An Adaptive and Dynamic Simulation Framework for Incremental, Collaborative Classifier Fusion

Gernot Bahle[1], Andreas Poxrucker[1], George Kampis[1,2,3], and Paul Lukowicz[1]

[1] DFKI German Research Centre for Artificial Intelligence
Trippstadter Str. 122, 67663 Kaiserslautern, Germany
gernot.bahle, andreas.poxrucker, george.kampis, paul.lukowicz@dfki.de
WWW home page: http://ei.dfki.de
[2] Eötvös University, Budapest, Hungary
[3] ITMO University, St. Petersburg, Russia

Abstract. To investigate incremental collaborative classifier fusion techniques, we have developed a comprehensive simulation framework. It is highly flexible and customizable, and can be adapted to various settings and scenarios. The toolbox is realized as an extension to the NetLogo multi-agent based simulation environment using its comprehensive Java-API. The toolbox has been integrated in two different environments, one for demonstration purposes and another, modeled on persons using realistic motion data from Zurich, who are communicating in an ad hoc fashion using mobile devices.

Keywords: ad hoc communication, incremental classifier fusion, collaborative computing

1 Introduction

We consider an environment in which agents can classify observations they make. The nature of this environment as well as the concept represented by an agent is thereby not fixed to a certain setting, but can be adapted to different scenarios (see applications). To classify testing data agents encounter during a simulation, every agent is associated with a certain knowledge, which essentially encapsulates an instance of a certain type of classifier.

In machine learning and statistics, classification is the problem of identifying an item to which an observation belongs, taken from a set of categories (sub-populations). Classification learning is the development of classifications by machine learning techniques. Classification learning is performed on the basis of a training set, that is, data containing observations (or instances) whose category membership is known. This known membership is used a feedback to direct a convergent learning processes to the right outcomes. Trained classifiers are statistical functions that use pattern matching to determine a closest match.

Every knowledge instance is initially trained using a set of training data. The latter may contain an equal amount of data for each class or may be biased towards certain classes with more training data initially available for certain classes than for others. The quality of classification can be described by e.g. the global average of true positive rates for every class.

During our simulations, agents move around in the world either randomly or following special movement patterns. Whenever two agents meet, they merge their classifiers based on a certain classifier fusion technique and update their classification output. The concepts of movement and meeting are thereby purely abstract and can be realized according to the specific needs of a certain scenario or research interests. Keeping track of the global average true-positive rate for every class, it is possible to observe how the classification quality changes in dependence of the chosen fusion method.

We note that knowledge fusion in general and classifier fusion as a special example are well studied in Artificial Intelligence (REF) and off-the-shelf methods exist with optimal properties [7]. By contrast, collaborative (incremental) classifier fusion is a new idea. The difference is that in the former, the entire set of knowledge instances (trained classifiers) must be simultaneously available; in the latter, this condition is relaxed. Collaborative fusion is that is a natural generalization to situations where access to knowledge items is limited by temporal, geographic or other constraints.

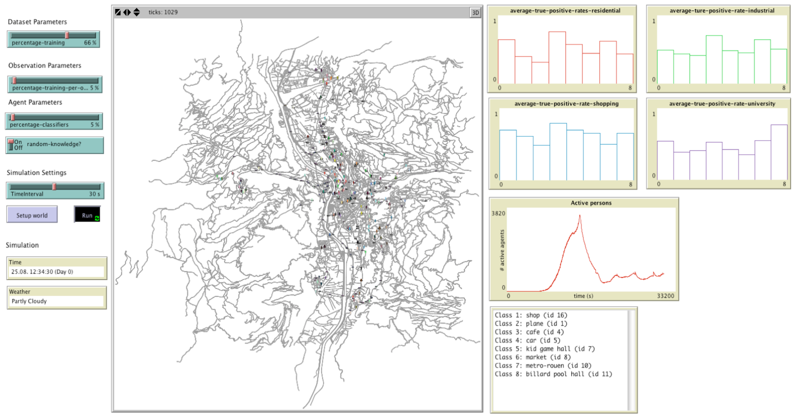
We have developed a simulation framework realized as an extension to the NetLogo multi-agent based simulation environment [8] using a comprehensive Java-API. We have applied it in three different settings to demonstrate the flexibility and adaptivity of our approach. In all three cases, we used classifiers from the WEKA library. For fusing classifiers, we experimented with the baseline case of simply exchanging training data as well as advanced classifier aggregation functions offered by WEKA. As data basis, we used a real-world environmental audio data set, based on a public dataset from the university of Rouen, France. Recordings have been performed using a Galaxy S3 smartphone equipped with Android by means of the Hi-Q MP3 recorder application. The dataset is composed of 19 classes and audio scenes forming a given class have been recorded at different locations. The classifier task is to approximate the original classification, and our populational learning algorithms aim to achieve that by incremental fusion [4, 3]. The current work belongs to a series of studies that started with simple collaborative tasks such as localization [2, 5] and developed through the study of fully distributed ad-hoc communication systems [1].

