# TOWARDS ROBUST FALL DETECTION

*Violeta Mirchevska[1], Mitja Luštrek[2], Matjaž Gams[2]*
[1]Result d.o.o., Bravničarjeva 11, 1000 Ljubljana, Slovenia
[2]Department of Intelligent Systems, Jožef Stefan Institute, Jamova 39, 1000 Ljubljana, Slovenia
e-mail: violeta.mircevska@result.si

**ABSTRACT**

This paper presents a method for classifier development by combining domain knowledge and machine learning. The development is performed in two phases: (1) development of initial hypothesis using domain knowledge or interactive machine learning and (2) refinement of the initial hypothesis using genetic algorithms. The method is presented in the domain of fall detection.

## 1 INTRODUCTION

Recently, there has been a growing interest in innovative ICT solutions that would aid the elderly to live independently for longer and counteract reduced capabilities caused by age. In this respect, sustainable and personalized healthcare has become one of the strategic interests of the European society. The aging of the population in Europe causes immense pressure on the healthcare expenditures, which already account for 9% of the EU's GDP spending [1]. The development of personalized healthcare systems is one of the research focuses of the fifth challenge of the ICT Work Programme under FP7. These systems would provide more effective care of patients by monitoring patients' health conditions using wearable, portable and implantable systems, providing health professionals with comprehensive monitoring and diagnostic data.

The European FP7 project Confidence – Ubiquitous care system to support independent living [2] aims at developing a system that will monitor the health conditions of its elderly user in real-time. It encompasses detection of falls as well as changes in behavior, such as limping, slow moving and physical inactivity. In case a health problem is detected, the system issues a warning to the user and alerts a caregiver if necessary. This way the system would provide the elderly with confidence to continue to live independently at home as long as possible.

In this paper we focus on the fall detection part of Confidence [3]. Development of a fall detection classifier was challenging because of the following reasons: First of all, high reliability is needed since, on one hand, undetected falls may be disastrous for the user and, on the other, too many false positives may be disturbing for the user and may lead the user to be unwilling to use the system. Second, representative dataset for falls is difficult to obtain because of the variety of fall types, variations depending on the user, as well as ethical issues and injury danger which prevent collection of large amounts of data by healthy persons simulating falls. Non-representative dataset may cause classifier overfitting and poor performance in the general case. The problem that we address in this paper is how to create a robust fall detection classifier when only a limited amount of data from the domain of interest is available.

We addressed the problem of generation of a robust fall detection classifier by combining domain knowledge (DK) and machine learning (ML). DK and ML complement each other. ML algorithms can discover characteristic domain patterns which may be too subtle for humans to detect, but they can only discover patterns that are present in the training dataset. DK, on the other hand, may be related to examples not present in the available domain dataset. Therefore, their combination may improve the reliability of the developed classifiers, if done in a proper way.

In the following, we present preliminary study of a method of development of rule-based classifier by combining DK and ML and compare it to classifiers developed by known ML algorithms that are in the form of rules, or that may be converted to rules.

This paper is organized as follows: Section 2 presents related work on the topic of methods for classifier development by combining DK and ML. Section 3 presents the approach for development of a fall detection classifier by combining DK and ML. Its evaluation is presented in Section 4. Section 5 concludes the paper.

## 2 RELATED WORK

Different approaches for model development by combining DK and ML are found in the literature.

A comprehensive overview of methods for incorporating DK into inductive ML is presented by Yu [4], who categorizes these methods in four groups: (1) methods that use prior DK to prepare training examples [5][6], (2) methods that use prior DK to initiate the hypothesis or hypothesis space [7][8], (3) methods that use prior DK to alter the search objective [9][10] and (4) methods that use prior DK to augment search [11][12].

In addition to this, methods for combining DK and ML are found in the field of interactive ML. Interactive ML basically refers to an iterative process of classifier generation by human-computer interaction. Two strategies for model generation using interactive ML can be distinguished: (1) iterative improvement of a single model by refining the input information used during ML induction [13][14] and (2) generation of multiple models in order to select one or several that are the most relevant from the user point of view [15][16]. In principle, the combination of ML and DK improves the generality of the generated model and/or the efficiency of the learning process.

## 3 METHOD

The method for model generation by combining ML and DK, an extension of the method presented in [17], can be separated in two phases:

- Determination of initial hypothesis by DK or interactive ML
- Refinement of the initial hypothesis by genetic algorithms based on data of the domain of interest.

Detailed presentation of the approach for determination of initial hypothesis is presented in [17]. Basically, an expert determines the format of the rules in the classifier by DK. The expert may explore the data of the domain of interest by generation of ML models and extract patterns from these models to be included in the rule-based classifier.

The initial hypothesis is refined by genetic algorithms.

Genetic algorithms [18] are stochastic search algorithms, whose search strategy mimics evolution and natural selection. Genetic algorithms have been used for solving certain optimization subproblems in machine learning. Kononenko and Kukar [19] identify four machine learning subproblems that involve optimization in a large search space and which may be addressed by genetic algorithms: feature subset selection, parameter tuning, constructive induction and hypothesis learning. Genetic algorithms have been applied for solving such machine learning subproblems in rule induction, in decision tree induction, in neural networks as well as in support vector machines induction.

