e., 70-100 reports) were run. It also took the algorithms between 30-50 batches to reach a 90% F-score.

Wang *et al.* offers a new approach to modeling targeted subtopics within text. Instead of extracting all larger topics within a corpus, they search for more specific subtopics using a *targeted topic model* (TTM) [17]. To do this, each sentence is treated as its own topic that focuses on only one aspect. These topics are deemed relevant or irrelevant based on a set of specified keywords. Their model was run on five datasets taken from Twitter that range in size from 10k to 50k samples.

Finally, several attempts have been made to create architectures that can be used for new domains. [8] proposes a Python framework to ameliorate the process of feature extraction in various different forms of media such as video, audio, and text. Their framework, Pliers, attempts to package the benefits of multiple other machine learning frameworks and services into one coherent feature extraction toolbox.

### 3 Methods

#### 3.1 Standard Approaches

We will now present brief descriptions of several widely applied methods used for target identification. This section is broken up into two additional subsections:

feature extraction and machine learning methods. All of the methods described in this work rely on one of two word embedding methods: bag-of-words (BOW) or Word2Vec [9, 10]. We briefly describe how these methods were implemented and incorporated into our work.

**Feature Extraction** In its simplest form, BOW is an orderless representation of word frequencies in a document. In the context of this sentence-level target identification problem, the word counts from each sentence are normalized into a word frequency matrix prior to classification. The Python natural language toolkit (nltk) and native Python String library [2] were used for this step. Python’s String library was used to parse out punctuation and stop words were removed using nltk. This was followed by stemming using nltk’s SnowballStemmer [2].

Word2Vec is an NLP system that utilizes neural networks in order to create a distributed representation of words in a corpus [9, 10]. While the BOW pipelines produce word frequency for each document respectively, Word2Vec creates vectors for each word present in a document. These vectors have a smaller distance between them for related words. The words Athens and Greece are examples of this, along with pluralities or tense switches, such as alumnus and alumni or walking and walked [10]. In order to map words to vectors, Word2Vec uses an underlying shallow neural network in addition to techniques seen in deep learning tasks. This unsupervised task takes each individual sentence for a given corpus and, within the neural network, encodes the context of word in the sentence, much like the deep learning autoencoders seen in restricted Boltzmann machines and Stacked Denoising Autoencoders [15, 16]. This is done through the usage of skip-grams, which calculate the probabilities of occurrence for words within a certain distance before and after a given word. Inter-relating these probabilities creates similar word vectors for those with higher probabilities.

For the purposes of evaluation in this paper, two Word2Vec models were used. The first was a model trained on the real estate corpus, and the second was a publicly available Twitter Word2Vec pre-trained model available at [http://yuca.test.iminds.be:8900/fgodin/downloads/word2vec\\_twitter\\_model.tar.gz](http://yuca.test.iminds.be:8900/fgodin/downloads/word2vec_twitter_model.tar.gz).

**Machine Learning Methods** Three standard machine learning methods that are often used in topic identification were selected for comparison: Naïve Bayes, SVM, Random Forest (RF), and K-Nearest Neighbor (KNN). Standard implementations of these algorithms are available in scikit-learn. Naïve Bayes was applied to the BOW features [7] using empirical priors and non-parametric settings. SVMs with BOW features have been shown to perform well on a wide range of text classification applications [14]. SVMs are not prone to error with high-dimensional datasets and have been previously shown to be useful in text-based classification problems [5]. Four standard kernels were tested: linear, polynomial, radial basis, and sigmoid. The penalty parameter ( $C$ ) was also varied as 0.01, 0.1, 1, and 10. For the presentation of the results, a single entry for SVMs is displayed that corresponds to the best parameter selection for each class. KNN

with two standard distance metrics was tested. Both distance metrics are built upon Word2Vec embeddings as opposed to BOWs. The two metrics used were the mean and maximum cosine-similarity between all pairs of words.

### 3.2 Novel Approaches

Two novel approaches were developed and evaluated in this work that stem from an alignment-based distance metric. The first approach is a standard KNN classifier utilizing the novel alignment-based distance metric in place of traditional BOW distance metrics. To classify a new unknown sample, the alignment-based KNN calculates the distance between the target sample and all labeled data. The k-nearest neighbors are then found and the majority class is returned. The second approach is an alignment-based threshold classifier that only requires positively labeled data and a predefined threshold. This threshold-based classifier measures the distance from a target unknown sample to only the positively annotated samples. If the score is above the threshold, a positive class prediction is returned. The success of both methods is dependent on the alignment-based distance metric described below.

**Alignment Distance Metric** All alignment-based approaches were implemented using the NeedlemanWunsch algorithm that has been made famous for its use in aligning biological sequences [12]. We have adapted the scoring and gap penalties for the target identification classification problem. The algorithm produces a numerical score based on the aggregation of misalignment penalties between words (cosine-similarity) combined with the penalties for skipping a word in either the labeled or unlabeled sentence. The cosine similarity misalignment score is the dot product between the vector representation of each word. There is no gap penalty for skipping a word in the unlabeled sentence; however, skipping a target word carries a high penalty and is therefore avoided by the algorithm. This forces the algorithm to match all target words that have been identified by a domain expert. An example alignment of two sentences is shown in Table 1. For all alignment-based methods, the Word2Vec implementation described in [10] was used.

**Standard Distance Metrics** In addition to these two novel approaches, we implemented and tested two standard Word2Vec distance metrics. The first metric was the maximum cosine-similarity between pairs of words in annotated and unknown sentences. The second was the average cosine-similarity score between pairs of scores.

## 4 Use Cases

### 4.1 Real Estate

The domain we designed our original system for involved property descriptions from real estate data. This data was created by realtors and contains information

Table 1: Example alignment of two sentences with an aggregate alignment score of 0.91. The first sentence has been annotated as a positive example of the Master Bedroom Downstairs target class. The second sentence has no known annotation. The optimal alignment shown results in a score of 0.91 with misalignment occurring between bedroom and bedrooms. Gaps in the annotated sentence matched to words in the unknown sentence are not penalized.

---

Annotated: — master bedroom — —— downstairs  
Unknown: two master bedrooms are located downstairs

---

about various real estate listings. Each listing has a large amount of metadata (e.g., location, images, basic features); however, the main piece of information we use in our system is the description written for each listing. This description is usually no more than a short paragraph and is created by the real estate agent to summarize the listing as a whole. It can include any information that the agent wishes to convey to the potential buyer, such as property features, kitchen appliances, etc. By analyzing a listing’s description, we attempt to extract specific features about the listing itself.

We created a web application using the Angular JavaScript framework. The main interface for this application provides a way to easily and quickly annotate real estate listings. This interface pulls listings that have not yet been annotated from our server, along with the sentence and word data associated with that listing. It also retrieves the topic information so that the listing can be properly annotated.

Once a listing has been retrieved, the annotator can select a sentence from the listing description to annotate. They can then toggle the individual words in the sentence that match a specific topic. If they believe that a pattern should be mapped to a topic that does not currently exist, then they have the option to create it. Once a topic has been defined, it can then be used by future annotators. When an annotator is satisfied with an annotation, they simply click submit and it is stored inside of the annotations database to be used in future alignment predictions.

Along with this annotation interface, we also included other pages within the web application that provide statistical information regarding the database. Some of these pages display simple information, such as the specific progress of different annotators. Other pages give annotators more control over the database itself, allowing them to take actions such as correcting mistakes in previous annotations or modifying topic names.

## 4.2 Twitter

Twitter has grown quite large in the past decade, along with the amount of textual data it has created. This data has proven to be a popular source for test-

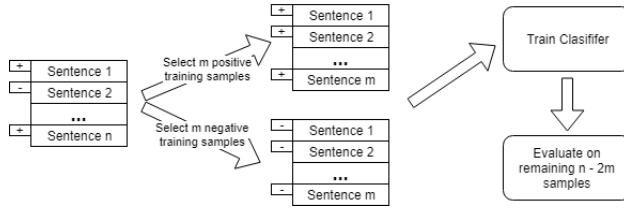


Fig. 1: One iteration of cross-validation. This was repeated 20 times in total for training set sizes 2 to  $(n/2 - 2)$

ing and implementing text-based machine learning algorithms [3, 6, 11, 17]. We tested our alignment algorithm on the Twitter dataset used in [6]. This dataset includes the text data for 12,348 tweets and annotations that have been made for each tweet. These annotations are based on the entire tweet and represent the overall sentiments conveyed therein. The possible sentiment values include positive, neutral, negative, objective, and objective-OR-neutral. In our experiment and the experiment done by [6], only positive, neutral, and negative sentiments are used.

In order to annotate this dataset, we created a simple interactive widget that runs inside of a Python Jupyter Notebook. This publicly available widget is very similar to the annotation interface we created for the real estate data and can be seen in supplemental Figure 4. We store the tweets in a text file, which the interface uses to select tweets at random. Once a tweet is selected, the annotator can toggle the words in the tweet that they believe match the sentiment that was predicted. For example, if a tweet was labeled as having a positive sentiment, then only words that are relevant to that sentiment will be selected and stored in the annotation. This allows us to store annotations in a very similar format to the ones that were stored for our real estate data.

## 5 Evaluation

Each classifier was subjected to iterative cross-validation as a function of the training size for each category. This procedure is summarized in Figure 1. The average  $F_1$  score across all iterations for each training set size was calculated and plotted as a function of training set size. Two examples are shown in Figure 2. The area under this curve was then calculated to measure the ability of each classifier to perform well for small as well as larger training set sizes. The 95% confidence interval was calculated for the area under the  $F_1$  curve as a function of the training set size.

## 6 Results and Discussion

We compared our alignment-based algorithms to bag-of-words derived SVM, Naïve Bayes, random forest, and KNN classifiers on 30 real estate categories

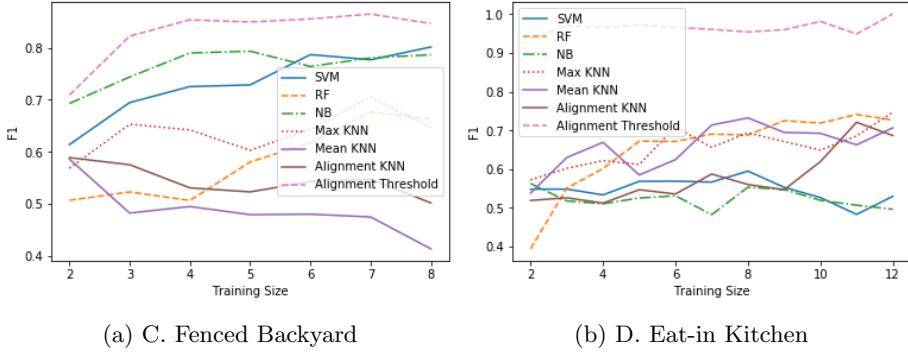


Fig. 2: Two sample categories showing the  $F_1$  score versus the training set size for the five different classifiers tested.

and positive/negative sentiment tweet prediction. Four algorithms based on Word2Vec word embedding were evaluated. Two standard distance metrics (mean and max) were evaluated as baseline classifiers in addition to two novel alignment-based classifiers (Alignment KNN and Alignment Threshold). These results are summarized in Table 2 which shows the iterative cross-validation confidence intervals of the area under the  $F_1$  curve versus training set size. The table is sorted by the average score for the Alignment Threshold method. The first column in the table represents a summary of how this method performed in comparison to the others. If the Alignment Threshold method had a higher and non-overlapping confidence interval when compared to all other methods, this is indicated with a  $W+$ . If the lower bound on the Alignment Threshold method was higher than any other method and *does* overlap, this is indicated with a  $W$ . A  $T$  was used if the Alignment Threshold confidence interval was not higher than any other method but still overlapped the best method. All other cases are indicated with an  $L$ . In total, there are 29%  $W+$ , 35%  $W$ , 19%  $T$ , and 16%  $L$ , meaning that the Alignment Threshold method was as good or better in 84% of the classes. These results demonstrate how the alignment-based algorithms are able to perform better on fewer samples or equivalent to standard approaches. This is significant even in the case of equivalent accuracy as the alignment-based threshold algorithm does not require negative training samples which reduces the number of samples an expert must annotate. Further, all alignment-based methods are built upon a semi-supervised learning approach where large amounts of unlabeled data is used to reduce the need for labeled data.