2 Different Motion Models

2.1 Adaptive, Personalized Multi-modal Urban Mobility in the Allow Ensembles Simulator

First we used the movement model from the Allow Ensembles simulator [6] for multi-modal urban mobility (Fig. 1). In this scenario, we assume an urban area (show on the example of Trento, Italy) where people can use various means of transportation to travel. These means include walking, cycling, going by car or taxi, or using public transportation like buses or trains. Whenever persons physically meet in the streets or share the same bus, classifier fusion is triggered by the meeting and the persons exchange classification information. What information is exchanged and how, it is determined by the respective fusion strategy tested.

We examined two different cases. In the first case, the agent population is divided into groups based on the region (such as residential, industrial, university, shopping) from where they come. Every member of a group is in turn assigned an initial training set, which is similarly biased within the same group providing different levels of expertise for the considered classes. In the second scenario, we divided the population based on the respective role of the person in the city population, e.g. worker, student, homemaker, or child. Results are discussed in a separate section.



Dataset Parameters

percentage-training 66 %

Observation Parameters

percentage-training-per-o... 5 %

Agent Parameters

percentage-classifiers 5 %

On Off random-knowledge?

Simulation Settings

TimeInterval 30 s

Simulation

Time: 25.08. 12:34:30 (Day 0)

Weather: Partly Cloudy

ticks: 1029

```
Class 1: shop (id 16)
Class 2: plane (id 1)
Class 3: cafe (id 4)
Class 4: car (id 5)
Class 5: kid game hall (id 7)
Class 6: market (id 8)
Class 7: metro-rouen (id 10)
Class 8: billard pool hall (id 11)
```

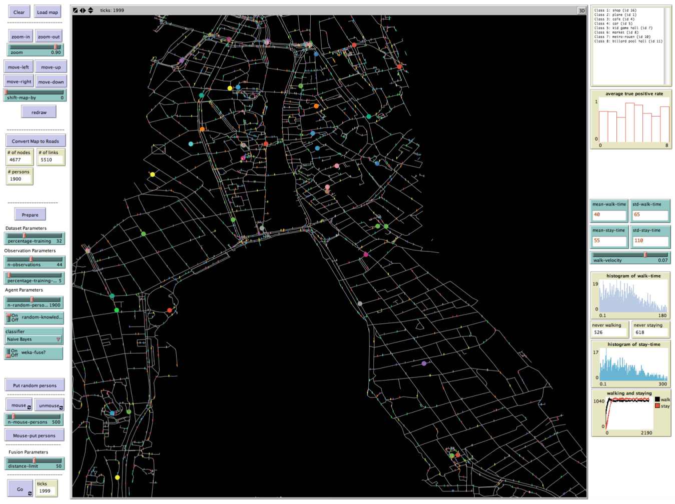
Setup world
Run
3D
average-true-positive-rates-residential
average-ture-positive-rate-industrial
average-true-positive-rate-shopping
average-true-positive-rate-university
1
0
0
8
Active persons
3820
# active agents
0
0
time (s)
33200

Fig. 1. Collaborative fusion based on the Allow Ensembles simulator

2.2 Using a Real-world Movement Model From Zrich

In a second case study we have been using a dataset obtained in the 2013 Zurich festival. The dataset contained GPS positions of users of a smart phone app. On this basis we estimated pedestrians' key walking parameters. We obtained data for average walking speed and average walking/standing times, together with the respective error terms. The current approach uses these numbers as averages over the various crowd densities experienced in the dataset, referring to the fact that in the interesting range (3 - 4,000 pedestrians using the app) these parameters do not change dramatically over the time.

