22nd European Conference
on Artificial Intelligence (ECAI 2016)

6th International Workshop on Combinations of Intelligent Methods and Applications

(CIMA 2016)

Proceedings

30 August 2016 - The Hague, Holland

Editors :
Ioannis Hatzilygeroudis
Vasile Palade

22nd European Conference on Artificial Intelligence
(ECAI 2016)

Proceedings

6th International Workshop on Combinations of Intelligent Methods and Applications (CIMA 2016)

August 30, 2016
The Hague, Holland

Ioannis Hatzilygeroudis, Vasile Palade
Editors

Table of Contents

Workshop Organization

Chairs-Organizers

Ioannis Hatzilygeroudis
University of Patras, Greece

Vasile Palade
Coventry University, UK

Program Committee

Ajith Abraham, MIR Labs

Plamen Agelov, Lancaster University, UK

Abdulrahman Atahhan, Coventry University,(UK)

Nick Bassiliades, Aristotle University of Thessaloniki (Greece)

Maumita Bhattacharya, Charles Sturt University, Australia

Kit-Yan Chan, Curtin University (Australia)

Gloria Cerasela Crisan, University of Bacau (Romania)

Artur D'Avila Garcez, City University London,(UK)

George Dounias, University of the Aegean (Greece)

Foteini Grivokostopoulou, University of Patras (Greece)

Andreas Holzinger, Medical University of Graz (Austria)

Constantinos Koutsojannis, T.E.I of Patras (Greece)

George Magoulas, Birkbeck College (UK)

Christos Makris, University of Patras (Greece)

Ashish Mani, Dayalbagh Educational Institute (India)

Antonio Moreno, University Rovira i Virgili (Spain)

Daniel C. Neagu, University of Bradford (UK)

Muaz Niazi, Comsats Institute of IT (Pakistan)

Camelia Pintea, Technical University of Cluj-Napoca (Romania)

Isidoros Perikos, University of Patras (Greece)

Rudolf Kruse, University of Magdeburg, Germany

David Sanchez, University Rovira i Virgili (Spain)

Kyriakos Sgarbas, University Of Patras (Greece)

Jun Sun, Jiangnan University (China)

Douglas Vieira, Enacom-Handcrafted Technologies (Brazil)

Maria Virvou, University of Piraeus (Greece)

Contact Chair

Ioannis Hatzilygeroudis
Dept. of Computer Engineering & Informatics
University of Patras, Greece
Email: ihatz@ceid.upatras.gr

Preface

The combination of different intelligent methods is a very active research area in Artificial Intelligence (AI). The aim is to create integrated or hybrid methods that benefit from each of their components. It is generally believed that complex problems can be easier solved with such integrated or hybrid methods.

Some of the existing efforts combine what are called soft computing methods (fuzzy logic, neural networks, genetic algorithms, swarm intelligence methods) either among themselves or with more traditional AI methods such as logic and rules. Another stream of efforts integrates case-based reasoning with soft-computing and more traditional AI or machine learning methods. Yet another integrates logic-based agent approaches with non-symbolic approaches. Some of the combinations have been quite important and more extensively used, like neuro-symbolic methods, neuro-fuzzy methods and methods combining rule-based and case-based reasoning. However, there are other combinations that are still under investigation, such as those related to natural language processing and the Semantic Web. In some cases, combinations are driven by theoretical aspects, but usually they are created in the context of specific applications.

The Workshop is intended to become a forum for exchanging experience and ideas among researchers and practitioners who are dealing with combining intelligent methods either based on first principles or in the context of specific applications. This year it is held in conjunction with ECAI 2016. There were totally eleven papers submitted to the Workshop. Some of them were extended versions of papers accepted as short papers at ECAI 2016. Each paper was reviewed by at least two members of the PC. We finally accepted all of them, given they were of required quality. Revised versions of the accepted papers (based on the comments of the reviewers) are included in these proceedings in alphabetic order (based on first author). The papers involve various intelligent methods and application domains.

We hope that this collection of papers will be useful to both researchers and developers. Given the success of the first six Workshops on combinations of intelligent methods and applications, we intend to continue our effort in the coming years.

Ioannis Hatzilygeroudis

Vasile Palade

Investigation of Cellular Automata Neighbourhoods in Image Segmentation

Anca Andreica and **Laura Dioşan** and **Andreea Şandor**[1]

Abstract.

Cellular Automata (CA) can be successfully applied to the task of image segmentation. The CA-based GrowCut algorithm is able to perform such a task and we aim to investigate the full emergence phenomenon that arises during the segmentation process. In fact, we want to investigate how the segmentation performance could depend on the choice of the neighbourhood topology that is used by a CA-based algorithm. Several types of neighbourhoods are investigated. The experiments are performed by considering both synthetic and real-world images. The segmentation performance is analysed by using different criteria (evaluation measures). The numerical results indicate the way the neighbourhood topology influences the segmentation process.

1 Introduction

A large number of computer vision applications require image segmentation, which is a critical step towards content analysis and understanding the visual information. Frequently, the image segmentation is aimed to partition an image into separate regions, which ideally correspond to different real-world objects.

The partitioning task is even more difficult as the images are larger, with more dimensions (3D images or 4D images) or with an increased number of different objects that must be identified along them (a multi-label segmentation must be performed in these cases).

The image segmentation approach investigated in this paper is a region based method named GrowCut [37] and provides multi-label segmentation for N-dimensional images. It is based on a cellular automata and it has a few advantages that makes it a good candidate. In terms of implementation it is not very complex and the most important part is that it allows a very performant parallel implementation. Other advantages include the possibility to have multiple labelled objects at once, labels which can (but not must) be defined by the user. The process is iterative, observable and can be dynamically modified. This gives the user the possibility to guide the segmentation process for the difficult images, however in most cases such a guidance is not necessary.

The contribution of this paper is two-fold.

- First, we propose to investigate different neighbourhood topologies involved in the CA mechanism. The majority of related approaches found in the literature mainly deal with the von Neumann or Moore neighbourhood, without investigating how the shape and the size of the neighbourhood could affect the performance of segmentation.
- Secondly, we propose to compare the performance of the segmentation process by taking into account several supervised evaluation measures. This research direction was followed due to the variety of errors that can occur in the segmentation process: new added regions of interest or new added background regions, holes inside some regions or holes placed on the object's edges. Therefore, the evaluation of image segmentation has to take into account, simultaneously, how many segmented objects are produced, what is the area of these segments and what is the content of them (existence of inside holes and boundary holes in the segmented region). In addition, the evaluation has to consider the degree of correspondence between the segmentation results and the ground truth segmentation, actually the precision of segmentation as a measure of repeatability. Finally, but no less important, the temporal efficiency of segmentation process must be considered, also.

The paper is organized as follows: Section 2 presents related work in the area of Cellular Automata, Image Segmentation and Cellular Automata for Image Segmentation. The next section contains the reasons why we believe that different neighbourhoods could have an important impact on the performance of the algorithm, by presenting such approaches for other problems that can be solved by the means of Cellular Automata. Numerical experiments on both real-world and synthetic data are presented in Section 4, followed by conclusions and ideas for further work in Section 5.

2 Related work

2.1 Cellular Automata

The one-dimensional binary-state CA capable of performing computational tasks has been extensively studied in the literature [7, 16, 23, 25, 35]. Usually, a one-dimensional lattice of N two-state cells is used for representing the CA. The state of each cell changes according to a function depending on the current states in the neighbourhood. The neighbourhood of a cell is given by the cell itself and its r neighbours on both sides of the cell, where r represents the radius of the CA. The initial configuration of cell states (0s and 1s) for the lattice evolves in discrete time steps updating cells simultaneously according to the CA rule.