Genetic algorithms are suitable for refining the initial hypothesis because the structure of the rules in the initial hypothesis is fixed and standard heuristics used for induction of rules are not suitable in our case. In addition to this, constraints defining relations between rules as well as relations between parameters within a rule can be specified in a straightforward way when using genetic algorithms.

We have used the Pittsburg approach for rule discovery using genetic algorithms, meaning that each individual in the population represents one possible solution. The individual is basically a vector which contains the parameters of all rules in the rule-based classifier. The elements are real valued and take values in a predefined interval. The fitness function used for evaluating the quality of each individual is the accuracy of the solution on the training dataset; fitness values fall within the interval [0, 1]. Elitism was used, meaning that the best individual is always transferred in the new population. By this we want to tune the parameters of the rule-based classifier as good as possible to the training dataset. Overfitting should be avoided because the structure of the rule-based classifier is defined by domain expert.

## 4 EVALUATION

The evaluation of the presented method was done in the domain of fall detection. The rule-based classifier developed for the need of fall detection is based on the pattern that if an elderly is lying or sitting on the ground for long period of time, then there is high probability of a fall. It contains the following types of rules:

1. IF falling activity within $T1_{fall}$ seconds AND the user was lying/sitting on the ground $P1_{activity}$% in $T1_{activity}$ seconds AND the user was not moving $P1_{moving}$% in $T1_{moving}$ seconds THEN fall
2. IF falling activity within $T2_{fall}$ seconds AND the user was lying/sitting on the ground area afterwards $P2_{activity}$% in $T2_{activity}$ seconds THEN fall
3. IF the user was lying/sitting on the ground for $P3_{activity}$% in $T3_{activity}$ seconds AND the user was not moving $P3_{moving}$% in $T3_{moving}$ seconds THEN fall
4. IF the user was lying/sitting on the ground for $P4_{activity}$% in $T4_{activity}$ THEN fall

A test scenario that contains clear-case and complex events of falls was designed in order to test the reliability and the robustness of the developed classifier. The clear-case events contain typical fall and non-fall events, such as normal behavior (e.g. standing, sitting, lying in bed), tripping, falling from chair when trying to stand up and searching for something on the ground on all fours and lying. The complex events represent atypical falls and non-fall events which may be easily mistakenly classified as falls, such as falling slowly trying to hold onto furniture from standing, lying down quickly on the bed, sitting down quickly on the chair, falling slowly trying to hold onto furniture when standing up from the chair or sitting on a low chair. The fall examples of the complex events do not have high acceleration towards the ground during the falling activity, which is a characteristic feature of falls, whereas the non-fall events contain high acceleration towards the ground during going down activity and activities, such as sitting on a low chair, that may be misclassified as lying or sitting on the ground.

All events present in the test scenario were recorded in single recordings interspersed with short periods of walking. Each recording lasted around 20 minutes. The recordings were made by 5 healthy volunteers (3 male and 2 female), 5 times by each.

The classifiers were generated using the straightforward events only. We used leave-one-person-out evaluation, meaning that the classifier was generated on the examples from four persons and tested on the examples of the fifth person, which was excluded from the training dataset. Firstly, the initial classifier was specified by domain expert. Then genetic algorithms were used for tuning the initial classifier to suit system- and user-related characteristics creating the refined classifier using the examples in the clear-case events. The accuracy of the classifiers generated in this way was tested on both clear-case and complex events of the person excluded from the training dataset. The test on the clear-case events shows how well the classifier performs on types of events present in the training dataset. The test on the complex events, on the other hand, is used to test the generality and robustness of the generated classifier, since the complex events are not present in the learning process. For comparison, fall detection classifiers were induced using machine learning only with attributes equivalent to the parameters of the rules in the rule-based fall detection classifier. We used decision trees (J48 in Weka [20]) and rules (JRip in Weka [20]) because these algorithms create models that can be converted to or are a set of rules.

Table 1 presents a comparison of the performance of the obtained fall detection classifiers. J48 and JRip seem to be biased toward recognition of fall events. They detected all fall examples in the clear-case and complex events, at the same time raising false positives in many cases. The initial classifier is less biased towards recognition of falls, as experts introduce patterns for which they are sure that are relevant for the recognition of the class of interest. This, however, causes certain fall examples not to be recognized. Finally, the refined classifier significantly increases the precision of the classifier on the clear-case scenarios. The refinement of the initial classifier with genetic algorithms contributes to a slight increase in the performance of the classifier as compared to the initial classifier.

## 4 CONCLUSION

The combination of DK and ML may increase the credibility and performance of the developed classifier in the general case. On one hand, ML algorithms can discover characteristic domain patterns which may be too subtle for humans to detect, but they can only discover patterns that are present in the training dataset.