It was our desire to test our algorithm against the multi-level kernel system developed by Kim *et al.* discussed in section 2; however, we were unable to find a public implementation of this algorithm, and our attempts to reach the authors were not successful. In their paper, the authors evaluated their algorithm using the Twitter dataset also used in this paper [11], which includes 12,348 tweets that are each matched with an overall sentiment value. The multi-level kernel system

was able to achieve accuracy between around 80% and 90% using samples sizes ranging from the low hundreds to the mid thousands, with their best accuracy result being 0.808 (with a standard test data split of 25%). No results were presented for sample sizes less than 50.

Further inspection of Table 2 shows that in some cases the Alignment Threshold classifier performs poorly while the Alignment KNN classifier, which uses both positive and negative training examples, performs better or equivalent to standard approaches. We believe this is due to the underlying word embedding vectors for the specific targets. Words such as those found in positive and negative tweets are relatively ubiquitous, while those found when mentioning a walk-in closet are relatively rare. It is reasonable to assume that hard coded rules could also be developed in some cases, but that the approach presented in this paper would be preferred as it is easily extended to additional categories without the need to maintain a rule management system.

The original annotation system was researched and built specifically for a company specializing in real estate technology. Because this data is proprietary, our algorithms are also tested our algorithm on a publicly available Twitter sentiment prediction dataset. The original Javascript Angular based system is proprietary, but we have implemented the core functionality of our system using Jupyter Notebook widgets. The widget loads directly inside of the Notebook and displays a tweet’s text, its overall sentiment value, and buttons that represent each word in the tweet. These buttons can be toggled to create the annotation pattern. All of this data is stored in a Pandas DataFrame and serialized to its own file, which can be used to train and evaluate our algorithm. We have made this version of the annotation interface open source in addition to all alignment-based implementations and evaluation code (<https://github.com/Anderson-Lab/sentence-annotation>).

## 7 Conclusions

Being able to extract topics from text is an ever-growing problem for many domains given the large amount of textual data that is constantly being created. Therefore, it is necessary to minimize the amount of annotating needed to achieve high levels of accuracy. To accomplish this, we introduced a novel topic prediction algorithm that requires only a small amount of human annotation. Our results show that this approach can provide significant performance benefits when the target labels are not known *a priori* or when the sample size is small. Future directions of this work include experiments to determine if these alignment-based distance metrics continue to provide non-redundant benefits as the sample size grows significantly. For community and reproducibility purposes, our methods are available in a public repository that includes a Jupyter Notebook annotation widget that allows for annotation to easily be carried out on other real world datasets.

Table 2: Area under the  $F_1$  vs. training set size curve. 95% confidence intervals are calculated for each method. Categories are ordered by the average  $F_1$  Alignment Threshold score.

| Category                       | # Samples | SVM         | RF          | NB          | Max KNN     | Mean KNN    | Align.      | KNN Align. | Threshold   |
|--------------------------------|-----------|-------------|-------------|-------------|-------------|-------------|-------------|------------|-------------|
| W+ Screened Porch              | 21        | 0.6 - 0.68  | 0.74 - 0.83 | 0.63 - 0.71 | 0.67 - 0.76 | 0.69 - 0.76 | 0.72 - 0.79 |            | 0.88 - 0.9  |
| W Walk in closet               | 25        | 0.78 - 0.83 | 0.83 - 0.88 | 0.75 - 0.81 | 0.67 - 0.76 | 0.56 - 0.64 | 0.68 - 0.75 |            | 0.88 - 0.9  |
| W+ Granite Countertops         | 31        | 0.62 - 0.69 | 0.64 - 0.73 | 0.64 - 0.71 | 0.6 - 0.7   | 0.7 - 0.79  | 0.57 - 0.67 |            | 0.86 - 0.9  |
| W Stainless Steel Appliances   | 20        | 0.7 - 0.76  | 0.77 - 0.85 | 0.74 - 0.8  | 0.62 - 0.71 | 0.51 - 0.59 | 0.7 - 0.77  |            | 0.81 - 0.85 |
| W+ Eat-in Kitchen              | 11        | 0.41 - 0.5  | 0.5 - 0.61  | 0.39 - 0.48 | 0.5 - 0.59  | 0.49 - 0.61 | 0.43 - 0.53 |            | 0.79 - 0.82 |
| T Near beach                   | 17        | 0.73 - 0.79 | 0.69 - 0.77 | 0.75 - 0.81 | 0.63 - 0.72 | 0.8 - 0.84  | 0.77 - 0.84 |            | 0.76 - 0.82 |
| W+ Sunroom                     | 10        | 0.46 - 0.55 | 0.51 - 0.63 | 0.52 - 0.62 | 0.53 - 0.62 | 0.48 - 0.62 | 0.52 - 0.63 |            | 0.77 - 0.8  |
| W Open Floor Plan              | 22        | 0.73 - 0.8  | 0.73 - 0.82 | 0.66 - 0.73 | 0.62 - 0.72 | 0.56 - 0.65 | 0.75 - 0.83 |            | 0.76 - 0.81 |
| W+ Dual Vanities               | 10        | 0.6 - 0.69  | 0.58 - 0.69 | 0.58 - 0.68 | 0.45 - 0.56 | 0.51 - 0.6  | 0.48 - 0.6  |            | 0.77 - 0.8  |
| W Crown molding                | 13        | 0.63 - 0.7  | 0.45 - 0.56 | 0.49 - 0.59 | 0.58 - 0.68 | 0.71 - 0.79 | 0.58 - 0.68 |            | 0.75 - 0.81 |
| W+ Heart of Pine Flooring      | 12        | 0.59 - 0.67 | 0.58 - 0.7  | 0.59 - 0.67 | 0.54 - 0.65 | 0.65 - 0.62 | 0.71 - 0.71 |            | 0.73 - 0.81 |
| W+ Breakfast Bar               | 8         | 0.57 - 0.65 | 0.52 - 0.63 | 0.58 - 0.63 | 0.66 - 0.45 | 0.56 - 0.43 | 0.55 - 0.47 |            | 0.75 - 0.77 |
| W Built-In Cabinets            | 16        | 0.64 - 0.72 | 0.64 - 0.74 | 0.61 - 0.69 | 0.6 - 0.69  | 0.63 - 0.73 | 0.66 - 0.74 |            | 0.72 - 0.79 |
| T Tennis Courts                | 9         | 0.62 - 0.7  | 0.57 - 0.7  | 0.53 - 0.63 | 0.75 - 0.79 | 0.68 - 0.74 | 0.75 - 0.79 |            | 0.7 - 0.77  |
| T Pool                         | 10        | 0.62 - 0.7  | 0.58 - 0.68 | 0.69 - 0.75 | 0.68 - 0.74 | 0.61 - 0.69 | 0.62 - 0.7  |            | 0.69 - 0.75 |
| W Garden Tub                   | 7         | 0.6 - 0.67  | 0.6 - 0.69  | 0.38 - 0.49 | 0.44 - 0.55 | 0.28 - 0.37 | 0.38 - 0.49 |            | 0.69 - 0.73 |
| W StructureUnit Brick Exterior | 7         | 0.62 - 0.7  | 0.48 - 0.6  | 0.58 - 0.65 | 0.55 - 0.63 | 0.43 - 0.53 | 0.49 - 0.59 |            | 0.69 - 0.73 |
| T Cul-De-Sac                   | 8         | 0.62 - 0.69 | 0.63 - 0.71 | 0.59 - 0.68 | 0.68 - 0.74 | 0.63 - 0.71 | 0.67 - 0.72 |            | 0.68 - 0.74 |
| W+ Vaulted Ceiling             | 8         | 0.43 - 0.52 | 0.5 - 0.61  | 0.48 - 0.58 | 0.48 - 0.58 | 0.55 - 0.63 | 0.45 - 0.56 |            | 0.68 - 0.73 |
| L Near Shopping                | 23        | 0.84 - 0.88 | 0.63 - 0.73 | 0.78 - 0.83 | 0.73 - 0.8  | 0.86 - 0.89 | 0.76 - 0.82 |            | 0.66 - 0.74 |
| W Open Kitchen                 | 10        | 0.46 - 0.55 | 0.55 - 0.67 | 0.5 - 0.61  | 0.44 - 0.55 | 0.36 - 0.47 | 0.36 - 0.45 |            | 0.65 - 0.73 |
| W Gas Log Fireplace            | 9         | 0.57 - 0.66 | 0.58 - 0.69 | 0.46 - 0.56 | 0.47 - 0.59 | 0.43 - 0.58 | 0.53 - 0.63 |            | 0.61 - 0.71 |
| W Master Bedroom Downstairs    | 8         | 0.48 - 0.59 | 0.46 - 0.6  | 0.35 - 0.46 | 0.44 - 0.56 | 0.42 - 0.52 | 0.54 - 0.63 |            | 0.61 - 0.7  |
| T Natural Light                | 15        | 0.63 - 0.71 | 0.5 - 0.6   | 0.59 - 0.68 | 0.53 - 0.64 | 0.62 - 0.7  | 0.62 - 0.72 |            | 0.6 - 0.68  |
| L Near Downtown                | 15        | 0.67 - 0.74 | 0.69 - 0.78 | 0.82 - 0.85 | 0.71 - 0.78 | 0.68 - 0.75 | 0.74 - 0.8  |            | 0.59 - 0.68 |
| W+ StructureUnit Deck          | 8         | 0.43 - 0.54 | 0.39 - 0.5  | 0.43 - 0.52 | 0.42 - 0.53 | 0.37 - 0.5  | 0.42 - 0.54 |            | 0.59 - 0.67 |
| W Fenced Backyard              | 7         | 0.51 - 0.6  | 0.37 - 0.5  | 0.53 - 0.62 | 0.43 - 0.53 | 0.32 - 0.41 | 0.35 - 0.47 |            | 0.59 - 0.67 |
| L Gas Range                    | 9         | 0.5 - 0.59  | 0.47 - 0.6  | 0.55 - 0.63 | 0.54 - 0.65 | 0.68 - 0.75 | 0.42 - 0.52 |            | 0.57 - 0.66 |
| T tweets                       | 23        | 0.53 - 0.61 | 0.39 - 0.48 | 0.49 - 0.58 | 0.43 - 0.52 | 0.45 - 0.54 | 0.52 - 0.6  |            | 0.52 - 0.6  |
| L Near Restaurants             | 7         | 0.61 - 0.67 | 0.49 - 0.62 | 0.5 - 0.59  | 0.61 - 0.67 | 0.59 - 0.65 | 0.6 - 0.67  |            | 0.49 - 0.61 |
| L Professionally Landscaped    | 7         | 0.45 - 0.55 | 0.32 - 0.41 | 0.41 - 0.51 | 0.49 - 0.59 | 0.37 - 0.47 | 0.41 - 0.51 |            | 0.38 - 0.47 |