CAs have been considered for a series of applications like computer processors, cryptography, physical reality modelling, image processing and many others. In image processing for example, two-dimensional CAs are usually involved. The pixels of the image represent cells of the CA and they update their state based on the states of the neighbouring cells (pixels). Multiple states of CA cells allow the processing of gray-level images or colour images. Identifying the

[1] Babes-Bolyai University, Cluj-Napoca, email: anca@cs.ubbcluj.ro, lauras@cs.ubbcluj.ro, andreea.sandor@math.ubbcluj.ro

rules that apply to cells in order to answer a certain request in image processing is nevertheless a nontrivial task.

Three-dimensional CAs have mainly been used within the framework of chemical systems for tasks like percolation description, dissociation of organic acid in solutions, bond interactions, simulation of diffusion controlled reaction kinetics, dissolution and many others [18].

2.2 Image Segmentation

Image segmentation refers to the task of partitioning an image in sets of regions and therefore identifying objects of interest in it. Each pixel of the image is labelled in such a way that pixels with the same label share some features (like belonging to the same organ/bone/blood vessel in a medical image).

The segmentation of an image, I, can be defined as the partitioning of I in L sub-regions, R^1, R^2, ..., R^L, such that:

- $\bigcup_{l=1}^{nL} R^l = I$
- $\forall \widetilde{x} \in R^l, S(\widetilde{x}) = l, \forall l \in \{1, 2, \ldots, L\}$
- R^l is a connected set, $\forall l \in \{1, 2, \ldots, L\}$
- $R^{l_1} \cap R^{l_2} = \phi, \forall l_1, l_2, l_1 \neq l_2$
- $Q(R^l) = True, l \in \{1, 2, \ldots, L\}$
- $Q(R^{l_1} \cup R^{l_2}) = False$ for each pair of adjacent regions R^{l_1}, R^{l_2}.

where Q is a logical predicate defined on the points of the considered region (Q is used for characterizing the objects of the image).

Image segmentation techniques are still far from being able to identify relevant features (such as tumours or even simple regions). It is in this process that we propose to make use of CAs where, through appropriate choice of evolution rules and topologies, we can identify pixels which belong together. There are only few attempts in the literature at using CAs for image segmentation, but they confirm the scientific potential of the proposition.

2.3 Cellular Automata for Image Segmentation

Cellular Automata have been used for various image processing tasks among which: geometric transformations, noise filtering, feature detection, edge detection. Image segmentation was also approached by the means of Cellular Automata, but there are only few attempts in the literature.

Tasks as edge detection, noise filtering, thinning and finding the convex hulls of all regions have been solved by CAs in [29, 31, 32]. In [24, 31, 32] a thresholded local update rule is proposed for edge detection in gray-level images.

In [29] a threshold decomposition method is used in order to perform some image processing tasks in gray-level images. The goal of this research is to identify the optimal rules for solving these tasks. Their approach works in two stages: training (that is performed on a binary image) and testing the optimal rules for a gray-level image. Another gray-level image processing model was investigated in [30] based on a 3-state CA.

Multistate CAs (that, according to our knowledge, have not been used for image segmentation until now) can be: Sierpinski cellular automaton [26], a sandpile cellular automaton (or the Abelian sandpile model) [4], a Greenberg-Hastings automaton [13].

Sierpinski CA can be equivalent to a classical one-dimensional binary CA with Rule 90: during the CA evolution, all the values are simultaneously replaced by the exclusive *or* of the two neighbouring values.

The sandpile model is defined on a grid. The height of a sand pile is retained in a cell. During the CA evolution the height at one of the points increases. If a height exceeds the limiting value, the sand must be moved to nearby points until the height at all points are once again below the limiting value.

The Greenberg-Hastings automaton works on a regular grid of cells in one or more dimensions. This CA was described as a simple model generating spatio-temporal structures similar to those that can be observed in the Belousov-Zaikin-Zhabotinsky oscillating chemical reaction [13]. The first two items describe a reaction rule, while the last one describes a diffusion rule. This model is simple due to the fact that the diffusion rule and the reaction rule do not act together for a same cell at the same time. In two dimensions, the Greenberg-Hastings automaton can exhibit complex spiralling behaviours.

2.4 GrowCut Algorithm

One of the most popular approaches found in the literature for image segmentation using Cellular Automata is given by the GrowCut algorithm [37]. In our opinion, Cellular Automata are used in this approach only as a parallelization tool, without taking advantage of the full emergence phenomenon that arises in a Cellular Automaton.

The GrowCut algorithm is using as input several labelled seeds chosen by the user from the pixels of the image. The algorithm automatically labels all the other pixels belonging to the image starting from the given seeds. A Cellular Automaton is used for propagating the labels of the seeds throughout the whole image. A cell of the CA corresponds to a pixel of the image. A cell state is given by a label, a strength and a feature vector (in the simplest case, this corresponds to the gray level of the pixel). The labels and the strength of the cells are being updated based on the state of the neighbouring cells. The authors give an intuitive explanation of the method using a biological metaphor, the labelling process being thus interpreted as the growth and struggle for domination of a number of types of bacteria. This process stops when it reaches a stable configuration. The algorithm performance highly depends on the choice of the seeds.

In [17] the authors show that the seeded GrowCut proposed by Vezhnevets [37] is essentially no different from the Ford-Bellman algorithm that computes shortest paths from a cell to all the other cells in the CA. The fact that Cellular Automata is no more than a parallelization tool becomes more clear after understanding their proof.

An unsupervised version of GrowCut is proposed in [11]. The main difference between this approach and the original GrowCut is that the seeds are randomly generated and their label is adjusted over time, while in GrowCut the seeds are given by the user and they are expected to be correctly labelled from the beginning.

Another version of GrowCut, that improves its ability to correctly detect the edges, is proposed in [5]. The method uses an edge detection filter in order to detect that image voxels that find themselves on edges of the segments. Other variants of GrowCut are proposed in [14, 19]. In [27] the authors propose an enhancement of GrowCut with automatic seed selection. In [6] the image noise is reduced (and therefore the GrowCut algorithm improved) by adding an anisotropic diffusion filter.

3 Cellular Automata Topology and Neighbourhood

The CA topology and neighbourhood structure used for a cell in applying the rule are crucial elements in the process of rule discovery and impact directly the rule performance. In the case of a one-

dimensional lattice of N cells, the neighbourhood of a cell is usually given by the cell itself and its r neighbours on both sides of the cell, where r represents the radius of the CA. The regular lattice topology and the described induced neighbourhood are engaged in most studies tackling 1D Cellular Automata.

In the case of 2D Cellular Automata, the topology is usually a two-dimensional grid that allows the definition of different neighbourhood schemes. The most popular neighbourhoods used in this case are the von Neumann and the Moore neighbourhoods. The von Neumann neighbourhood is given by the set of all cells that are orthogonally-adjacent to the core cell (the core cell itself may or may not be considered part of the neighbourhood, depending on context). The Moore neighbourhood is the set of all cells that surround the core cell.

In the case of 1D Cellular Automata, different neighbourhood schemes have been investigated in order to study their influence on the rule performance. In [8, 9, 36, 38, 39], network topologies are evolved for cellular automata. In these studies it is shown that, compared to regular lattices, the evolved topologies have better performance for the CA majority and synchronization tasks.

A node-weighted network model proposed in [12] and extended in [3] allows the use of weights for each node in the network topology. As already mentioned, the state of each cell in CA changes according to a function depending on the current states in the neighbourhood. The neighbourhood of a cell is given by the cell itself and its neighbours. In current existing approaches, each neighbour (including the cell itself) has the same vote weight when deciding which is the next state of the current cell. The network topology allowed the introduction of neighbours with different vote weights when deciding the next state of the current cell. Thus, a node-weighted network is obtained where each node has a certain associated weight reflecting the varying importance represented by nodes.