In this paper, we present a preliminary study of a method for generation of a classifier by combining DK and ML. The presented method can be divided into two phases: (1) creation of initial hypothesis by DK or interactive ML and (2) refinement of the initial hypothesis using genetic algorithms. Preliminary tests show that the incorporation of DK in the learning process improves the performance of the classifier in the general case. Additionally, tests show that refinement by genetic algorithms may contribute to additional performance improvement.

As future work, we need to apply the presented method in other domains in order to test its generality. We need to explore different settings of the genetic algorithm in the refinement phase (e.g. different fitness functions, different recombination possibilities).

## ACKNOWLEDGEMENTS

This research is partly financed by the European Union, European Social Fund and partly by the European Community's Framework Programme FP7/2007–2013 under grant agreement No. 214986.

**Table 1 Classifier comparison**

| | *CLEAR-CASE EVENTS* | | | *COMPLEX EVENTS* | | | *ALL EVENTS* | | |
|---|---|---|---|---|---|---|---|---|---|
| | Precision | Recall | F-measure | Precision | Recall | F-measure | Precision | Recall | F-measure |
| J48 | 0.76 | 1.00 | 0.86 | 0.28 | 1.00 | 0.44 | 0.41 | 1.00 | 0.58 |
| JRip | 0.76 | 1.00 | 0.86 | 0.27 | 1.00 | 0.43 | 0.40 | 1.00 | 0.57 |
| Initial classifier | 0.84 | 0.98 | 0.91 | 0.33 | 0.98 | 0.49 | 0.48 | 0.98 | 0.64 |
| Refined classifier | 0.94 | 0.91 | 0.92 | 0.34 | 0.96 | 0.51 | 0.50 | 0.93 | 0.65 |

## REFERENCES


[1] European commission, ICT research in FP7, ICT challenge 5: Towards sustainable and personalized healthcare. http://cordis.europa.eu/fp7/ict/programme/overview5_en.html. 01.09.2011

[2] Confidence. http://www.confidence-eu.org. 01.09.2011

[3] B. Kaluža, V. Mirchevska, E. Dovgan, M. Luštrek, and M. Gams (2010) An agent-based approach to care in independent living, in Proceedings of AmI 2010, pp. 177-186

[4] T. Yu (2007) *Incorporating Prior Domain Knowledge into Inductive Machine Learning Its implementation in contemporary capital markets*, PhD thesis

[5] S. Kambar (2005) *Generating synthetic data by morphing transformation for handwritten numeral recognition (with v-svm)*, Master thesis

[6] P. Niyogi, F. Girosi, T. Poggio (1998) Incorporating prior information in machine learning by creating virtual examples, in Proceedings of the IEEE, pp. 2196-2209

[7] Z. Zhu, P. Liu (2010) Feasibility research of text information filtering based on genetic algorithm, *Scientific Research and Essays* 5(22), pp. 3405-3410

[8] B. D. Burns, A. P. Danyluk (2000) Feature Selection vs Theory Reformulation: A Study of Genetic Refinement of Knowledge-based Neural Networks, *Machine Learning - Special issue on multistrategy learning*, pp. 89-107

[9] I. Davidson, S. S. Ravi (2005) Hierarchical clustering with constraints: Theory and practice, in Proceedings of the Ninth European Principles and Practice of KDD (PKDD), pp. 59-70

[10] X. Wu, R. Srihari (2004) Incorporating prior knowledge with weighted margin support vector machines, in Proceedings of the tenth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp.326-333

[11] D. Decoste, B. Scholkopf (2002) Training invariant support vector machines machine learning, *Machine Learning 46*, pp. 161-190

[12] M. Pazzani, C. Brunk, G. Silverstein (1991) A knowledge-intensive approach to learning relational concepts, in Proceedings of the Eighth International Workshop on Machine Learning, pp. 432-436

[13] S. Sun, D. R. Hardoon (2010) Active learning with extremely sparse labeled examples, *Neurocomputing 73*, pp. 2980-2988

[14] S. Stumpf, V. Rajaram, L. Li, W.-K. Wong, M. Burnett, T. Dietterich, E. Sullivan, J. Herlocker, (2009) Interacting meaningfully with machine systems: Three experiments, *International journal of human-computer studies 67*, pp.639-662

[15] V. Vidulin, M. Gams, (2011) Impact of higher-level knowledge on economic welfare through interactive data mining, *Applied artificial intelligence*, pp. 267-291

[16] K.-M. Osei-Bryson, (2004) Evaluation of decision trees: a multi-criteria approach, *Computers and Operations Research*, pp. 1933-1945

[17] V. Mirchevska, M. Gams, (2009) Towards robust rule engine for classifying human posture, in Proceedings of the 12$^{th}$ international multiconference IS 2009

[18] A. E. Eiben, J. E. Smith, (2003) Introduction to Evolutionary Computing

[19] I. Kononenko, M. Kukar, (2007) Machine learning and data mining

[20] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, I. H. Witten, (2009) The WEKA Data Mining Software: An Update, *SIGKDD Explorations11(1)*, pp. 10-18