## References

1. Andrzejewski, D., Zhu, X., Craven, M.: Incorporating domain knowledge into topic modeling via Dirichlet Forest priors. In: Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09. pp. 1–8 (2009). <https://doi.org/10.1145/1553374.1553378>, <http://portal.acm.org/citation.cfm?doid=1553374.1553378>
2. Bird, S., Klein, E., Loper, E.: Natural language processing with Python. No. January 2009, O'Reilly Media, Inc. (2009). <https://doi.org/10.17509/ijal.v1i1.106>, <https://books.google.com/books?id=KGIBfiiP1i4C{\&}pg=PR5>
3. Boella, G., Caro, L.D., Ruggeri, A., Robaldo, L.: Learning from syntax generalizations for automatic semantic annotation. Journal of Intelligent Information Systems **43**(2), 231–246 (2014). <https://doi.org/10.1007/s10844-014-0320-9>
4. Buhrmester, M., Kwang, T., Gosling, S.D.: Amazon's mechanical Turk: A new source of inexpensive, yet high-quality, data? Perspectives on Psychological Science **6**(1), 3–5 (2011). <https://doi.org/10.1177/1745691610393980>
5. Joachims, T.: Text categorization with support vector machines: Learning with many relevant features. In: Lecture Notes in Computer Science (including sub-series Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). vol. 1398, pp. 137–142 (1998). <https://doi.org/10.1007/s13928716>
6. Kim, J., Rousseau, F., Vazirgiannis, M.: Convolutional Sentence Kernel from Word Embeddings for Short Text Categorization. Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (September), 775–780 (2015)
7. Lewis, D.D.: Naive (Bayes) at forty: The independence assumption in information retrieval pp. 4–15 (1998). <https://doi.org/10.1007/BFb0026666>, <http://link.springer.com/10.1007/BFb0026666>
8. McNamara, Q., de la Vega, A., Yarkoni, T.: Developing a comprehensive framework for multimodal feature extraction. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (2017). <https://doi.org/10.1145/3097983.3098075>
9. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient Estimation of Word Representations in Vector Space. CoRR **abs/1301.3**, 1–12 (2013). <https://doi.org/10.1162/153244303322533223>, <http://arxiv.org/abs/1301.3781>
10. Mikolov, T., Yih, W.t., Zweig, G.: Linguistic regularities in continuous space word representations. Proceedings of NAACL-HLT (June), 746–751 (2013). <https://doi.org/10.3109/10826089109058901>, <http://www.aclweb.org/anthology/N13-1090>
11. Nakov, P., Rosenthal, S., Ritter, A., Wilson, T.: SemEval-2013 Task 2: Sentiment Analysis in Twitter. Proceedings of the International Workshop on Semantic Evaluation (SemEval-2013) **2**(SemEval), 312–320 (2013)
12. Needleman, S.B., Wunsch, C.D.: A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of Molecular Biology **48**(3), 443–453 (1970). [https://doi.org/10.1016/0022-2836\(70\)90057-4](https://doi.org/10.1016/0022-2836(70)90057-4)
13. Nguyen, H., Patrick, J.: Text Mining in Clinical Domain. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16 pp. 549–558 (2016). <https://doi.org/10.1145/2939672.2939720>, <http://dl.acm.org/citation.cfm?doid=2939672.2939720>
14. Pawar, P., Gawande, S.: A comparative study on different types of approaches to text categorization. International Journal of Machine Learning and Computing **2**(4), 423–426 (2012). <https://doi.org/10.7763/IJMLC.2012.V2.158>

15. Srivastava, Salakhutdinov: Multimodal Learning with Deep Boltzmann Machines. Advances in neural information processing systems (NIPS) **15**, 2222–2230 (2012). <https://doi.org/10.1109/CVPR.2013.49>
16. Vincent PASCALVINCENT, P., Larochelle LAROCHEH, H.: Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion Pierre-Antoine Manzagol. Journal of Machine Learning Research **11**, 3371–3408 (2010). <https://doi.org/10.1111/1467-8535.00290>
17. Wang, S., Chen, Z., Fei, G., Liu, B., Emery, S.: Targeted Topic Modeling for Focused Analysis. In: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '16 (2016). <https://doi.org/10.1145/2939672.2939743>
18. Zhang, W., Ahmed, A., Yang, J., Josifovski, V., Smola, A.J.: Annotating Needles in the Haystack without Looking. Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining - KDD '15 pp. 2257–2266 (2015). <https://doi.org/10.1145/2783258.2788580>, <http://dl.acm.org/citation.cfm?doid=2783258.2788580>

## 8 Supplemental

Our work was directly motivated to reduce the burden of interaction with human supervisors, and therefore, we present an overview of our architecture that can be broken up into four distinct phases: preprocessing, annotation, learning, and evaluation. Figure 3 illustrates these phases and the steps that are taken in each.

**Preprocessing & Cleaning** The first step in our pipeline involves processing the raw listing description data and converting it into a consistent format. Each description is stripped of any extra whitespace, transformed into all lowercase letters, and decoded using the UTF-8 character encoding. Once this has been done, the description is separated into its constituent sentences. These sentences are then broken up and become lists of words that can be used in our alignment algorithm.

**Annotation** In order to expedite the annotation process, we made two simple annotation interfaces; one for each of the datasets that we tested our algorithm on. A screenshot of the Jupyter widget is shown in Figure 4.

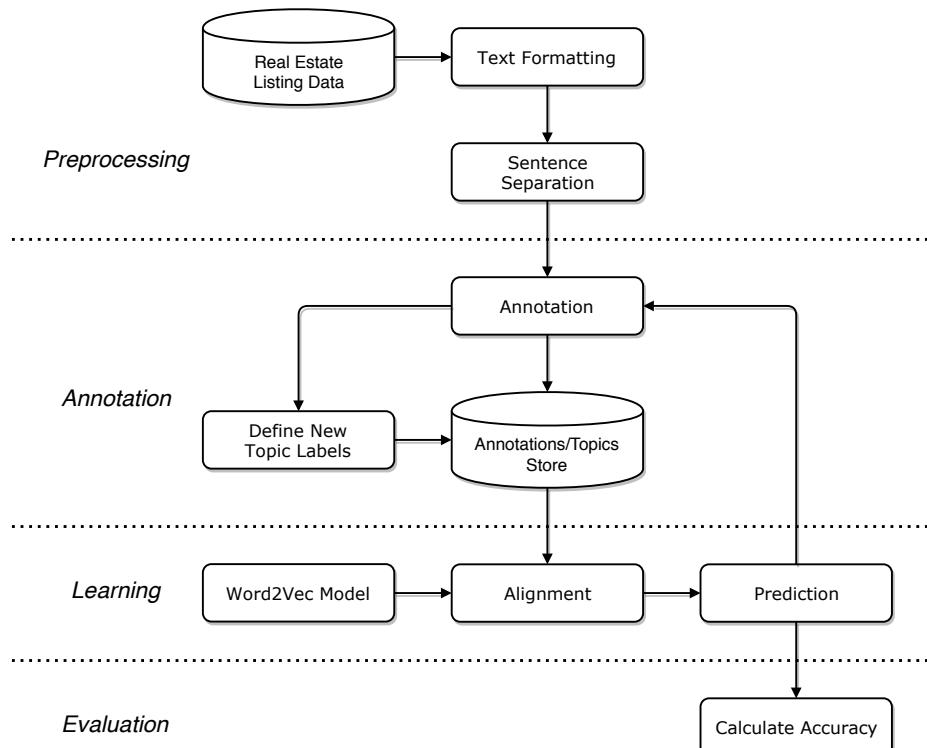


Fig. 3: Architecture Pipeline

```
In [9]: run_annotator(ORIGINAL_FILE_NAME, MASK_FILE_NAME)
```

"positive"

DRY by PJ Harvey is a brilliant record. I listened to the  
whole thing yesterday on a break from editing. #filmmaking  
<http://bit.ly/pu4ud5>

Create Mask

Fig. 4: Python Tweet Annotation Tool

---

# Human-Technology relations in a machine learning based commuter app.\*

Lars Holmberg<sup>1</sup>

Faculty of Technology and Society, Malmö University, Sweden  
Internet of Things and People Research Center, Malmö University, Sweden  
[Lars.Holmberg@mau.se](mailto:Lars.Holmberg@mau.se)

## 1 Introduction

Artifacts that take advantage of Machine Learning (ML) influence our daily life to an increasing extent. The development and use of these artifacts often leaves out the human actors and the context and thus risks to become technocentric [6].

Many ML techniques, for example active learning, interactive learning or machine teaching, calls for user involvement [3, 10, 12, 13, 15]. To our knowledge there is a lack of research concerning relations between humans and ML-based artifacts, the attention is instead often on the desired outcome [4, 11], for example trust.

If we are to understand how ML can support humans, we need to turn attention to the human-technology relations. In this work we will use a postphenomenological lens and focus on how different ML techniques can create, build and maintain relations between humans and technology [5, 14].

Our work is explorative and use material sketching to produce artifactual knowledge [7], methodologically this fits into a Research Through Design approach [16].

In this work, our artifact and the relevance of a postphenomenological approach is the main contribution. In the work presented here the focus is on adaptive learning in a background relation [5] and identifying situations where the background relation can promote a transition to another type of relation (embodied, hermeneutic, alterity). We hope to initiate a discussion on this approach and inspire further work by our contribution.

In this extended abstract, we will focus on the artifact and its relevance for human-technology relations and only briefly expand on application context, theory and methodology.

## 2 Related research and Methodology

As a blueprint for this work we have followed the directions of Ohlin et al. [9] and we aim at promoting transitions between relations initiated either by the

---

\* This work partially financed by the Knowledge Foundation through the Internet of Things and People research profile. I would also like to thank Paul Davidsson and Carl Magnus Olsson for invaluable support during the work.

artifact or the user. We start by creating an adaptive ML-backend for an existing commuter app<sup>1</sup><sup>2</sup>. The ML-backend predicts and presents the users next journey when the app is started, the prediction is based on historical individual travel patterns and the users ML-model is trained online. To create a baseline for our predictions, personas based on a local transportation company are used [8].

### 3 Result and discussion

In our process to create an artifact that meets our initial design goals regarding: prediction accuracy, cloud service cost and performance, we have iterated and re-framed our artifact multiple times. The users trust in the predictions is central to be able to build relations, but trust doesn't automatically follow prediction accuracy [4]. This implies an adaptive backend in the background that delivers reasonable accurate predictions for data sets that reflects use over weeks, months and year.

The current artifact iteration is a solution that extends the Android commuter app and adds a backend based on Google Cloud services (Firebase, BigQuery, compute engine), TensorFlow estimators [1, 2] and Node.js<sup>3</sup>. With our personas in focus we created one travel pattern that is periodic, one that drifts over time and one more random. Based on this data an adaptive ML-backend was created that uses different ML-Algorithms depending on amount of labeled data and noise in input data. The resulting system delivers journey predictions to the app in an acceptable time-frame (less than 2 seconds) and identifies situations where transition to another type of relation is appropriate.

The main contribution in this work is identifying situations where there is a need for a transition to a new type of human-technology relation. Since the work presented here builds a background relation the transitions calls for an ML-technology that allows or invites users to get involved.

### 4 Conclusion

This extended abstract set out to turn attention to the human-technology relations ML-artifacts promote. We have done that by using the vocabulary of post-phenomenology and by exploring the design space created by a commuter app. By specifically focusing on adaptive learning in a background relation and identifying situations that calls for a shift in type of relation (embodied, hermeneutic, alterity) we have laid a base for future research.

In terms of future research we particularly suggest to include active learning, interactive learning and machine teaching to explore the relations these techniques promote.