In this application, we use GIS shape files of the city (the simulation is thus ready to work for any city, but we only have validated the motion data for the Zurich example). We represent streets by their segments, and our motion model consists of navigation along the network defined by the resulting segments. Pedestrians share a common average walking speed but have individual walking and staying times, taken from two normal distributions, which are characterized by the average walking/standing times and their standard errors, respectively. Pedestrians are always heading towards an adjacent segment and upon reaching it select next a new adjacent destination segment, but never turn back in a single step (Figure 2).



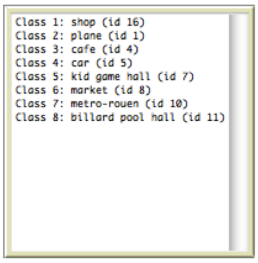
zoom 0.90
shift-map-by 0
# of nodes 4677
# of links 5510
# persons 1900
percentage-training 32
n-observations 44
percentage-training-... 5
n-random-perso... 1900
On Off random-knowled...
classifier Naive Bayes
On Off weka-fuse?
n-mouse-persons 500
distance-limit 50
ticks 1999

Class 1: shop (id 16)
Class 2: plane (id 1)
Class 3: cafe (id 4)
Class 4: car (id 5)
Class 5: kid game hall (id 7)
Class 6: market (id 8)
Class 7: metro-rouen (id 10)
Class 8: billard pool hall (id 11)

mean-walk-time 40
std-walk-time 65
mean-stay-time 55
std-stay-time 110
walk-velocity 0.07
never walking 526
never staying 618

Fig. 2. Using the Zurich motion dataset

In each of the two case studies, we applied 8 artificial classes (Figure 3).



```
Class 1: shop (id 16)
Class 2: plane (id 1)
Class 3: cafe (id 4)
Class 4: car (id 5)
Class 5: kid game hall (id 7)
Class 6: market (id 8)
Class 7: metro-rouen (id 10)
Class 8: billard pool hall (id 11)
```

Fig. 3. The artificial classes used in the collaborative classifier fusion experiments

3 Classifier Fusion Strategies

Based on the simulation tools above, many different settings for classifier fusion can be tested. The following section gives an overview of different initial settings and strategies for action when two agents meet.

At the start of each simulation, there is a choice on how to initialize the training data each agent has available. Currently, there are two options: a random subset of all training data, or the first k% of the training data for each class. A random selection can, after enough knowledge exchange, potentially lead to a complete training set for agents, while the second alternative provides better control about what information is available at all. Also, the second option provides a natural hierarchy of experts (i.e. the training data of all agents knowing less than 70% are subsets of the experts data).

For the actual classifier fusion, we consider the following strategies:

- fusion based on exchange of training data. This strategy can serve as a baseline for other algorithms. Whenever agents meet, they exchange their set of training data. Depending on the initialization, this will eventually lead to either a complete set of training data (for the random method) or a convergence towards the level of the best informed expert (for the first k% method). Once the data has been exchanged, classifiers are retrained using the new set.
- fusion based on classifier models. This strategy involves fusing classifiers themselves, without resorting to the training data. In the case of the Naive-Bayes classifier, which specifies a standard deviation and mean for each class, this involves aggregating those two values. In the simplest case, this could be done via averaging. More elaborate schemes incorporate weighting to incorporate the reliability of a given classifier (e.g. because of a larger set of training data). In more general terms, given two classifiers C and D fully described by a set of parameters P and Q, the aim is to provide a function $f : P \times Q \rightarrow R$, so that the new classifier E incorporates the knowledge of C and D. Note that this general principal can be applied both to classifiers of the same type (e.g. two trees, two neural networks) as well as to heterogeneous ones. In the latter case, concrete strategies will have to be tailored to each specific pairing (e.g. Bayes and Tree).
- fusion based on artificial training data. In order to provide a more general way to fuse two different types of classifiers while at the same time not relying on storage or exchange of training data, we are proposing yet another strategy. Based on the trained classifier an agent already has, it is possible to create random points of the feature space (or guided by some heuristic), label them via the known classifier and provide those to another agent. The agent can then incorporate them into its own classifier, either by some incremental learning algorithm or by simply retraining. This method abstracts from the specific type of classifier, has however the drawback of transmitting incorrectly labeled data some of the time.