A new hybrid topology and a mixed induced neighbourhood that keeps invariable the number of neighbours was also proposed in [1]. The neighbourhood of a node is given by the radius r. Each node has r neighbours on the left hand side and r neighbours on the right hand side, which gives a neighbourhood of $2 * r + 1$, because we also consider the node itself. In order to create the new topology of radius r, the authors started with a regular ring lattice of radius $r - n$. The other $2 * n$ nodes that node i is connected to, are long distance neighbours. They are randomly chosen from the rest of the nodes, but following some rules that ensure the equilibrium of the neighbourhood. This means that i always remains the central node of the neighbourhood and the distance between node i and the long distance neighbours places half of them (n nodes) on the left hand side and the other half (n nodes) on the right hand side of i.

The hybrid topology described before allowed the enhancement of the neighbourhood scheme with neighbours having different vote weights when deciding the next state of the current cell [2]. The hybrid topology involves the presence of two kinds of neighbours: local and far neighbours. The proposed rule gives different vote weights to local neighbours and to far neighbours.

The computational experiments performed for the density task emphasize that the proposed topologies and neighbourhoods induce an improved performance of the considered rules, compared to the standard ones. Moreover, the CA performance remains stable when dynamic changes are introduced in the neighbourhood structure.

These results have shown how novel topologies and neighbourhoods can trigger good performance in CA tasks and they can be adapted so as to improve segmentation techniques. Our research in this direction begins with the study presented in this paper, where several different neighbourhood schemes are investigated for the image segmentation task within the framework of the popular GrowCut algorithm, while most implementations of the GrowCut algorithm are using the Moore neighbourhood. The four neighbourhood schemes that are being studied in this paper are formalized below and depicted in Figure 1.

The von Neumann neighbourhood of a cell (x_0, y_0) is given by:
$N^V(x_0, y_0) = \{(x, y) : |x - x_0| + |y - y_0| \leq r\}$.
The Moore neighbourhood of a cell (x_0, y_0) is given by:
$N^M(x_0, y_0) = \{(x, y) : |x - x_0| \leq r, |y - y_0| \leq r\}$.
In both cases, r represents the range of the neighbourhood and $r = 1$ for the standard case. In our analysis we investigate von Neumann and Moore neighbourhoods of range 1 and 2.

4 Numerical experiments

4.1 Details about data

We have carried out experiments using some synthetic data and some real images. A first experiment was dedicated to analyze the performance of segmentation obtained by GrowCut with different neighbourhood schemes for two-dimensional synthetic images. Several regular geometric shapes (square, rectangle, circle) were automatically generated as foreground in images, with uniform intensity and with noise. The original synthetic images (with a circle, a rectangle and a square, respectively) are presented in Figure 2.

Regarding the real-world image, Berkeley Segmentation Dataset [21] and The Interactive Segmentation (IcgBench) Dataset [33] contain natural images with corresponding human segmentations (ground-truth) and we use some of them in order to validate our approach. One example of such two-dimensional images (and the corresponding ground-truth) is presented in Figure 3.

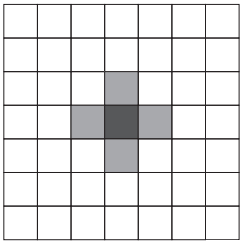
4.2 Details about performance measures

It is very important to establish the way we define similar regions or segmentations. The obtained segmentations and their boundaries could be compact, discontinuous, smooth etc.

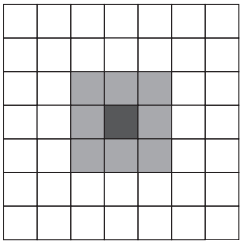
One of the metrics that we are using in our experiments is the Dice coefficient, which computes the overlap between regions, quantifying the similarity of two segmentations. This measure is especially useful when the volume changes are of great importance in the analysis process. The Dice similarity coefficient [10] (denoted as DICE in this paper) is computed as the ratio between the number of pixels belonging to the intersection (of two possible segmentations) and the average of their sizes.

Another frequently used measure for evaluating the segmentation performance is the Global Consistency Error (GCE) [21]. An error-based measure is actually an "opponent" to a similarity measure (two segmentations are identical if an error-based measure is 0). This measure is computed as an average over the error of pixels/voxels belonging to two segmentations.

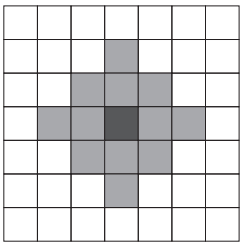
Some metrics that compute the distance between two segmented regions by taking into account the pixel location could be also considered. They quantify the dissimilarity of two segmentations and they are useful when the contours (the shapes of the boundary of the structure) are of importance for the image analysis. A distance value of 0 corresponds to a perfect match between the computed segmentation and the ground truth, while greater values indicate higher errors. In our approach, we consider the directed Hausdorff distance between two segmentations (called HDRFDST and defined as the maximal distance from a point in the first segmentation to a



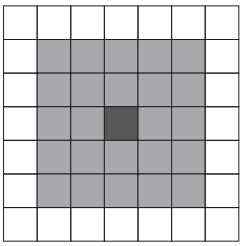
(a) von Neumann neighbourhood



(b) Moore neighbourhood

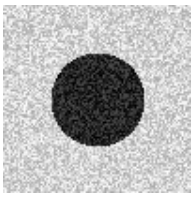


(c) extended von Neumann neighbourhood

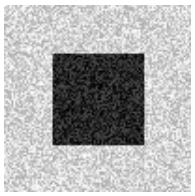


(d) extended Moore neighbourhood

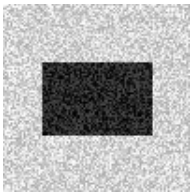
Figure 1: Different neighbourhood schemes for 2D Cellular Automata



(a) A circle



(b) A square



(c) A rectangle

Figure 2: 2D synthetic images

nearest point in the other one [15]) and the Mahalanobis Distance (MAHLNBS) [20] (that regards the correlation among all the pixels from the set that the considered two pixels are belonging).

A similarity measure able to evaluate both clusterings and classifications (because it is not based on labels) is the Rand Index (RNDIND), proposed in [28].

The similarity of two segmentations can be computed by taking into account their areas or volumes, also. In order to determine such a measure, a distance between two volumes must be defined (the similarity being 1 - this distance). A possible definition [34] for the volumetric distance is the ratio of the absolute volume difference and the sum of the compared volumes. Such a measure is useful when the segmentation purpose is to identify the changes in size since it is sensitive to miss-estimations of the segmented volume more than anything else. In the case of a single measurement, the volume error (two times the volume difference over the volume sum) can be used, while in the case of multiple measurements (when an average result is of interest) the absolute volume error can be used.

4.3 Numerical results

The aim of the first experiment is to investigate if and how the neighbourhood topology (e.g. Moore *versus* Neumann or standard *versus* extended size) influences the performance of the segmentation process. Since GrowCut algorithm requires some initial seeds (randomly given), 30 randomly generated sets of seeds have been considered for each image and for each neighbourhood type. For each set of seeds, the GrowCut algorithm has been run more iterations, until no cell/pixel changes its state. After the last iteration, the segmented image is compared to the corresponding ground truth and the evaluation measure is determined.

In Figure 4 we present the obtained results for a particular image.

In Figures 5 and 7 we present the average number of iterations (over all 30 sets of seeds) required by GrowCut algorithm to converge for several synthetic and real-world images, respectively, and for all considered neighbourhoods. The results given in Figures 6 and 8 correspond to the Dice similarity obtained in these runs.