---

<sup>1</sup> <https://skanependlaren.firebaseio.com>

<sup>2</sup> <https://play.google.com/store/apps/details?id=se.k3larra.alvebuss>

<sup>3</sup> <https://github.com/k3larra/commuter/>

## References

1. Cheng, H.T., Haque, Z., Hong, L., Ispir, M., Mewald, C., Roumpos, G., Sculley, D., Smith, J., Soergel, D., Tang, Y., Tucker, P., Wicke, M., Xia, C., Xie, J.: TensorFlow Estimators: Managing Simplicity vs. Flexibility in High-Level Machine Learning Frameworks. In: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. pp. 1763–1771 (2017). <https://doi.org/10.1145/3097983.3098171>, <https://terrytangyuan.github.io/data/papers/tf-estimators-kdd-paper.pdf>
2. Cheng, H.T., Koc, L., Harmsen, J., Shaked, T., Chandra, T., Aradhye, H., Anderson, G., Corrado, G., Chai, W., Ispir, M., Anil, R., Haque, Z., Hong, L., Jain, V., Liu, X., Shah, H.: Wide & Deep Learning for Recommender Systems. In: Proceedings of the 1st Workshop on Deep Learning for Recommender Systems. pp. 7–10 (2016). <https://doi.org/10.1145/2988450.2988454>, <http://arxiv.org/abs/1606.07792>
3. Fails, J.A., Olsen, D.R.: Interactive machine learning. Proceedings of the 8th international conference on Intelligent user interfaces - IUI '03 p. 39 (2003). <https://doi.org/10.1145/604045.604056>, <http://dl.acm.org/citation.cfm?doid=604045.604056>
4. Glass, A., McGuinness, D.L., Wolverton, M.: Toward establishing trust in adaptive agents. Proceedings of the international conference on Intelligent user interfaces (IUI '08) pp. 227–236 (2008). <https://doi.org/10.1145/1378773.1378804>
5. Ihde, D.: Technology and the Lifeworld, vol. 1 (1990). <https://doi.org/10.1049/et:20060114>
6. Johnson, D.G., Verdicchio, M.: Reframing AI Discourse. *Minds and Machines* **27**, 575–590 (2017). <https://doi.org/10.1007/s11023-017-9417-6>
7. Löwgren, J.: On the significance of making in interaction design research. *Interactions* **23**(3), 26–33 (2016). <https://doi.org/10.1145/2904376>
8. Nielsen, L.: Personas - User Focused Design. Springer (2013). <https://doi.org/10.1007/978-1-4471-4084-9>, <http://link.springer.com/10.1007/978-1-4471-4084-9>
9. Ohlin, F., Olsson, C.M.: Beyond a utility view of personal informatics. Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers - UbiComp '15 pp. 1087–1092 (2015). <https://doi.org/10.1145/2800835.2800965>
10. Porter, R., Theiler, J., Hush, D.: Interactive machine learning in data exploitation. *Computing in Science and Engineering* **15**(5), 12–20 (2013). <https://doi.org/10.1109/MCSE.2013.74>
11. Ribeiro, M.T., Singh, S., Guestrin, C.: "Why Should I Trust You?": Explaining the Predictions of Any Classifier (feb 2016). <https://doi.org/10.1145/2939672.2939778>, <http://arxiv.org/abs/1602.04938>
12. Simard, P.Y., Amershi, S., Chickering, D.M., Pelton, A.E., Ghorashi, S., Meek, C., Ramos, G., Suh, J., Verwey, J., Wang, M., Wernsing, J.: Machine Teaching: A New Paradigm for Building Machine Learning Systems. Tech. rep., Microsoft Research (2017), <http://arxiv.org/abs/1707.06742>
13. Smith, A., Kumar, V., Boyd-Graber, J., Seppi, K., Findlater, L.: Closing the loop: User-centered design and evaluation of a human-in-the-loop topic modeling system. In: International Conference on Intelligent User Interfaces, Proceedings IUI. pp. 293–304 (2018). <https://doi.org/10.1145/3172944.3172965>, <http://cs.colorado.edu/>

*Human-Technology Relations in a Machine Learning Based Commuter App*

---

14. Verbeek, P.P.: What things do. No. 9 (2013).  
<https://doi.org/10.1017/CBO9781107415324.004>
15. Zhu, X., Singla, A., Zilles, S., Rafferty, A.N.: An Overview of Machine Teaching (2018), <http://arxiv.org/abs/1801.05927>
16. Zimmerman, J., Forlizzi, J., Evenson, S.: Research Through Design as a Method for Interaction Design Research in HCI design research in HCI. Proceedings of the SIGCHI conference on Human factors in computing systems pp. 493–502 (2007). <https://doi.org/http://doi.acm.org/10.1145/1240624.1240704>, <http://repository.cmu.edu/hcii>

---

# An Interactive Learning Scenario for Real-time Environmental State Estimation Based on Heterogeneous and Dynamic Sensor Systems

Agnes Tegen, Paul Davidsson, and Jan A. Persson

Malmö University, School of Technology, Sweden  
Internet of Things and People Research Center  
`{firstname.lastname}@mau.se`

With the ongoing advances in the area of Internet of Things, the number of devices with sensors streaming data in our surroundings is growing rapidly. This will create new possibilities in continuously monitoring the state of the environment. However, this increasingly more complex setting is also posing new challenges, e.g. how to properly fuse data from different types of sensors with uncertain availability.

We are focusing on a setting where the task is to do real-time continuous estimations of certain aspects of the state of an environment. These estimations are based on data streams from a heterogeneous and dynamic set of sensors in that environment. Typically, data from different types of sensors needs to be fused in order estimate the aspect. For instance, within an office setting this could be what type of activity is currently taking place in a room or the number of people in a certain area of a building. In previous work [1], the concept of dynamic intelligent virtual sensors was suggested as a framework for data fusion. Common for many scenarios of this type, is that there is no available model that fuses the data and estimates the desired aspect of the state of the environment. Thus, such a model needs to be learned based on the streamed data provided by the sensors. Although the sensors may generate large amounts of data, there is typically a lack of labeled data that can be used for supervised learning.

The interactive learning challenge described above has been identified within an ongoing project with a number of industrial partners, partially validating its relevance to many real world applications.

## 1 Application Scenario Description

In a given environment, we define the set of all sensors that generates data as  $S = \{s_1, s_2, \dots\}$ . While all sensors in  $S$  produce at least one instance of data, note that they do not necessarily have available data at all times. We also introduce a set  $S_t \subseteq S$ , containing all sensors from which data is available regarding the current point in time  $t$ . The data is generated by the sensors in a sequential fashion. Each instance of data contains the following information: *id* (unique identifier for the sensor), *data* (numerical or categorical measurement of the environment), *ts* (timestamp, the point in time when the data was measured according to the device).

We define a set of states,  $Y$ , representing the possible values of the aspect of the environment that we are interested in. If the entire state set is known beforehand, it can be defined along with the task. If not, the state set has to be defined over time, as labeled data becomes available. The labels provided by a user, denoted  $y_{ts} \in Y$ , may be used as training data and can be seen as ground truth for the state of the environment at time  $ts$ . They can be provided by the user's own initiative or when queried by the learner. The labeled data are stored for a certain time  $c_t$ , which can be set based on e.g. data storage possibilities.

The problem discussed in this paper can now be stated as follows: At any given point in time  $t$ , the task is to maximise the accuracy of the estimation of an aspect of the current state of the environment  $\hat{y}_t \in Y$ , using data from  $S_t$ , as well as labels and data, where  $(t - c_t) \leq ts < t$ , as input.

## 2 Discussion

In active learning the learner asks an oracle to label data [3]. Compared to the more common pool-based setting, where the learner starts with all unlabeled data and can ask for labels in any order, our setting is stream-based. Starting with a shortage, or non-existence, of labeled data, the algorithm must learn and adapt over time, as labeled data gradually becomes available. At each point in time the learner must decide whether to query or not, as it is not possible to query for old data points. Still, how much and when to query needs to be balanced, as there is a cost attached to it. Also, the aspect of reliability of the provided labels has to be considered, as human feedback is typically noisy. Different humans do not always agree on definitions or even with themselves over time.

Another complicating factor of the problem is that the entire state set might not be known from the beginning. This means that the learner must query not only to obtain training data, but also to learn the state space. If, for instance, the algorithm is not able to classify a state at a given time with sufficient certainty, it could query the user for the current state.

Generally, the different sensors do not provide data at synchronized points in time, as some are time triggered, with non-standardized time intervals, while others are event triggered. Furthermore, the timestamp attached to each data point is based on the sensor's individual clock. The learner, or a preprocessing step, needs to accommodate for this and align the data time-wise.

Sensor intensive systems can produce large amount of data and while it might be possible to store some data for a specified length of time, chances are all data cannot be stored indefinite. The learner must therefore incorporate the knowledge obtained from the data, so that it is not lost if the data is later discarded.

Transfer learning techniques is another way to handle the shortage of labeled data. For instance, if a model is trained to classify an aspect based on data from sensors in room A, the model could be adapted to do the corresponding task in room B. While transfer learning has been used successfully in many applications, the case where the input feature space for source and target (e.g. sensors in room A and B respectively) differ, is still a relatively new field of research [2, 4].

**Acknowledgements** This work was partially financed by the Knowledge Foundation through the Internet of Things and People research profile.

## References

1. Mihailescu, R.C., Persson, J., Davidsson, P., Eklund, U.: Towards collaborative sensing using dynamic intelligent virtual sensors. In: International Symposium on Intelligent and Distributed Computing. pp. 217–226. Springer (2016)
2. Pan, S.J., Yang, Q., et al.: A survey on transfer learning. IEEE Transactions on knowledge and data engineering **22**(10), 1345–1359 (2010)
3. Settles, B.: Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison (2009)
4. Weiss, K., Khoshgoftaar, T.M., Wang, D.: A survey of transfer learning. Journal of Big Data **3**(1), 9 (2016)

---

# Adaptive Selection of Gaussian Process Model for Active Learning in Expensive Optimization

Jakub Repický<sup>1,2</sup>, Zbyněk Pitra<sup>1,3</sup>, and Martin Holeňa<sup>1</sup>

<sup>1</sup> Institute of Computer Science, Czech Academy of Sciences, [repicky@cs.cas.cz](mailto:repicky@cs.cas.cz)

<sup>2</sup> Faculty of Mathematics and Physics, Charles University

<sup>3</sup> Faculty of Nuclear Sciences and Physical Engineering, Czech Technical University

## 1 Introduction

*Black-box optimization* denotes the optimization of objective functions the values of which are only available through empirical measurements or experiments. Such optimization tasks are most often tackled with evolutionary algorithms and other kinds of metaheuristics methods (e.g., [12]), which need to evaluate the objective function in many points. This is a serious problem in situations when its evaluation is *expensive* with respect to some kind of resources, e.g., the cost of needed experiments.

A standard attempt to circumvent that problem is to evaluate the original objective function only in a small fraction of those points, and to evaluate a *surrogate model* of the original function [5] in the remaining points. Once a model has been trained, the success of the optimization in the remaining iterations depends on a *resource aware selection* of points in which the original function will be evaluated, which is a typical *active learning task*.

The surrogate model used in the reported research is a *Gaussian process* (GP) [11], which treats the values of an unknown function as jointly Gaussian random variables. The advantage of GP compared to other kinds of surrogate models is its capability of quantifying the uncertainty of prediction, by calculating the variance of the posterior distribution of function values.

## 2 Novel Approach to GP-Based Active Learning

So far, active learning of points in which the original objective function will be evaluated during a GP-based surrogate modelling nearly always used a fixed covariance function of the surrogate model and a fixed active learning algorithm (e.g., [9, 15]).

In the reported research, an *adaptive selection of the covariance function* according to the available data is investigated.

To this end, a pool of kernels of 8 various kinds has been established, including non-stationary kernels and composed kernels of depth one (Figure 1).

Adaptive selection of GP covariance functions is known from GP regression literature [3, 6, 8]. Differently to our approach, the number of available kinds of simple kernels is smaller; on the other hand, they can be recurrently composed to

an arbitrary depth through algebraic operations. An additional difference concerns the criteria according to which the models are selected: Whereas in [6], only likelihood is used, and in [3, 8], only the Bayes information criterion (*BIC*) [13] was used, our research includes the investigation of suitability of different criteria for the selection of GP covariance functions in surrogate modelling. Since the maximum likelihood estimate is prone to favour overfitting models, we consider only criteria that account for model complexity. Apart from the *BIC* used in [3, 8], we consider also the Akaike information criterion (*AIC*) [1], the deviance information criterion (*DIC*) [14], and a criterion proposed by Watanabe in [16], called by him widely applicable information criterion (*WAIC*).