4 Implementation

Classifiers. Classifiers are represented by an instance of the Knowledge class which offers an interface for training the classifier, obtaining classification results given some test data, and fusing it with another classifier instance. With this approach the actual implementation of the classifier is hidden from the simulation itself making it easily possible to experiment with different types of classifiers and classifier fusion methods. Even different types of classifiers are possible within the same simulation run. In our current implementation (see application scenarios below), we use the WEKA library which offers a wide range of different classifiers.

Classifier Fusion. The implementation of the fusion mechanism can be done in different ways. In the simplest case, the agents simply exchange their training data sets and retrain their classifier instances based on the merged training data. Although this approach does not involve an algorithm fusing the actual instances of the classifier can be used as a baseline. For some classifiers like the Naive Bayes for example, the WEKA library already defines methods for merging. For other classifiers or classifier combinations, it will be necessary to develop merging algorithms.

Data Basis. Just like the classifiers, the implementation of our toolbox does not impose any restrictions on the data basis to be used for training and testing. In our current implementation for example, we use a database of audio files for which we computed a set of features and stored them in the WEKA ARFF format in order to be compatible with the WEKA classifiers. It would, however, be easily possible to experiment with different data sets instead (provided they are converted to the ARFF format in case WEKA is used). In case of a completely different classification framework the simulation can be easily adapted to the necessary file formats.

5 Results

5.1 The Trento case study

In the Trento urban scenario, we are comparing two different scenarios:

1. (i) agents have initial knowledge based on their role (worker, student, etc.)
2. (ii) agents have initial knowledge based on their geographic position

Furthermore, for each of those cases, we look at the exchange of training data vs. the parameter based fusion of the classifiers themselves. In both cases, data has been classified using a NaiveBayes classifier provided by WEKA, and training data has been assigned randomly. However, for the two different fusion methods, the same random seed has been used, so the initial distribution is the same for both fusion algorithms. The following results have been obtained.

Role based knowledge. Figure 4 shows the initial distribution for scenario 1 (role based); Figures 5 left and right, respectively, show the true positive rate of classifiers for each class after a virtual day has elapsed for fusion based on training data and parameter based classifier fusion. As can be seen, the exchange of training data works better for all classes in general. This is to be expected, because this method grows the training data set for all agents, resulting in a more and more complete set as the day moves on. However, parameter based classifier fusion performs reasonably well if there already is some initial useful knowledge to build on (see for example classes one, four and seven). A second interesting effect: the number of interactions matters a lot more in the case of parameter based fusion. In both scenarios, the class of agents least interacting are homemakers (based on their profile, they leave their home to bring children to school or go shopping). As visible from Figure 5 (left side), the exchange of training data is less sensitive to a small number of interactions; this is probably the case because of the cumulative effect of size increases in training data sets.

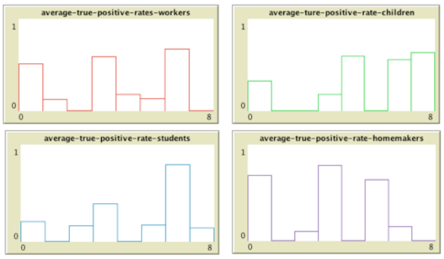


Fig. 4. Knowledge by role, random initial distrubution for each role (worker, children, students,homemakers). Figures show the average positive classification rate.