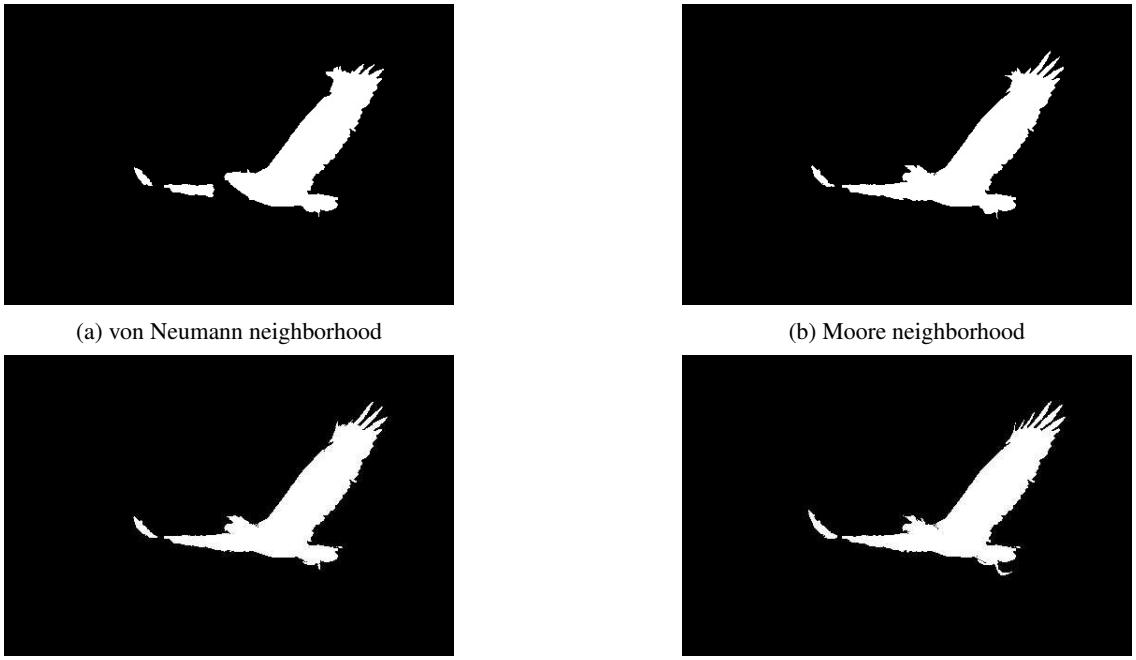
In Tables 1 and 2 we present the mean (over all images) and the corresponding confidence intervals of the results for all considered evaluation measures computed in this experiment.

Regarding the synthetic images, by taking into account the mean of iterations from Table 1, we can observe that the GrowCut is able to perform the best segmentation by using Neumann neighbourhood. If we take into account the average Dice similarity, the average GCE, the average MAHLNBS, the average RNDIND and the average VOLSMTY, both extended neighbourhoods are able to perform a perfect segmentation. These contradictory remarks could mean that GrowCut with Neumann neighbourhood becomes trapped in some local optima. Furthermore, in some cases, the extended neighbourhoods can help the algorithm improve the evaluation measure, even if they required a larger computational effort.

Regarding the real-world images, by taking into account the mean of iterations from Table 2, we can observe that the GrowCut is able to perform the best segmentation by using extended Moore neighbourhood. If we take into account all the other performance criteria, the best results are those obtained with the Moore neighbourhood.

(a) 135037 (b) 135037 GT

Figure 3: Real-world image



(a) von Neumann neighborhood (b) Moore neighborhood

(c) extended von Neumann neighborhood (d) extended Moore neighborhood

Figure 4: Results obtained for image number 135037

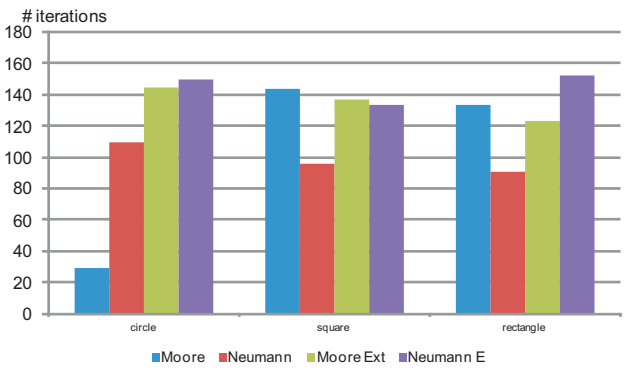


Figure 5: Number of iterations required to segment each synthetic image by considering the four neighbourhood topologies.

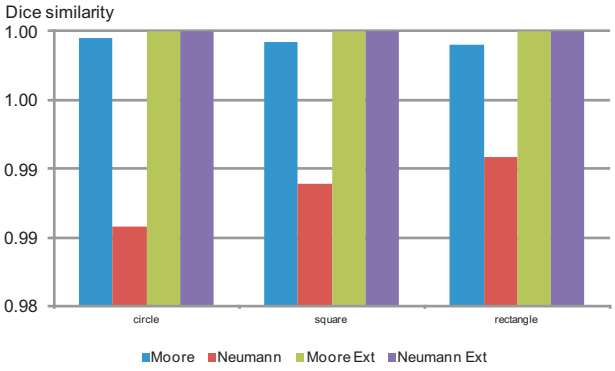


Figure 6: Dice similarity for tested synthetic images by using all considered neighbourhoods

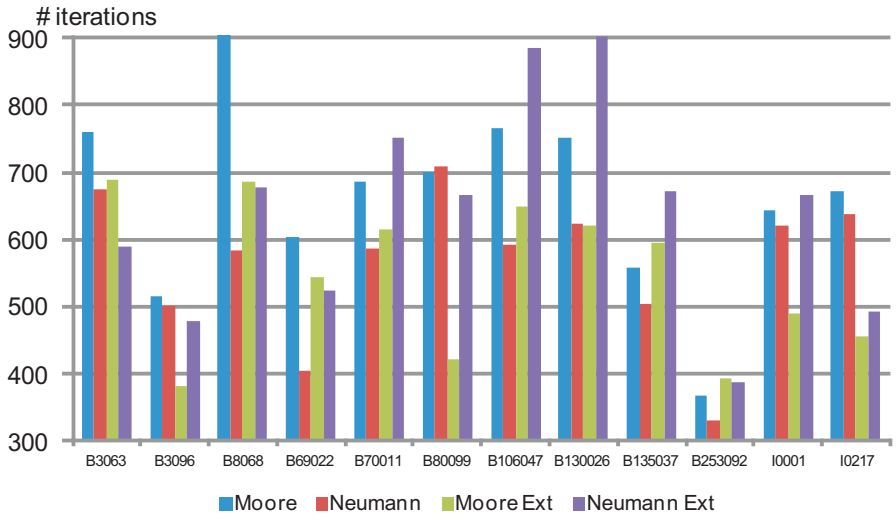


Figure 7: Number of iterations required to segment several real-world images by considering the four neighbourhood topologies.