## 2.1 Algorithm of Active Learning

The adaptive selection of covariance function is based on GP-based Doubly trained surrogate covariance matrix adaptation evolutionary strategy (DTS-CMA-ES) [2, 10]. A GP model is built in each generation of the evolution strategy. When the model is trained, a fraction of the population maximizing the probability of improvement [7] is selected for evaluation with the expensive objective function. The model is retrained with the newly evaluated points and used to rank the whole population. The fraction of points for evaluation is adapted according to recent performance of the surrogate model, more precisely, to its ranking difference error in the last iteration. A novel contribution of the reported research is a selection of the covariance function according to one of the aforementioned criteria, taking place at the training phase.

## 3 Results and Conclusion

The best performing model selection in our experiments are those based on *AIC* and *BIC*, with no clear difference between them. Results for *DIC* and *WAIC* are not shown because we have not yet succeed to implement these.

On average, optimization results do not show an improvement on the DTS-CMA-ES. Nevertheless, a promising result has been obtained on the “Step ellipsoidal” function, characterized by plateaus lying on a quadratic structure, especially in multi-dimensional variants (Figure 2). The most frequently selected kernel for this function under both *AIC* and *BIC* has been the sum of the squared exponential and the quadratic kernel, which provides an intuitive interpretation of the function.

This result suggests that composite kernels can capture global features of the objective function landscape and provide better predictive performance, but extending the pool of covariances might be needed e. g., by composite kernels of arbitrary depth. This is challenging due the computational cost of searching through an open-ended space of kernel expressions. A possible solution might be found in a co-evolution of the kernel for the surrogate model and the candidate solution to the objective function.

### Acknowledgment

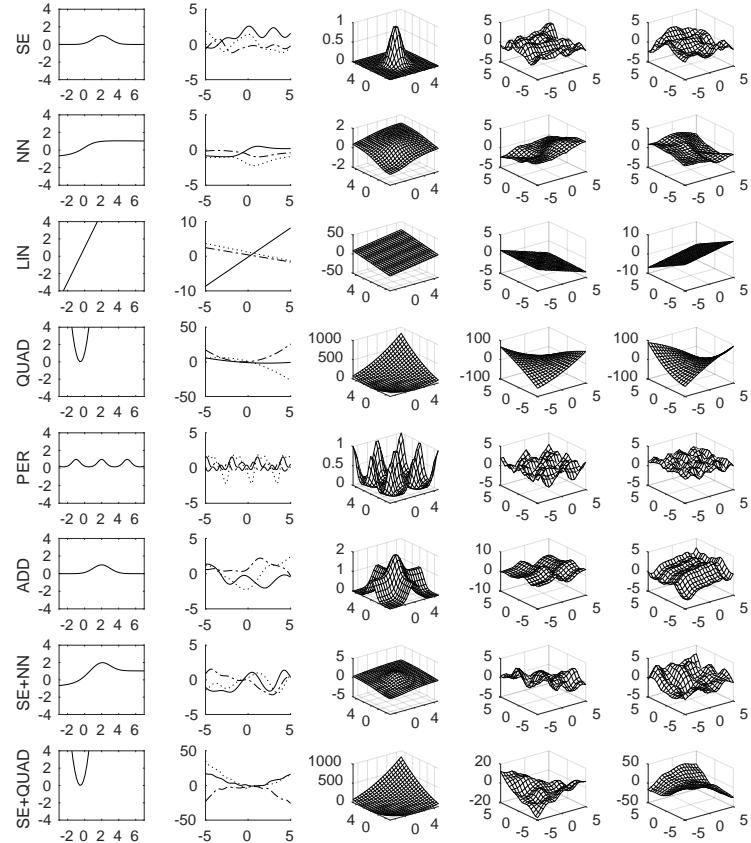
The research reported in this paper has been supported by the SVV project number 260 453 and the Czech Science Foundation (GAČR) grant 17-01251.

## References

1. Akaike, H.: Information Theory and an Extension of the Maximum Likelihood Principle, pp. 199–213. Springer New York (1973)
2. Bajer, L.: Model-based evolutionary optimization methods. Ph.D. thesis, Faculty of Mathematics and Physics, Charles University in Prague, Prague (2018)
3. Duvenaud, D., Lloyd, J., Grosse, R., Tenenbaum, J., Zoubin, G.: Structure discovery in nonparametric regression through compositional kernel search. In: Dasgupta, S., McAllester, D. (eds.) Proceedings of the 30th International Conference on Machine Learning. vol. 28, pp. 1166–1174. PMLR, Atlanta, Georgia, USA (Jun 2013)
4. Hansen, N., Finck, S., Ros, R., Auger, A.: Real-parameter Black-Box Optimization Benchmarking 2012: Experimental setup. Tech. rep., INRIA (2012)
5. Jin, Y.: Surrogate-assisted evolutionary computation: Recent advances and future challenges. *Swarm and Evolutionary Computation* **1**(2), 61–70 (Jun 2011)
6. Kronberger, G., Kommenda, M.: Evolution of covariance functions for gaussian process regression using genetic programming. In: Moreno-Díaz, R., Pichler, F., Quesada-Arencibia, A. (eds.) Computer Aided Systems Theory - EUROCAST 2013. pp. 308–315. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
7. Kushner, H.J.: A new method of locating the maximum point of an arbitrary multipeak curve in the presence of noise. *J. Basic Eng.* **86**(1), 97–106 (1964)
8. Lloyd, J.R., Duvenaud, D., Grosse, R., Tenenbaum, J.B., Ghahramani, Z.: Automatic construction and natural-language description of nonparametric regression models. *CoRR* **abs/1402.4304** (Apr 2014)
9. Lu, J., Li, B., Jin, Y.: An evolution strategy assisted by an ensemble of local Gaussian process models. In: Proceeding of the fifteenth annual conference on Genetic and evolutionary computation conference - GECCO '13. ACM (2013)
10. Pitra, Z., Bajer, L., Holeňa, M.: Doubly trained evolution control for the surrogate CMA-ES. In: Parallel Problem Solving from Nature – PPSN XIV, pp. 59–68. Springer International Publishing (2016)
11. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. Adapative computation and machine learning series, MIT Press (2006)
12. Schaefer, R.: Foundations of Global Genetic Optimization. Springer Publishing Company, Incorporated, 1st edn. (2007)
13. Schwarz, G.: Estimating the dimension of a model. *The Annals of Statistics* **6**(2), 461–464 (1978)
14. Spiegelhalter, D., Best, N., Carlin, B., Van Der Linde, A.: Bayesian measures of model complexity and fit. *Journal of the Royal Statistical Society. Series B: Statistical Methodology* **64**(4), 583–616 (12 2002)
15. Volz, V., Rudolph, G., Naujoks, B.: Investigating Uncertainty Propagation in Surrogate-assisted Evolutionary Algorithms. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 881–888. GECCO '17, ACM, New York (2017)
16. Watanabe, S.: Algebraic Geometry and Statistical Learning Theory. Cambridge Monographs on Applied and Computational Mathematics, Cambridge University Press (2009)

## A Appendix

### A.1 Covariance Functions



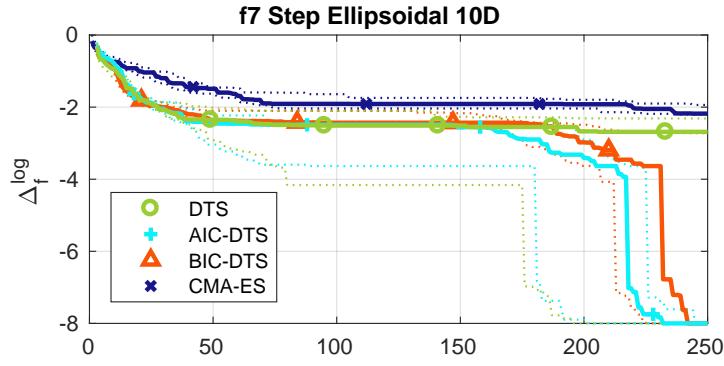
**Fig. 1.** Rows: Covariance functions. SE: Squared exponential. NN: neural network. LIN: linear. QUAD: quadratic. PER: periodic. ADD: additive. SE+NN: sum of squared exponential and neural network. SE+QUAD: sum of squared exponential and quadratic. Columns 1–2: The covariance function on  $\mathbb{R}$  centered at point 2 (Col. 1) and three independent samples from the GP (Col. 2). Columns 3–5: The covariance function on  $\mathbb{R}^2$  centered at  $[2 \ 2]^T$  (Col. 3) and two independent samples from the GP (Col. 4 and 5).

### A.2 Experimental Setup

The ongoing experiments are performed within the framework Comparing continuous optimisers [4], in particular on the 24 benchmark functions forming its

noiseless testbed. Each function is defined everywhere on  $\mathbb{R}^D$  and has its global optimum in  $[-5, 5]^D$  for all dimensionalities  $D \geq 2$ . For every function and two dimensionalities,  $2D$  and  $10D$ , 15 independent trials of each algorithm are conducted on multiple function instances, defined by transformations (translations, rotations and shifts) of both the search space and  $f$ -values. A trial terminates when the optimum is reached within a small tolerance or when a given budget of function evaluations,  $250D$  in our case, is exhausted.

### A.3 Experimental Results



**Fig. 2.** Medians (solid) and 1<sup>st</sup>/3<sup>rd</sup> quartiles (dotted) of the distances to the optima against the number of function evaluations in 10D for all satisfactorily implemented algorithms. CMA-ES: Covariance matrix adaptation evolution strategy. DTS-CMA-ES: Doubly trained surrogate-CMA-ES. AIC-DTS, BIC-DTS: DTS-CMA-ES with the adaptive covariance selection according to AIC and BIC, respectively. The medians and quartiles were calculated from 15 independent runs on different function instances. Distances to optima are shown in the  $\log_{10}$  scale.

---

# Towards Interactive Feature Selection with Human-in-the-loop

Maialen Larrañaga, Dimitra Gkorou, Thiago Guzella, Alexander Ypma,  
Faegheh Hasibi, Robert Jan van Wijk

ASML, De Run 6501, Veldhoven 5504DR, the Netherlands

## 1 Introduction

Feature Selection (FS) has been applied to numerous domains, and shown to be effective in increasing the performance of machine learning algorithms. In the semiconductor industry, FS is part of various prediction tasks that aim at avoiding production stops and yield loss. For example, it can be used for: (i) *diagnostics*, wherein relevant features constitute potential root causes, with their identification being the initial step in a detailed investigation of process defects [3]; (ii) *control*, as the values of a small set of relevant features can be used to group objects and apply actions per group [1]; (iii) *improving prediction performance and interpretability*, by enforcing sparsity [4, 7]. Nevertheless, when analyzing manufacturing datasets, one faces two particular challenges, as in other real-world datasets:

- **High-dimensionality:** typically, the number of features  $p$  to be evaluated in FS can reach hundreds of thousands and it is much larger than the number of instances  $n$ .
- **Collinearity:** some features may be strongly correlated with one another.

These characteristics challenge the robustness of FS against spurious correlations. To overcome these challenges, we propose an interactive FS scheme in which a user provides expert feedback, by assessing the relevance of the features with respect to a performance metric and thereby separating relevant features from spurious correlations. An initial approach has been implemented as a web application in ASML, the leading manufacturer of lithography machines and major player in the semiconductor industry. Next, we give a detailed explanation of the proposed interactive scheme, while the supplementary material provides further details on the implementation.