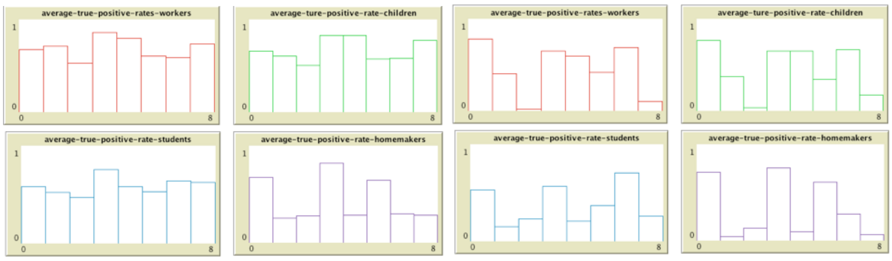


Fig. 5. Knowledge by role after 1 day. Left: training data exchange. Right: parameter fusion. While perfect classification is never achieved, in some categories and some roles the ideal situation is well approximated (especially on the left panel).

Geography based knowledge. Figure 6 shows the initial distribution for scenario 2, where initial training data is based on geographic information. Figures 7 left and right, respectively, once again show final true positive rates after a virtual day has elapsed.

Once again, the baseline fusion of training data outperforms the classifier fusion approach. However, other than in the role-based scenario discussed above, a strong initial knowledge of classes does not automatically yield a better performance for parameter based classifier fusion here. This is due to the reduced ”mingling” of agents in this scenario. While role based agents (workers, students, etc.) can come from many different areas of origin and have many different destinations, in the geography based scenario, such diversity is reduced. For instance, workers tend to live close to their area of work, and thus many interactions will actually draw from the same set of training data.

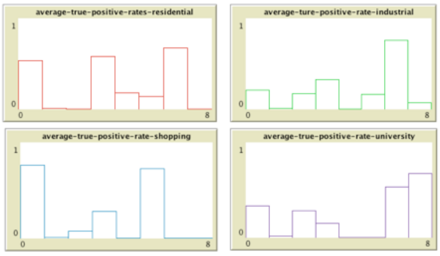


Fig. 6. Knowledge by region, random initial distribution for each region (residential, industrial, shopping, university).

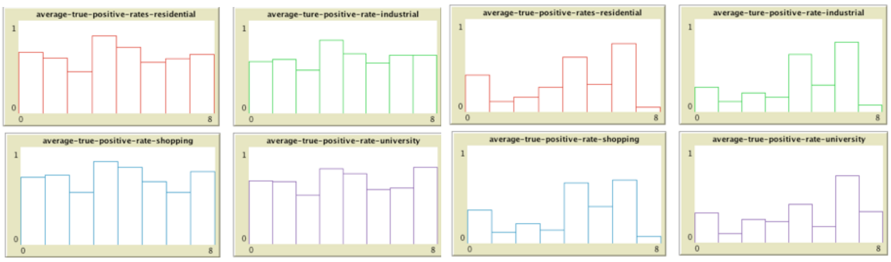


Fig. 7. Knowledge by region after 1 day. Left: training data exchange. Right: parameter fusion. Again, while perfect classification is never achieved, in some categories and some regions this ideal situation is approximated (especially on the left panel).

5.2 The Zurich case study

The Zurich case study uses real motion data in a statistical simulation. Its basis is the daily recording of pedestrian's motion speed and its variance during the 2013 Zurich city festival.

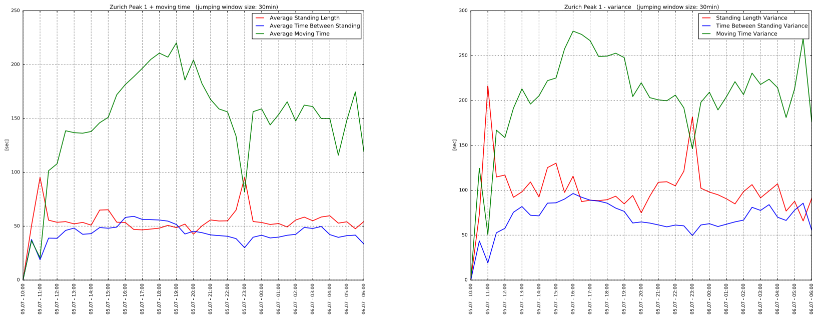


Fig. 8. Moving time (left) and its variance (right) in the Zurich motion dataset. The simulation uses the average values; note that the actual motion speed of the crowd is uneven and motion comes to a full halt at a given point of time (when the fireworks are on). Left: green - average moving time; blue - average time between standing; red - average standing time. Right: variances of the same.