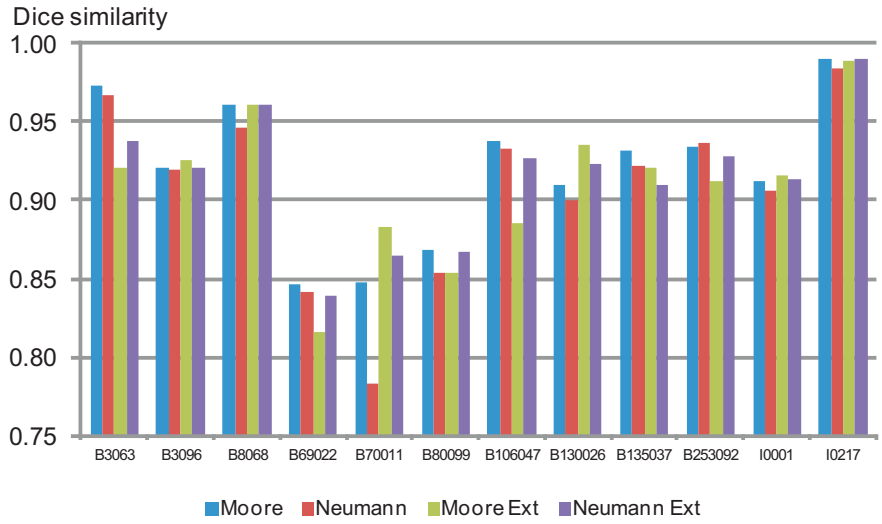


Figure 8: Dice similarity for several real-world images by using all considered neighbourhoods

	ITERATIONS	DICE	GCE	HDRFDST	MAHLNBS	RNDIND	VOLSMTY
Moore	102.150 ± 29.224	0.999 ± 0.000	0.002 ± 0.000	1.231 ± 0.161	0.001 ± 0.000	0.998 ± 0.000	0.999 ± 0.000
Neumann	**98.544** ± 4.377	0.989 ± 0.001	0.033 ± 0.003	5.566 ± 0.179	0.016 ± 0.002	0.966 ± 0.004	0.990 ± 0.001
MooreExt	134.467 ± 4.994	**1.000** ± 0.000	**0.000** ± 0.000	**0.000** ± 0.000	**0.000** ± 0.000	**1.000** ± 0.000	**1.000** ± 0.000
NeumanExt	144.911 ± 4.746	**1.000** ± 0.000	**0.000** ± 0.000	0.122 ± 0.039	**0.000** ± 0.000	**1.000** ± 0.000	**1.000** ± 0.000

Table 1: Average performances over all tested synthetic images for the considered neighbourhood topologies

	ITERATIONS	DICE	GCE	HDRFDST	MAHLNBS	RNDIND	VOLSMTY
Moore	664.958±80.372	**0.919**±0.025	**0.051**±0.026	**26.210**±8.051	**0.104**±0.040	**0.948**±0.026	**0.954**±0.020
Neumann	564.039±60.022	0.908±0.031	0.056±0.026	30.176±8.480	0.119±0.042	0.943±0.026	0.951±0.020
MooreExt	**545.300**±61.228	0.910±0.025	0.059±0.029	34.704±12.816	0.120±0.037	0.940±0.029	0.949±0.023
NeumanExt	641.228±85.445	0.915±0.023	0.054±0.025	29.759±8.984	0.114±0.033	0.946±0.026	0.951±0.021

Table 2: Average performances over all tested real-world images for the considered neighbourhood topologies

In the same time, for real-world images, if we compare the two standard neighbourhoods (Moore *versus* Neumann), Moore neighbourhood is better than Neumann in 6 cases out of 7. If we compare the two extended neighbourhoods (Moore Ext *versus* Neumann Ext), Moore Ext neighbourhood is better than Neumann Ext in only one case out of 7 (when the iteration criterion is taken into account). If we compare the standard neighbourhood with the extended one (Moore *versus* Moore Ext and Neumann *versus* Neumann Ext, respectively), we can observe that in 7 cases out of 14 the standard neighbourhood is better than the extended one and all these cases correspond to Moore neighbourhood. In the case of Neumann neighbourhood, the extended version is better than the standard one.

In a similar way we can compare neighborhoods taking into account the DICE similarity illustrated in Fig. 8. More precisely, if we compare the two standard neighborhoods, Neumann neighborhood is better only in one case out of 12 and also the difference between the two measures in this case is almost insignificant, which means that the Moore neighborhood is the best choice.

If we compare the extended neighborhoods, Moore extended is better in 5 out of 12 cases, Neumann extended is better in 6 out of 12 cases and in the remaining case they have equal values. This means that their performance depends on the characteristics of the images.

If we compare the standard neighborhoods with their extended versions, we obtain that standard Moore neighborhood is more efficient in 7 out of 12 cases and we have one case of equality. In the Neumann case, the extended version gives better results in 7 out of 12 images, which means that it is the best option between the two of them.

Obtained results indicate that an adaptation schema could be involved in the segmentation process (e.g. adapt the shape or the size of the neighbourhood), depending on some features of the images. From a statistical point of view, we cannot say which is the best neighbourhood, more experiments being planned to be performed in the future.

5 Conclusions and further work

The image segmentation problem was considered in this paper. A CA-based segmentation algorithm, named GrowCut, has been investigated by considering different neighbourhood topologies. Several real-world images and synthetic images have been used for conducting the study. The segmentation performance have been studied by using different supervised measures. Results show how the neighbourhood shape and size could influence the segmentation process.

This study will continue with the investigation of different topologies for 2D CAs for the image segmentation task, like using graph-based CAs instead of grid-based CAs. Other unsupervised performance measures will be investigated and analysed.

Acknowledgment

This work was supported by a grant of the Romanian National Authority for Scientific Research and Innovation, CNCS - UEFISCDI, project number PN-II-RU-TE-2014-4-1130. The first two authors contributed equally to this work.

REFERENCES

[1] Andreica, A., Chira, C., Using a Hybrid Cellular Automata Topology and Neighborhood in Rule Discovery, Proceedings of the 8th International Workshop on Hybrid Artificial Intelligence Systems (HAIS 2013), Salamanca, Spain, Lecture Notes in Artificial Intelligence, Springer-Verlag, vol. 8073, p. 669-678 (2013)

[2] Andreica, A., Chira, C., Weighted Majority Rule for Hybrid Cellular Automata Topology and Neighborhood, Studia Universitatis Babe-Bolyai, Informatica series, Vol. LVIII, No. 2, p. 65-76 (2013)

[3] Andreica, A., Chira, C., Evolution and Dynamics of Node-Weighted Networks for Cellular Automata Computation, Logic Journal of the IGPL (2014)

[4] Bak, P., Tang, C., Wiesenfeld, K., Self-organized criticality: An explanation of 1/f noise, Phys. Rev. Lett., vol. 59, pp. 381-384, 1987.

[5] Beasley RA, Semiautonomous medical image segmentation using seeded cellular automaton plus edge detector, ISRN Signal Processing, pp. 1-9 (2012)

[6] Bi L, Kim J, Cellular Automata and Anisotropic Diffusion Filter based Interactive Tumor Segmentation for Positron Emission Tomography, Proc IEEE Eng Med Biol Soc. (2013)

[7] Chira, C., Gog, A., Lung, R. I., Iclanzan, D., Complex Systems and Cellular Automata Models in the Study of Complexity, Studia Informatica series, Vol. LV, No. 4, pp. 33-49 (2010)

[8] Darabos, C., Giacobini, M., Tomassini, M., Performance and Robustness of Cellular Automata Computation on Irregular Networks. Advances in Complex Systems 10, p. 85-110 (2007)

[9] Darabos, C., Tomassini, M., Di Cunto, F., Provero, P., Moore, J.H., Giacobini, M., Toward robust network based complex systems: from evolutionary cellular automata to biological models, Intelligenza Artificiale 5(1), p. 37-47 (2011)

[10] Dice, L.R., Measures of the Amount of Ecologic Association Between Species, Ecology 26(3), 297-302 (1945)

[11] Ghosh P, Antani SK, Long LR, Unsupervised Grow-Cut: cellular automata-based medical image segmentation. Healthcare Informatics, Imaging and Systems Biology (HISB), 2011 First IEEE International Conference on IEEE, pp. 40-47 (2011)

[12] Gog, A., Chira, C., Dynamics of Networks Evolved for Cellular Automata Computation, Proceedings of the 7th International Workshop on Hybrid Artificial Intelligence Systems (HAIS 2012), Salamanca, Spain, Hybrid Artificial Intelligent Systems, Springer-Verlag, vol. 7208-7209, p. 359-368 (2012)

[13] Greenberg J. M., Hastings, S. P., Spatial patterns for discrete models of diffusion in excitable media, SIAM Journal on Applied Mathematics, vol. 34, pp. 515-523, 1978.