## 2 Interactive Feature Selection

For integrating expert feedback, we use the knowledge elicitation framework proposed by Daei et al. [2]. It is based on a bayesian regression model with spike and slab (s&s) sparsity-enforcing priors [2, 5]. Let us denote by  $\mathbf{y} \in \mathbb{R}^n$  the target of interest, and let  $\mathbf{X} \in \mathbb{R}^{n \times p}$  be the feature set. Assume  $\mathbf{w} \in \mathbb{R}^p$  to be the regression coefficients. The goal is to define the posterior distribution of  $\mathbf{w}$ , given that  $\mathbf{y} \sim \mathcal{N}(\mathbf{X}\mathbf{w}, \sigma^2 I)$  and  $w_j \sim z_j \mathcal{N}(0, \psi^2) + (1 - z_j)\delta_0$  (i.e., s&s prior), where  $\delta_0$  is a Dirac delta, and  $z_j$  encodes the relevance of the features ( $z_j = 1$  if feature  $j$  is expected to contribute to the regression and  $z_j = 0$  otherwise). For ease of notation, we consider  $\sigma^2$  and  $\psi^2$  to be constants. The computation of the posterior distribution of  $w_j$  for all  $j = 1, \dots, p$  depends on the s&s prior but also on the feedback provided by a domain expert (*relevant*, *irrelevant*, *do not know*). These

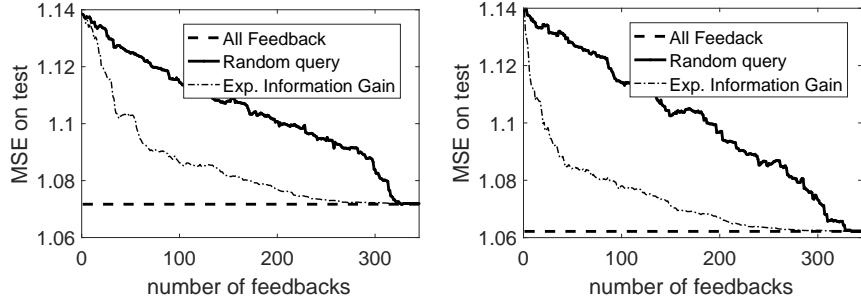


Fig. 1: Evaluation of FS approach for different query strategies and user feedback. Left: Evaluation with imperfect user feedback. Right: Evaluation with real expert feedback.

postiors are sequentially adapted every time an expert is queried about the relevance of a feature. In order not to overwhelm the expert, we use smart query strategies that reduce the number of required user feedback. We implement the Expected Information Gain (EIG) strategy proposed in Daee et al. [2].

We evaluated the implementation of our FS scheme with an ASML expert. We used a dataset with 344 features and 100 instances from the logs of ASML machines. The results are shown in Fig. 1, for a simulated expert (Fig. 1, left) and for a knowledgeable domain expert (Fig. 1, right), with the former being less trustworthy than the latter. In these results, “All feedback” refers to the performance of the bayesian regression with s&s priors after all expert feedback is received; we note that the best achieved MSE in Fig. 1 left is higher than the best MSE in Fig. 1 right, but in both cases the EIG strategy reaches the best MSE after  $\sim 250$  feedbacks. Even with a simulated expert (Fig. 1, left) we see an improvement on the predictive performance with respect to not having an expert in the loop (i.e., when the number of feedback equals 0). This is crucial for the robustness of our application.

### 3 Open questions and future work

At present, we still face several challenges in the implementation of the proposed scheme. First, our datasets have typically thousands of features, such that asking feedback from a domain expert for all these features is unfeasible. Second, domain experts are often not sure of the feedback for some particular features, and their input might also be biased. To address these challenges, we are adopting the following steps:

- group features that relate to the same effects/machines in a single category and ask the expert to provide feedback on the entire category at once.
- visualize the data in an informative way to help the expert make a decision on the relevance of a particular feature category and avoid biases.

How to group features into categories and how to visualize the data in the *most informative* way remain an open question. In particular, the data visualization is a challenging task since (1) dimensionality reduction methods show artifacts, (2) assessing the quality of a visualization is not straightforward and (3) often only well-known effects and structures emerge in the visualization. One needs to get rid of the latter to discover the underlying patterns.

## References

1. H. E. Cekli, J. Nije, A. Ypma, V. Bastani, D. Sonntag, H. Niesing, L. Zhang, Z. Ullah, V. Subramony, R. Somasundaram, W. Susanto, M. Matsunobu, J. Johnson, C. Tabery, C. Lin, and Y. Zou. A novel patterning control strategy based on real-time fingerprint recognition and adaptive wafer level scanner optimization. volume 10585, 2018.
2. Pedram Daee, Tomi Peltola, Marta Soare, and Samuel Kaski. Knowledge elicitation via sequential probabilistic inference for high-dimensional prediction. *Machine Learning*, 106:1599–1620, 2017.
3. M. Giollo, A. Lam, D. Gkorou, X. Lan Liu, and R. van Haren. Machine learning for fab automated diagnostics. *Proc. SPIE*, 10446, 2017.
4. F. Hasibi, L. van Dijk, M. Larrañaga, A. Pastol, A. Lam, and R. van Haren. Towards fab cycle time reduction by machine learning based overlay metrology. *Proc. SPIE*, 2018.
5. J. M. Hernández-Lobato, D. Hernández-Lobato, and A. Suárez. Expectation propagation in linear regression models with spike-and-slab priors. *Machine Learning*, 99(3):437–487, 2015.
6. L. Kirsch, N. Riekenbrauck, D. Thevessen, M. Pappik, A. Stebner, J. Kunze, A. Meissner, A. K. Shekar, and E. Müller. Framework for exploring and understanding multivariate correlations. In *Machine Learning and Knowledge Discovery in Databases*, pages 404–408, 2017.
7. A. Lam, A. Ypma, M. Gatefait, D. Deckers, A. Koopman, R. van Haren, and J. Beltman. Pattern recognition and data mining techniques to identify factors in wafer processing and control determining overlay error. *Proc. SPIE*, 9424, 2015.

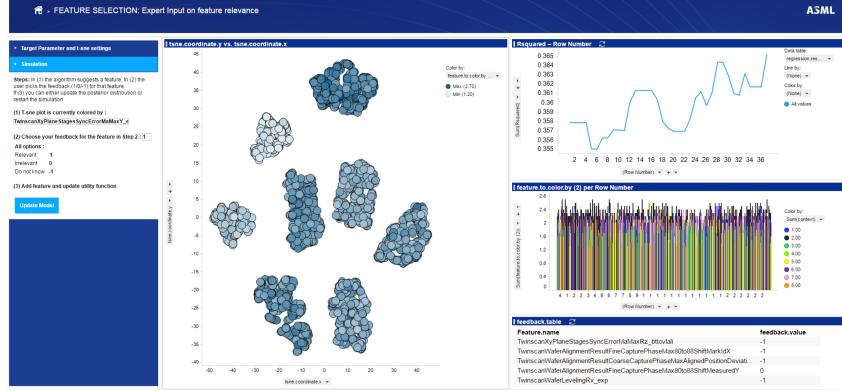


Fig. 2: Snapshot of the visualization of the proposed FS framework

#### 4 Supplementary Material: web application

In this work, we sought to use smart query strategies to reduce the number of required user feedback. While some state-of-the-art tools allow for user feedback, none of them are directly applicable to our problem setting. RapidMiner, Weka, and FEXUM [6] focus on supervised and unsupervised problems allowing only offline user feedback. KNIME has an active learning component, but it cannot be directly applied to FS. Therefore, we have implemented a web application in ASML for the interactive Feature Selection scheme.

The expected users of our framework are domain experts without Machine Learning knowledge but with scientific background e.g., domain experts, field engineers, etc. The experts can use our tool to easily find the relevant features to a selected target variable. The User Interface of the proposed FS scheme is presented in Fig. 2. The expert initializes a workflow by selecting one or more performance metrics to monitor (targets  $y$ ) on the left panel in Fig. 2. The application, with non-linear embeddings (Fig. 2 middle plot), helps the user understand complex relations and interconnections between multiple features and the target. Particularly, users can select multiple features and observe interactively, via a t-SNE embedding, their 2D visualization colored by the values of the target.

In each iteration a user is provided with a feature (left most panel) for which he/she is required to give feedback (*relevant* feature, *irrelevant* feature or *do not know*). As mentioned in the abstract, this feature is selected sequentially via knowledge elicitation with the EIG query strategy. The user interacts with the application to provide feedback on a certain number of features. To help the decision-making process, on Fig. 2 (right), three plots are shown: (i) the  $R^2$  value of the linear regression model after each expert feedback. This provides information on the predictive power of the model in each iteration; (ii) the values of the feature that the user is evaluating colored per cluster. This gives information on how the target value relates to the underlying structure of the data; (iii) a list with all the feedback that he/she has already provided. Once the decision has been made, the feedback is given on the left panel, the posterior distribution of the regression is updated and the algorithm suggests a new feature.

In this way, actively querying the expert for feedback results in better user experience as, the users are not expected to examine all the features to find those whose relevance they should assess.

---

# Transfer of Knowledge for Surrogate Model Selection in Cost-Aware Optimization

Zbyněk Pitra<sup>1,2</sup>, Jakub Repický<sup>1,3</sup>, and Martin Holeňa<sup>1</sup>

<sup>1</sup> Institute of Computer Science, Academy of Sciences of the Czech Republic  
`{pitra, repicky, holena}@cs.cas.cz`

<sup>2</sup> Faculty of Nuclear Sciences and Physical Engineering, CTU in Prague

<sup>3</sup> Faculty of Mathematics and Physics, Charles University in Prague

## 1 Introduction

Surrogate model selection is an active-learning approach to cost-aware continuous black-box optimization in domains where the evaluation of the black-box objective function is expensive, e.g., obtained experimentally or resulting from comprehensive simulations. Active reusing of knowledge represented by landscape properties of the objective function across different tasks can provide additional information for more reliable decisions in terms of a suitable surrogate model and a suitable setting of its hyperparameters. However, research into using metalearning [13] and especially *Exploratory Landscape Analysis* (ELA) [14] in this context is only starting [20]. Our goal is to develop a learning system capable to recommend a surrogate model on the basis of the knowledge obtained in previous black-box optimization tasks.

In this paper, we provide a first step necessary to construct a learning system applying knowledge from previous tasks to a new one: a study of the applicability of ELA to two important kinds of surrogate models – Gaussian processes (GP) [18] and ensembles of regression trees (random forests, RF) [3,4,6]. Results using the noiseless benchmarks of the Comparing-Continuous-Optimisers (COCO) platform [9] in the expensive scenario, where at most 50D evaluations are available, are analysed for statistical dependences between model performance and a broad variety of landscape features.

## 2 Exploratory Analysis of Fitness Landscapes

In order to achieve our goal, the relationships between data properties and surrogate model performances has to be analysed in detail first. Second, the investigated relationships will be used to design a system capable to transfer knowledge about relationships from processed tasks to new ones.

The surrogate model selection problem is analogous to the algorithm selection problem (ASP) formulated in [19] and it aims at selecting the most suitable surrogate model for a specific optimization task. Considering ASP, ELA [14] aims at characterizing the landscape of an investigated function and deriving rules how those characteristics influence the performance of the optimization algorithm.

We analyze relationships between the mean-squared error (MSE) of 29 different settings of GP or RF described in Appendix A.1 and 79 out of 91 ELA features (see also Appendix A.1) which didn't yield constant over 24 noiseless benchmark functions from the COCO framework [9] in dimensions  $D = \{2, 5, 10\}$  and their 15 instances<sup>4</sup> for any of the tested GP or RF settings. The datasets consisting of  $50D$  points for each instance per function were generated by a random improved Latin Hypercube design [1] covering the input space  $[-5, 5]^D$ . The overall predictive performance of the surrogate models was tested through 5-fold cross-validation on the generated datasets.