Sample walking data are presented on Figure 8. The motion model is based on a "mean field" approximation of the pedestrian population, hence we use the averages for walking speed 0,5 m/s, standing length 55s (std 100) and time between standing 40s (std 65), respectively. We take these as fixed external parameters in our simulations to test the internal parameters of the fusion algorithm. Simulated pedestrian motion is generated from a Gaussian distribution with a mean and variance as above.

With the motion model realized, we find that about half of all agents are immobile at any given time, about 20% never move, and another 20% never stand still (those values seem intuitive for anyone having visited a street festival). These facts will be significant when evaluating simulation results.

On Figure 9, the available classifications are shown in the same histogram as above; the histogram comes from the top panel of the Zurich simulation interface.

In a comparison of the Trento and Zurich models, that is, the original large-scale collaborative learning simulation model with the merged "super model" of Zurich shows that learning and convergence take place in both but in the Zurich case in an understandably slower rate as there are pedestrians that never meet and therefore do not enter (or can be improved) by knowledge fusion.

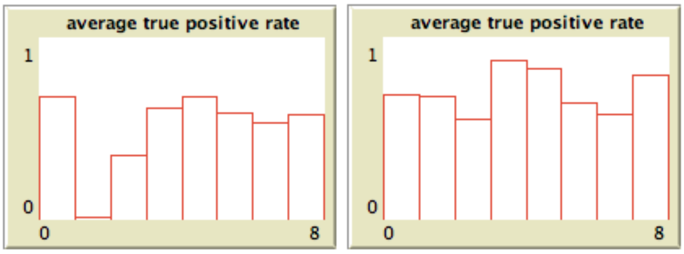


Fig. 9. Initial random (left) and evolved (right) classifications in the Zurich example. Figures show the average positive classification rate. Evolved classifications shown after 2,000 time steps (scaled for 200 minutes).

6 Acknowledgements

This work is supported by the European Community (FP7/2007-2013) under grant agreement #600854 “Smart Society” as well as the H2020 Program under “FET Proactive: Global Systems Sciences” (GSS), grant agreement #641191 (CIMPLEX, `https://www.cimplex-project.eu`). The paper is partially supported by the Russian Scientific Foundation, grant #14-21-00137.

References

1. T. Franke, G. Kampis, and P. Lukowicz. Leveraging human mobility in smartphone based ad-hoc information distribution in crowd management scenarios. Submitted to MobiSys 2015. `http://www.sigmobile.org/mobisys/2015/`.
2. G. Kampis, J.W. Kantelhardt, K. Kloch, and P. Lukowicz. Analytical and simulation models for collaborative localization. *J. Computational Science*, 6(1):110, 2015.
3. G. Kampis and P. Lukowicz. Collaborative knowledge fusion by ad-hoc information distribution in crowds. ICCS 2015, International Conference on Computational Science, Reyjavik.
4. G. Kampis and P. Lukowicz. Collaborative localization as a paradigm for incremental knowledge fusion. 5th IEEE CogInfoCom 2014 Conference, 2014. `http://coginfocom.hu/conference/CogInfoCom14/downloads/Program_CogInfoCom_2014_final.pdf`.
5. K. Kloch, P. Lukowicz, and C. Fischer. Collaborative PDR localisation with mobile phones. In *Proceedings of the 2011 15th Annual International Symposium on Wearable Computers, ISWC 11*, pages 37–40. IEEE Computer Society, Washington, DC, USA, 2011.
6. A. Poxrucker, G. Bahle, and P. Lukowicz. Towards a real-world simulator for collaborative distributed learning in the scenario of urban mobility. In Self-Adaptive and Self-Organizing Systems Workshops (SASOW), 2014 IEEE Eighth International Conference, pp. 44-48, 2014.
7. D. Ruta and B. Gabrys. An overview of classifier fusion methods. *Computing and Information systems*, 7(1):1–10, 2000.
8. S. Tisue and U. Wilensky. Netlogo: A simple environment for modeling complexity. In *International conference on complex systems*, pages 16–21, 2004.