[14] Hamamci A, Kucuk N, Karaman K, Engin K, Unal G, Tumor-Cut: Segmentation of Brain Tumors on Contrast Enhanced MR Images for Radiosurgery Applications, IEEE Transactions on Medical Imaging, vol. 31, issue 3, pp. 790-804 (2012)

[15] Hausdorff, F. (1914). Grundzüge der Mengenlehre, Chelsea, New York, Reprinted by Chelsea, 1949.

[16] Juille, H., Pollack, J.B.: Coevolving the 'ideal' trainer: Application to the discovery of cellular automata rules. Genetic Programming 1998: Proceedings of the Third Annual Conference (1998)

[17] Kauffmann C., Piche N., Seeded ND medical image segmentation by cellular automaton on GPU, International Journal of Computer Assisted Radiology and Surgery, vol. 5, no. 3, pp. 251-262 (2010)

[18] Korte, A.C.J., Brouwers, H.J.H., A cellular automata approach to chemical reactions; 1 Reaction controlled systems, Chemical Engineering Journal, 228, 172 (2013)

[19] Liu Y, Cheng HG, Huang J, Zhang Y, Tang X, An Effective Approach of Lesion Segmentation Within the Breast Ultrasound Image Based on the Cellular Automata Principle, J. Digital Imaging, vol. 25, pp. 580-590 (2012)

[20] Mahalanobis, P. C., On the generalised distance in statistics, In *Proc. of the Nat. Inst. of Sci. of India*, pages 49–55, Published as Proc. of the Nat. Inst. of Sci. of India, volume 2, number 1 (1936)

[21] Martin, D., Fowlkes, C., Tal, D., Malik, J., A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics, Proceedings of the 8th IEEE International Conference on Computer Vision (ICCV '01), vol. 2, pp. 416-423 (2001)

[22] Mitchell, M., Crutchfield, J.P., Das, R., Evolving cellular automata with genetic algorithms: A review of recent work, In Proceedings of the First International Conference on Evolutionary Computation and Its Applications (EvCA'96). Russian Academy of Sciences (1996)

[23] Mitchell, M., Thomure, M. D., Williams, N. L.: The role of space in the Success of Coevolutionary Learning. Proceedings of ALIFE X - The Tenth International Conference on the Simulation and Synthesis of Living Systems (2006)

[24] Mofrad, M. H. , S. Sadeghi, A. Rezvanian, and M. R. Meybodi, Cellu-

lar edge detection: Combining cellular automata and cellular learning automata, fAEUg - International Journal of Electronics and Communications, vol. 69, no. 9, pp. 1282-1290, 2015.
[25] Oliveira, G.M.B., Martins, L.G.A., de Carvalho, L.B., Fynn, E., Some investigations about synchronization and density classification tasks in one-dimensional and two-dimensional cellular automata rule spaces, Electron. Notes Theor. Comput. Sci., 252 (2009), pp. 121-142.
[26] Peitgen, H.-O., H. Jurgens, and D. Saupe, Chaos and fractals - new frontiers of science. Springer, 1992.
[27] RajKumar RS, Niranjana G, Image Segmentation and Classification of MRI Brain Tumor Based on Cellular Automata and Neural Networks, International Journal of Research in Engineering and Advanced Technology, vol 1, issue 1 (2013)
[28] Rand, W. M., Objective criteria for the evaluation of clustering methods, American Statistical Association Journal, 66(336):846–850 (1971)
[29] Rosin, Training cellular automata for image processing, in SCIA, 2005, pp. 195-204. [Online]. Available: http://dx.doi.org/10.1007/11499145 22
[30] Rosin,P. L., Image processing using 3-state cellular automata, Computer Vision and Image Understanding, vol. 114, no. 7, pp. 790-802, 2010.
[31] Safia, D., Oussama, D., Chawki, B., Image segmentation using continuous cellular automata, in Programming and Systems (ISPS), 2011 10th International Symposium on, April 2011, pp. 94-99.
[32] Safia D., Chawki, B., Image segmentation using an emergent complex system: Cellular automata, in Systems, Signal Processing and their Applications (WOSSPA), 2011 7th International Workshop on, May 2011, pp. 207-210.
[33] Santner J., Pock T.,Bischof H., Interactive Multi-Label Segmentation, Proceedings 10th Asian Conference on Computer Vision (ACCV), Queenstown, New Zealand, 2010.
[34] Taha, A. A. and Hanbury, A., Metrics for evaluating 3D medical image segmentation: analysis, selection, and tool, BMC Medical Imaging, 15:29 (2015).
[35] Tomassini, M., Venzi, M.: Evolution of Asynchronous Cellular Automata for the Density Task. Parallel Problem Solving from Nature - PPSN VII, Lecture Notes in Computer Science, Springer Berlin / Heidelberg, 2439, 934-943 (2002)
[36] Tomassini, M., Giacobini, M., Darabos, C., Evolution and dynamics of small-world cellular automata, Complex Systems 15, p. 261-284 (2005)
[37] Vezhnevets V., Konushin V, "GrowCut" - Interactive Multi-Label N-D Image Segmentation by Cellular Automata, Proc. of GraphiCon, pp. 150-156 (2005)
[38] Watts, D.J., Strogatz, S.H., Collective dynamics of small-world networks. Nature, Vol. 393 No. 6684, p. 440-442 (1998)
[39] Watts, D.J., Small Worlds: The Dynamics of Networks Between Order and Randomness, Princeton University Press, Princeton, NJ (1999)
[40] Wolfram, S., Theory and Applications of Cellular Automata, Advanced series on complex systems, World Scientific Publishing, 9128 (1986)

Generic Convolutional Neural Network with Random Pooling Area

Zhidong Deng [1] and **Zhenyang Wang** [2] and **Shiyao Wang** [3]

Abstract. As an automated hierarchical feature extractor, deep convolutional neural network (CNN) is increasingly in the spotlight. In order to further improve feature representation capabilities of CNN, this paper proposes a novel SAPNet model that incorporates a stochastic area pooling (SAP) method with a generic stacked T-shaped CNN architecture. In our SAP method, pooling area is randomly transformed and max pooling operation is then conducted on such areas, which means that regular pooling area of fixed upright squares are no longer exploited in the training phase of our SAPNet. In a sense, it could be viewed as the use of feature-level augmentation. Meanwhile, we present a generic CNN architecture that structurally resembles three stacked T-shaped cubes. In such architecture, the number of kernels in convolutional layer preceding any pooling layer is doubled and all learnable weight layers are combined with batch normalization and dropout with a small ratio. Interestingly, all SAPNets have the same structures and similar parameter settings on different benchmarks. Finally, on CIFAR-10, CIFAR-100, MNIST, and SVHN datasets, the experimental results show that our SAPNet requires fewer parameters than regular CNN models but nevertheless achieves superior recognition performances for all the four benchmarks.

1 Introduction

A major problem for pattern recognition is how to represent features. In the last decade, algorithmic level features such as SIFT, HOG, and LBP have been demonstrated to make success in a variety of object recognition applications. But traditional pattern recognition methods are essentially based on manually selected features, which seems to be getting dim compared to a sequence of impressive results achieved by deep convolutional neural network (CNN) in many areas, particularly in computer vision and speech recognition, since 2012 [13].

Unbelievable recognition capabilities of CNN could be attributed to automated hierarchical or multiple granularity feature extraction. It functionally has resemblance to biological viusal cortex pathaway. In recent years, extensive prominent CNN models have been emerging, although there still exist many unresolved problems. How to enhance feature representation capabilities of CNN and how to design an generic but effective architecture increasingly become a research hot topic.