As a first step, we performed a simple correlation analysis using the Spearman correlation coefficient between the MSE of the considered models and the investigated ELA features. However, no single ELA feature was found to be discriminative for surrogate model performance, although a few features were positively (or negatively) correlated with all considered models, which indicates the landscape to be difficult (or easy) for fitting any of them.

As a second step, a classification tree representing a multivariate statistical analysis was built using the obtained results. The resulting tree is depicted in Figure 1. 79 ELA features were classified into 29 classes according to which of the 29 considered settings of GP and RF achieved the lowest MSE for the respective combination of dimension and function among all evaluated settings. Features describing the global structure of the objective function landscape were detected as most distinctive (f12, f15, f61, f62, f64, f70, and f71). Global structure of the landscape can possibly influence the performance of a particular model. Very interesting is the discovered importance of basic features such as dimension (f1) or extreme values of the objective function (f3). In addition, skewness (f35) and the kurtosis (f36) also had influence on surrogate model selection. The last mentioned observations may suggest that even a set of simple features can provide valuable information about the model suitability.

### 3 Discussion

The results suggest that clear relationships between the performance of the 29 compared settings of GP and RF models and the considered features are not easy to derive. Features describing global properties of the landscape are very useful in case of selection of the surrogate model and its settings. On the other hand, simple features can also provide important knowledge useful for future decisions.

The intended direction for our future research is to apply the obtained knowledge to select a suitable surrogate model for previously unseen data in designing a metalearning system. Another important research direction is to investigate the impact of the sampling strategy in the input space to the resulting landscape features and their relationship with the performance of the considered models and their various settings.

---

<sup>4</sup> Function instances are defined by transformations (translations, rotations, and shifts) of both the search space and function values.

**Acknowledgements** The reported research was supported by the Czech Science Foundation grant No. 17-01251, by the Grant Agency of the Czech Technical University in Prague with its grant No. SGS17/193/OHK4/3T/14, and by Specific College Research project number 260 453. Computational resources were provided by the CESNET LM2015042 under the programme "Projects of Large Research, Development, and Innovations Infrastructures".

## References

1. Beachkofski, B., Grandhi, R.: Improved distributed hypercube sampling. In: Proceedings of the 43rd AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference. p. 1274. American Institute of Aeronautics and Astronautics (2002)
2. Beirlant, J., Dudewicz, E.J., Györfi, L., Van der Meulen, E.C.: Nonparametric entropy estimation : an overview. International Journal of Mathematical and Statistical Sciences **6**(1), 17–39 (1997)
3. Breiman, L.: Classification and regression trees. Chapman & Hall/CRC (1984)
4. Breiman, L.: Bagging predictors. Machine learning **24**(2), 123–140 (1996)
5. Chaudhuri, P., Huang, M.C., Loh, W.Y., Yao, R.: Piecewise-polynomial regression trees. Statistica Sinica **4**(1), 143–167 (1994)
6. Chen, T., Guestrin, C.: XGBoost: A scalable tree boosting system. pp. 785–794. KDD '16, ACM (2016)
7. Dobra, A., Gehrke, J.: SECRET: A scalable linear regression tree algorithm. pp. 481–487. KDD '02, ACM (2002)
8. Duvenaud, D.K., Nickisch, H., Rasmussen, C.E.: Additive gaussian processes. In: Advances in Neural Information Processing Systems 24, pp. 226–234. Curran Associates, Inc. (2011)
9. Hansen, N., Finck, S., Ros, R., Auger, A.: Real-parameter black-box optimization benchmarking 2009: Noiseless functions definitions. Tech. Rep. RR-6829, INRIA (2009), updated February 2010
10. Hinton, G.E., Revow, M.: Using pairs of data-points to define splits for decision trees. In: Advances in Neural Information Processing Systems. vol. 8, pp. 507–513. MIT Press (1996)
11. Kerschke, P.: Comprehensive feature-based landscape analysis of continuous and constrained optimization problems using the R-package flacco. ArXiv e-prints (2017)
12. Kerschke, P., Dagefoerde, J.: flacco: Feature-Based Landscape Analysis of Continuous and Constraint Optimization Problems (2017), <https://cran.r-project.org/package=flacco>, R-package v. 1.7
13. Lemke, C., Budka, M., Gabrys, B.: Metalearning: a survey of trends and technologies. Artificial Intelligence Review **44**(1), 117–130 (Jun 2015)
14. Mersmann, O., Bischl, B., Trautmann, H., Preuss, M., Weihs, C., Rudolph, G.: Exploratory landscape analysis. pp. 829–836. GECCO '11, ACM (2011)
15. Murthy, S.K., Kasif, S., Salzberg, S.: A system for induction of oblique decision trees. J. Artif. Int. Res. **2**(1), 1–32 (1994)
16. Neal, R.M.: Bayesian Learning for Neural Networks. Springer-Verlag New York, Inc., Secaucus, NJ, USA (1996)
17. Pitra, Z., Repický, J., Holeňa, M.: Boosted regression forest for the doubly trained surrogate covariance matrix adaptation evolution strategy. ITAT 2018, CreateSpace Independent Publishing Platform, North Charleston, USA (2018)

18. Rasmussen, C.E., Williams, C.K.I.: Gaussian Processes for Machine Learning. Adaptive computation and machine learning series, MIT Press (2006)
19. Rice, J.R.: The algorithm selection problem. Advances in Computers, vol. 15, pp. 65 – 118. Elsevier (1976)
20. Yu, H., Tan, Y., Sun, C., Zeng, J., Jin, Y.: An adaptive model selection strategy for surrogate-assisted particle swarm optimization algorithm. pp. 1–8. SSCI '16 (2016)

## A Appendix

### A.1 Experimental setup

The GP regression model in `gpml` implementation<sup>5</sup> was employed using 9 different covariance functions, listed in Table 1, and constant mean  $\mu(\mathbf{x}) = \text{mean}(\mathbf{y})$ , where  $\mathbf{y}$  are the outputs of the training set. The hyperparameters were optimized with MATLAB's `fmincon` using 5 optimization trials, except for the additive covariance function  $k_{\text{ADD}}$ , which was optimized with only 3 trials due to its relatively high complexity. The rest of initial values for hyperparameters, together with their bounds are reported Table 2. The initial values for repeated optimization trials were sampled.

The RF models were tested using the full-factorial design on the ensemble method, splitting method, and error gain function. In addition, the number of trees  $n_{\text{tree}}$ , the number of points  $N_t$ , and the number of dimensions used for training the individual tree  $n_D$  were sampled from the values in Table 3. Thus, the RF experimental part sampled RF models from 1600 different settings. MSE ( $\text{err}_{\text{MSE}}$ ), variance of predicted  $y$ -values ( $\text{err}_{\text{var}}$ ), and nearest-neighbor entropy estimator [2] ( $\text{err}_{\text{NN}}$ ) were employed as error gain functions ( $\text{err}$ ). In bagged RF, cross-validation pruning [3] was utilized to optimize the tree structure. In addition, the following five regression models were used in leaves: constant, linear, linear with interactions, quadratic without interactions, and full quadratic. The model providing the best fit according to the MSE loss function was always selected for the relevant leaf and appropriate data. In boosted RF, the maximum tree depth was set to 8, in accordance with [6].

Considering decision tree settings regardless the ensemble method, the five splitting methods from the following algorithms were employed: CART [3], SCRET [7], OC1 [15], SUPPORT [5], and a method from [10] (PAIR). The remaining decision tree parameters have been taken identical to settings from [17].

The 91 calculated landscape features were from the following 11 ELA feature sets [11,12]: *y-Distribution*, *Levelset*, *Meta-Model*, *CM-Angle*, *CM-Gradient Homogeneity*, *CM-Concavity*, *NBC*, *Dispersion*, *Information Content*, *Basic*, and *PCA*. Feature sets requiring additional evaluations of the objective function (*Concavity*, *Local Search*, and *Curvature*) and cell-mapping feature sets with high computational or memory requirements in higher dimension (*GCM*, *Barrier Trees*, and *Linear Model*) were omitted. All landscape features were calculated using default settings from [12].

---

<sup>5</sup> <http://www.gaussianprocess.org/gpml/code/matlab/doc/>

**Table 1.** Experimental settings of GP covariances:  $d$  – metric  $d(\mathbf{x}_p, \mathbf{x}_q)$ ,  $P$  – isotropic distance measure  $P = l^{-2} I_D$ ,  $\tilde{\mathbf{x}}_p, \tilde{\mathbf{x}}_q$  – inputs augmented by a bias unit,  $k^{(i)}(\mathbf{x}_p^{(i)}, \mathbf{x}_q^{(i)})$  – one-dimensional  $k_{\text{SE}}$ ,  $R \subseteq \{1, \dots, D\}$  – set of selected degrees of interactions.  $k_{\text{SE}}$  and  $k_{\text{RQ}}$  were used in both isotropic and automatic relevance determination (ARD) versions ( $k_{\text{SE}}^{\text{ARD}}$ ,  $k_{\text{RQ}}^{\text{ARD}}$ ).

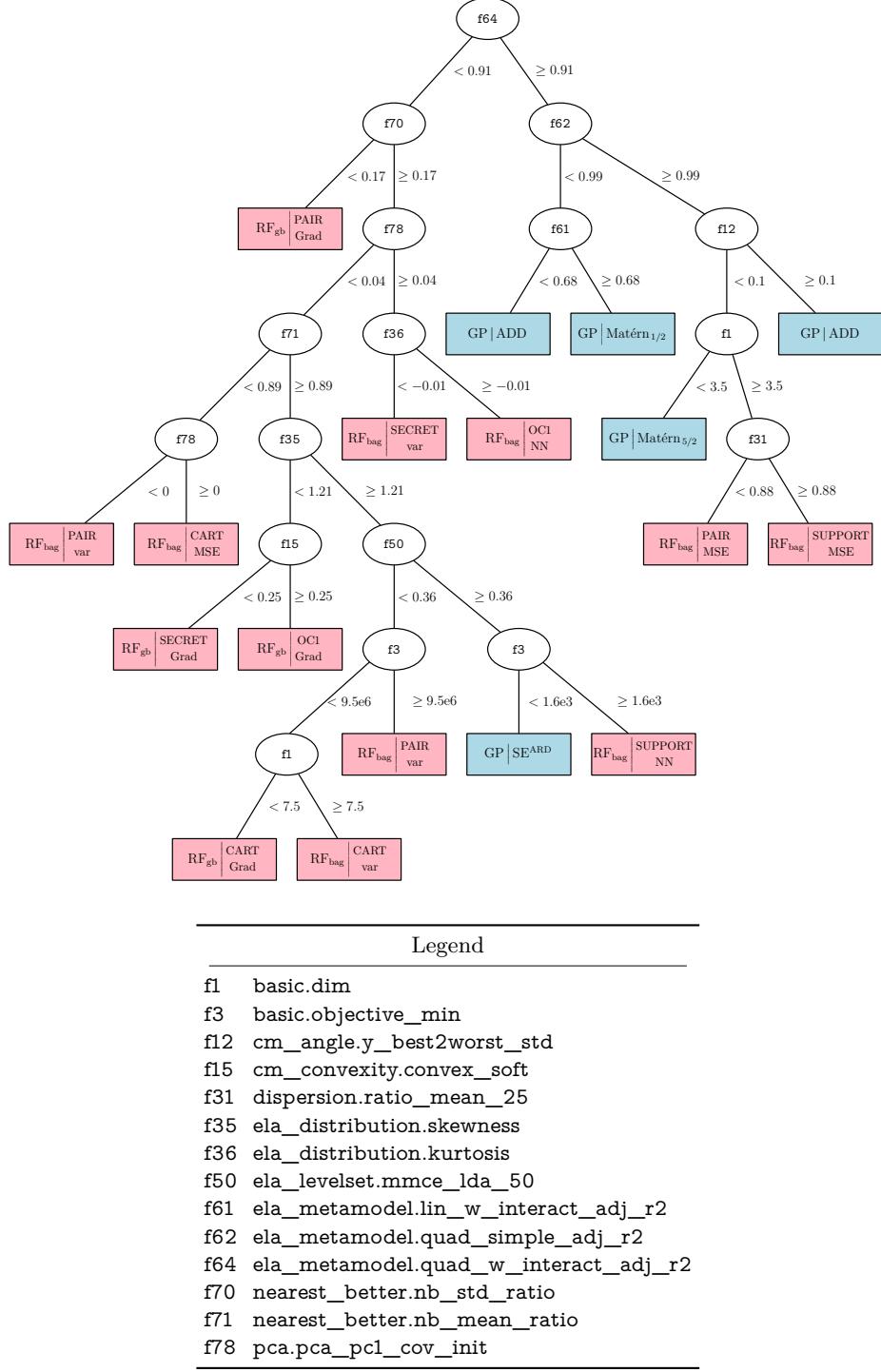
| name                | kernel  |
|---------------------|---|
| Matérn family       | $k_{\text{SE}}(d; \sigma_f, l) = \sigma_f^2 \exp\left(-\frac{d^2}{2l^2}\right)$   |
|                     | $k_{\text{Matérn}}^{\nu=\frac{1}{2}}(d; \sigma_f, l) = \sigma_f^2 \exp\left(-\frac{d}{l}\right)$  |
|                     | $k_{\text{Matérn}}^{\nu=\frac{3}{2}}(d; \sigma_f, l) = \sigma_f^2 \left(1 + \frac{\sqrt{3}d}{l}\right) \exp\left(-\frac{\sqrt{3}d}{l}\right)$   |
|                     | $k_{\text{Matérn}}^{\nu=\frac{5}{2}}(d; \sigma_f, l) = \sigma_f^2 \left(1 + \frac{\sqrt{5}d}{l} + \frac{5d^2}{3l^2}\right) \exp\left(-\frac{\sqrt{5}d}{l}\right)$   |
| rational quadratic  | $k_{\text{RQ}}(d; \sigma_f, l) = \sigma_f^2 \left(1 + \frac{d^2}{2l^2\alpha}\right)^{-\alpha}$  |
| neural network [16] | $k_{\text{NN}}(\mathbf{x}_p, \mathbf{x}_q) = \sigma_f^2 \arcsin\left(\frac{2\tilde{\mathbf{x}}_p^T P \tilde{\mathbf{x}}_q}{\sqrt{(1+2\tilde{\mathbf{x}}_p^T P \tilde{\mathbf{x}}_p)(1+2\tilde{\mathbf{x}}_q^T P \tilde{\mathbf{x}}_q)}}\right)$ |
| additive [8]        | $k_{\text{ADD}}(\mathbf{x}_p, \mathbf{x}_q, R) = \sum_{r \in R} \sigma_f^{(r)} \sum_{1 \leq i_1 < i_2 < \dots < i_r \leq D} \left( \prod_{d=1}^r k^{(i_d)}(\mathbf{x}_p^{(i_d)}, \mathbf{x}_q^{(i_d)}) \right)$                                 |

**Table 2.** Experimental settings of GP.  $\sigma_n$  and  $l$  apply to all covariances.  $\sigma_f$  applies to all except the  $k_{\text{ADD}}$ , in which  $\sigma_f^{(r)}$  scales each degree  $r \in R$ ,  $R = \{1, 2, 3, 5, 7, 10\} \cap \{1, \dots, D\}$  of interaction separately.  $\sigma_f^{(r)}$  and its upper bound were initialized proportionally to  $\binom{D}{r}$ .

| GP hyperparam | initial                                   | value       | constraints |
|---------------|---|-------------|-------------|
| $\sigma_n$    | 1e-2                                      | [1e-3, 1e1] |             |
| $l$           | std(X)                                    | [1e-2, 1e2] |             |
| $\sigma_f$    | $\frac{\text{std}(\mathbf{y})}{\sqrt{2}}$ | [1e-2, 1e6] |             |

**Table 3.** Experimental settings of RF:  $n_{\text{tree}}$  – number of trees in RF,  $N_t$ ,  $n_D$  – number of tree points and dimensions,  $N$  – number of RF points,  $D$  – input space dimension,  $\text{err}_{\text{Grad}}$  – gradient error gain. Split methods and error gain functions err are tested using full-factorial design,  $n_{\text{tree}}$ ,  $N_t$ , and  $n_D$  are sampled.

| RF param          | bagging  | boosting            |
|-------------------|--|---------------------|
| err               | {err <sub>MSE</sub> , err <sub>var</sub> , err <sub>NN</sub> } | err <sub>Grad</sub> |
| split             | {CART, SECRET, OC1, SUPPORT, PAIR}                             |                     |
| $n_{\text{tree}}$ | {64, 128, 256, 512, 1024}                                      |                     |
| $N_t$             | [{0.25, 0.5, 0.75, 1} · $N$ ]                                  |                     |
| $n_D$             | [{0.25, 0.5, 0.75, 1} · $D$ ]                                  |                     |



**Figure 1.** Classification tree demonstrating the influence of ELA features on model suitability. Light blue: Gaussian processes. Light pink: Random forests. The RF models use the notation  $\text{RF}_{\text{ensemble method}}^{\text{method}}$ , where bag = bagging, gb = gradient boosting, and Grad = gradient error gain. The GP models use the notation  $\text{GP} \mid \text{covariance function}^{\text{ARD}}$ , where  $\text{ARD}$  = automatic relevance determination (scaling in each dimension separately).

---

# Embodiment Adaptation from Interactive Trajectory Preferences

Michael Walton, Ben Migliori, John Reeder

Space and Naval Warfare Systems Center Pacific

<http://www.public.navy.mil/spawar/Pacific>

{michael.walton, benjamin.migliori, john.d.reeder}@navy.mil

**Keywords:** Imitation Learning · Preference Learning · Reinforcement Learning

## 1 Introduction

Imitation learning provides an attractive approach to communicate complex goals to autonomous systems in domains where explicit reward functions are unavailable, tedious to specify or rely on substantial or high-cost expert knowledge. Standard Imitation Learning implicitly assumes that the embodiment of the learning agent and the teacher are either the same or intuitively compatible from the perspective of the demonstrator. In this work, we consider control tasks which violate these assumptions and propose a framework for estimating *embodiment adaptors* using human feedback expressed through pairwise preferences over control trajectories.

## 2 Background

Recent advances in reinforcement learning (RL) have largely been driven by scaling algorithms well understood in simple task domains to complex, high-dimensional problems using deep neural networks for value function approximation [6] and policy learning [5]. In the standard formulation of a reinforcement learning problem, often posed as a Markov Decision Process (MDP), one assumes access to a reward function  $R : S \times A \rightarrow \mathbb{R}$  which associates a scalar reward with agent actions  $a \in A$  taken in states  $s \in S$ . The agents' objective, therefore, is to maximize its cumulative reward. In many well posed control tasks, this objective may be straightforward to specify: the score of a game, the goal configuration in robotic manipulation tasks, forward velocity for walking or crawling.

Complementary to RL, Imitation Learning provides an approach for learning a control policy without an explicit reward function. This approach is desirable in problems domains where a concise goal statement may be challenging to express [1], [2]. Prior work has also explored imitation learning to improve the sample efficiency of reinforcement learning [3], [4]. Conventional approaches to imitation learning, however, fundamentally rely on the availability of demonstrations of expert control in the form of observation, action tuples. Demonstration data may

be acquired through teleoperation<sup>1</sup> or kinesthetic teaching<sup>2</sup>. In the former case, the imitator and the demonstrator are assumed to have the same embodiment, eg. their state and action spaces are assumed to be consistent. In the latter, the demonstrator must inhabit the same physical space as the embodied agent and must be able to efficiently pose and manipulate its effectors.

Many complex control tasks may exhibit incompatibilities between the embodiments of the demonstrator and the imitating agent. Consider for instance a robotic arm we may wish to train to perform household tasks such as preparing food; pose estimates of a human demonstrator's arm will yield sequences of actions with different degrees of freedom and dynamics than the imitating arm.

### 3 Methods

Our proposed approach takes two stages: In the first stage the human demonstrator provides undirected feedback to the agent to optimize a policy  $\pi_\alpha : A_H \rightarrow A_\ell$  which translates between the demonstrators action space  $A_H$  and agent's action space  $A_\ell$ . This is achieved through trajectory preference learning [1], however in our formulation preferences are assigned to the trajectory that best matched the demonstrators' desired action. Formally, we state that a trajectory  $\tau^1$  is preferred, denoted  $\succ$  to  $\tau^2$  following a reward function  $r$  known only to the demonstrator if:

$$\tau^1 \succ \tau^2 \equiv \sum_t r(a_t^1, \pi_\alpha(a_t^1)) > \sum_t r(a_t^2, \pi_\alpha(a_t^2)) \quad (1)$$

After each interaction, a pairwise preference is assigned between the two trajectories and an reward function approximation  $\hat{r}$  is estimated using the method specified in [1]. The embodiment adaptation policy is then subsequently trained to maximized  $\hat{r}$  using standard reinforcement learning. After learning an embodiment adaptation policy, the second phase uses this mechanism to learn a behavior policy  $\pi_\beta$  from translated demonstrations using (for instance) behavioral cloning. In this simple formulation, the optimal policy given expert demonstrations  $D$  is the policy that minimizes the divergence between  $\pi_\beta$  and the expert actions translated by  $\pi_\alpha$ ; assuming continuous actions, we may define this objective in terms of the quadratic loss:

$$\pi_\beta^* = \arg \min_{\pi_\beta \in \Pi_\beta} \mathbb{E}_{(s,a) \sim D} [(\pi_\alpha(a) - \pi_\beta(s))^2] \quad (2)$$

We propose two proof of concept embodiment translation tasks to demonstrate the utility of our method: a classic gridworld with discrete state and action

---

<sup>1</sup> The demonstrator directly controls the agent which records action selections for imitation

<sup>2</sup> The demonstrator physically manipulates an embodied agent by applying force to its effectors; demonstration in these scenarios may be, for instance, resultant torques on the joints of a robotic arm

spaces and the continuous control problem *lunar lander*. In the lunar lander task, for instance, the human demonstrator must select thrust directions using the up, left and right keys; it is observed in [7] that humans tend to fail on this task. Distinct from previous work, we hypothesize that this is an unintuitive interface for a human operator to demonstrate correct behavior. A more natural interface, perhaps, may be a joystick-like interface. We apply our method to learn an embodiment adaptor policy  $\pi_\alpha$  which translates continuous forces applied to a joystick to sequences of discrete thruster pulses which are compatible with the imitator's embodiment.

## References

1. Christiano, P., Leike, J., Brown, T.B., Martic, M., Legg, S., Amodei, D.: Deep reinforcement learning from human preferences (2017)
2. Hadfield-Menell, D., Dragan, A., Abbeel, P., Russell, S.: The off-switch game (2016)
3. Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Horgan, D., Quan, J., Sendonaris, A., Dulac-Arnold, G., Osband, I., Agapiou, J., Leibo, J.Z., Gruslys, A.: Deep q-learning from demonstrations (2017)
4. Hester, T., Vecerik, M., Pietquin, O., Lanctot, M., Schaul, T., Piot, B., Sendonaris, A., Dulac-Arnold, G., Osband, I., Agapiou, J., Z. Leibo, J., Gruslys, A.: Learning from demonstrations for real world reinforcement learning (04 2017)
5. Lillicrap, T.P., Hunt, J.J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., Wierstra, D.: Continuous control with deep reinforcement learning (2015)
6. Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., Riedmiller, M.: Playing atari with deep reinforcement learning (2013)
7. Reddy, S., Dragan, A.D., Levine, S.: Shared autonomy via deep reinforcement learning (2018)