Data augmentation is a simple but useful way to promptly improve recognition and generalization capabilities of CNN. It can help expand data space and further enhance diversity of input images. In this paper, we take advantage of such idea to investigate expansion on feature map space. Accordingly, we propose a novel stochastic area pooling (SAP) to better feature representation of CNN. Instead of regular pooling areas of fixed upright squares, new pooling areas in our SAP method could be randomly translated, scaled, and rotated with a tiny fluctuation. Jaderberg *et al*. [10] introduce a spatial transformer network that input feature map is transformed. The difference is that the transformations in spatial transformer network are a deterministic function of the input and the transformation parameters and are explicitly optimized over, while the transformations in our SAP method are randomly drawn from pre-specified distributions. In addition, Jaderberg *et al*. aim at generating scale and rotation invariance features by introducing an addition learning-based spatial transformer network, which requires a large amount of parameters and computational cost. SAP, however, plays the role of feature-level augmentation, bearing only a little extra computational burden.

Meanwhile, in order to simplify design of new convolutional models, we present a generic CNN architecture that structurally looks like three stacked T-shaped cubes. In such architecture, the number of kernels in convolutional layer just preceding any pooling layer is doubled to overcome representational bottleneck [32]. All learnable weight layers are embedded with regularization of batch normalization, ReLU, and dropout with a small ratio of 0.1. As a result, all SAPNets with different number of convolutional kernels in the first convolutional layer have similar structures and parameter settings, which are demonstrated to be stable, reliable, and efficient. In addition, we found out that the size of receptive field in the last convolutional layer directly determines depth of CNN networks. For example, on CIFAR-10, CIFAR-100, MNIST, and SVHN datasets, it is sufficient to choose networks with 9-12 layers.

Finally, on CIFAR-10, CIFAR-100, MNIST, and SVHN benchmarks, the experimental results show that our SAPNet requires fewer parameters than regular CNN models but achieves excellent recognition performances. Specifically, our SAPNet-64 yields a state-of-the-art test error of 5.57% on CIFAR-10. On CIFAR-100, the SAPNet-64 achieves a test error of 27.59%. On MNIST, SAPNet-32 with 0.76 million parameters obtains a test error of 0.29%, which breaks record on MNIST if there is no any data augmentation applied. On SVHN, SAPNet-64 achieve a result of 1.71%, ranking the second place.

The paper is organized as follows: In Section 2, the related work is reviewed. Section 3 proposes our SAP method for a generic CNN ar-

[1] State Key Laboratory of Intelligent Technology and Systems, Tsinghua National Laboratory for Information Science and Technology, Department of Computer Science, Tsinghua University, Beijing 100084, China, email: michael@tsinghua.edu.cn

[2] State Key Laboratory of Intelligent Technology and Systems, Tsinghua National Laboratory for Information Science and Technology, Department of Computer Science, Tsinghua University, Beijing 100084, China, email: crazycry2010@gmail.com

[3] State Key Laboratory of Intelligent Technology and Systems, Tsinghua National Laboratory for Information Science and Technology, Department of Computer Science, Tsinghua University, Beijing 100084, China, email: sywang14@mails.tsinghua.edu.cn

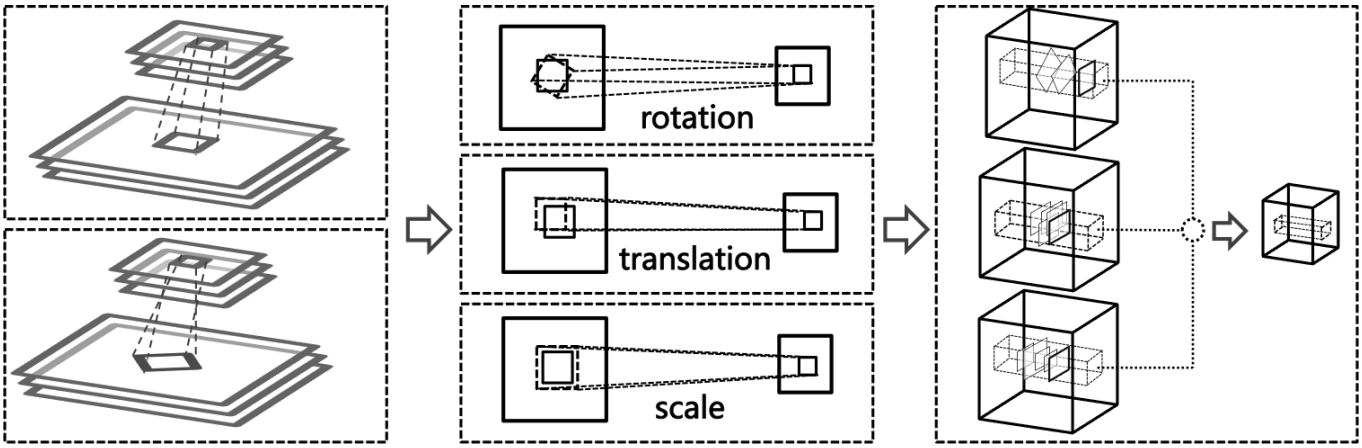


Figure 1. Schematic diagram of stochastic area pooling

chitecture. Section 4 gives details on training and test of the SAPNet and provides our experimental results. Finally, Section 5 concludes this paper with a brief summary.

2 Related Work

Cognitron [3] and Neocognitron [4] are generally viewed as the earliest CNN, where a few widely-used concepts in CNN such as convolution, pooling, receptive field, and ReLU are initially presented. In 1989, error backpropagation was first introduced to Neocognitron by LeCun *et al*. [14]. Since 2012, significant progress in very large scale object classification task with CNN has been made due to very efficient GPU implementation of convolution operation and regularization for preventing overfitting [13]. From the outset of this, attention from both academic and industrial communities has been paid to such deep CNN.

In the pipeline of CNN, pooling is a standard module, which could help extract translation invariance features in terms of contiguous pooling areas and stable pooling operation. In addition to regular deterministic pooling methods, such as max pooling and average pooling operations, stochastic pooling is presented by Zeiler and Fergus [34] through randomly picking activation within each pooling area according to a multinomial distribution. This pooling is mainly used to prevent over-fitting when training deep convolutional networks. Lee *et al*. [16] propose a mixed/gate max-average pooling, which improves pooling by pursuing a learnable pooling function to adapt to complex and variable patterns. Furthermore, stochastic unpooling is presented by Pu *et al*. [22] to link consecutive layers in deep generative models.

On visual object recognition datasets including CIFAR-10, CIFAR-100, MNIST, and SVHN, there are also researchers working on applications of variants of CNNs. For instance, Maxout is presented to improve model averaging techniques [6]. Network In Network (NIN) proposed by [19] is used to enhance local discriminability of model, where global average pooling simplifies classification and prevents overfitting. Wan *et al*. [33] introduce DropConnect as generalization of Dropout for regularizing fully-connected layers. DasNet presented by [30] seems like powerful by allowing it to focus its internal attention on some of convolutional filters. Lee *et al*. [17] propose a deeply-supervised nets (DSN) and simultaneously minimizes classification errors of hidden layers. Lately, a recurrent CNN (RCNN) is presented by incorporating recurrent connections into each convolutional layer [18], in order to integrate the context information. Srivastava *et al*. [29] propose a novel highway network that allows information flow across many layers on information highway. This work makes it possible to train a extreme deep model. In the same year, a residual learning framework known as ResNet is designed to ease training [8], which achieves great success on ILSVRC 2015, CIFAR-10, and COCO datasets.

3 SAPNet Method

In the following, a novel stochastic area pooling (SAP) method and a generic CNN architecture with three stacked T-shaped cubes are proposed and discussed.

3.1 SAP Layer

SAP consists of two consecutive steps: area selection and pooling operation. As the first step, a pooling area is generated using random affine transformation. Second, regular pooling operations such as max or average pooling are conducted on such randomly transformed areas, thus giving rise to output of SAP layer.

Instead of regular pooling on fixed square, the random pooling area of SAP is transformed by an affine mapping: $f : A \rightarrow B$, where $A \in \mathcal{R}^n$ and $B \in \mathcal{R}^m$ stand for affine spaces. Assume $x \in A$ indicates an n-dimensional vector. The affine transformation can be represented as $f(x) = Tx + b$, where $T \in \mathcal{R}^{m \times n}$ denotes an affine transformation matrix and $b \in B$ a translation vector. In 2D case, we only consider shift, rotate, and scale of four corners of a pooling area that slides over feature maps, as shown in Figure 1.

Suppose that θ denotes angle of rotation, $[c_x, c_y]$ coordinates of a center point, $[o_x, o_y]$ a shift vector, and $[s_x, s_y]$ scaling factors. We form a 7-tuple of parameters $\{\theta, c_x, c_y, o_x, o_y, s_x, s_y\}$ so as to express such an affine transformation, i.e.,

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s_x \cos\theta & s_y \sin\theta & t_x \\ -s_y \sin\theta & s_x \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \tag{1}$$

$$t_x = c_x(1 - s_x \cos\theta) - c_y s_y \sin\theta + o_x \tag{2}$$

$$t_y = c_x(1 + s_y \sin\theta) - c_y s_x \cos\theta + o_y \tag{3}$$

Actually, pooling area in SAP is produced by assigning a random distribution to each parameter. The Gaussian distribution that each parameter must satisfy is given below,

$$\theta \sim N(0, \sigma_\theta), \tag{4}$$

$$c_x, c_y \sim N(0.5d, \sigma_c) \tag{5}$$

$$s_x, s_y \sim N(1, \sigma_s) \tag{6}$$

where standard deviations σ_θ, σ_c and σ_s are called hyper-parameters, which should be pre-specified before training, and d indicates the height or width of incoming feature map.

Note that pixels within random pooling areas are usually transformed onto non-integral boundaries. Thus bilinear interpolation is required before pooling operation.

During every forward pass, a set of different affine transformation parameters are randomly generated produced via Equations (4), (5), and (6). After that, these transformation parameters or the resulting pooling areas in the SAP layer remain unchanged and are not optimized over during either forward pass or back-propagation. Sequentially, max pooling is then done on such areas. Apparently, our SAP layer has the same forward and backward pass as that for regular max pooling except for transformed pooling area.

In the test phase, we set the standard deviations of parameters $\sigma_\theta = 0$, $\sigma_c = 0$, and $\sigma_s = 0$ for the SAP layers, which means that we employ the same fixed upright pooling squares as that of regular pooling method.

3.2 Stacked T-shaped Cubic Architecture

In addition to pooling strategies, design of CNN architecture is also crucial to improvements in capabilities of hierarchical feature representation. Basically, this used to be an empirical trick. For example, VGG [23] and GoogLeNet [31] suggest that deeper network architecture, smaller receptive field, and finer stride can help improve recognition accuracy. But some of lately published experimental results show that accuracy become stagnant or even degradation as depth going deeper [8]. In the study of [32], it is demonstrated that model design should avoid representational bottleneck problems originated from [32], especially early in network. In fact, how to find an appropriate value of depth, how to configurate convolutional and pooling layers, how to select the size and number of convolutional kernels /pooling areas, and even how to prevent overfitting for specific problem are open problem.

Table 1. The architecture of SAPNet.

SAPNet-16	SAPNet-32	SAPNet-48	SAPNet-64
conv3-16	conv3-32	conv3-48	conv3-64
conv3-32	conv3-64	conv3-96	conv3-128
Stochastic area pooling (SAP) layer			
conv3-32	conv3-64	conv3-96	conv3-128
conv3-32	conv3-64	conv3-96	conv3-128
conv3-64	conv3-128	conv3-192	conv3-256
Stochastic area pooling (SAP) layer			
conv3-64	conv3-128	conv3-192	conv3-256
conv3-64	conv3-128	conv3-192	conv3-256
conv3-128	conv3-256	conv3-384	conv3-512
Global average pooling layer			
Softmax layer			

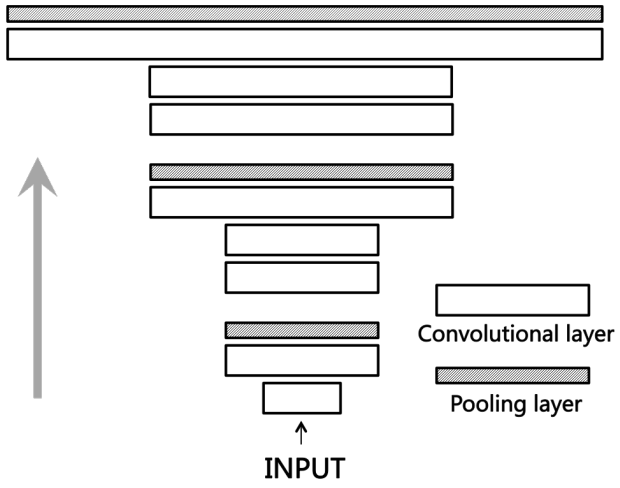


Figure 2. Generic CNN architecture with three stacked T-shaped cubes.

In this paper, we deliberately design a generic CNN architecture that structurally looks like three stacked T-shaped cubes (shown in Figure 2), together with two SAP layers as separator. Note that the number of kernels in convolutional layer preceding each pooling layer is always doubly expanded. We call such a general model as SAPNet. The SAPNet architecture with different kernel sizes is listed in Table 1. It is clearly observed that all SAPNets have the same structures and similar parameter settings. Specifically, every SAPNet comprises three T-shaped ensembles and one softmax layer, each of ensembles containing three convolutional layers. A total of 9 convolutional layers are separated by two SAP layers and single global average pooling layer.

In SAPNet, all convolutional layers make use of small receptive fields of 3×3 with padding of 1 so as to preserve spatial resolution. In two SAP layers, we take 2×2 initial pooling areas with stride of 2. Following the last convolutional layer, single global average pooling is adopted. Similarly, we exploit a softmax layer as the final output. Additionally, all learnable weight layers are embedded with regularization of batch normalization (BN) [9], ReLU nonlinearity, and dropout with a small ratio of 0.1. The BN adopted in this paper is slightly different from [9]. Our modified BN is only for normalization without any scale and shift operations.

In SAPNet, the depth of network is always set to be 9, although it is not the best one (also see Table 2). Apparently, comparison given in Table 2 is unfair because deeper models need more computation and parameters. It is clear to see from Table 2 that models with 9-12 layers are sufficient for such a task, among which 9-layer model has the least computation and parameters. In fact, the selection of model depth depends on the size of receptive field in a sense. Specifically, the size of receptive field in the last convolutional layer, which is gradually enlarged as network going deeper, directly determines depth of CNN networks. As the last convolutional layer's receptive field reaches as about 1.5-2.0 times large as raw input image size, recognition accuracy is expected to come to peak. A continuously increase of depth seems to be unfavorable to improvements in performance. For CIFAR-10, CIFAR-100, MNIST, and SVHN datasets, we could choose networks with 9-12 layers as a compromise solution.

Table 2. The performance of generic CNN network architecture with different depths. 'Architecture' row specifies the structure of each CNN network. For instance, '2331' means that this CNN network is composed of two convolutional layers, one max pooling layer, three convolutional layers, one max pooling layer, three convolutional layers, one global average pooling layer, and one softmax layer.

#layers	7	8	9	10	11	12	13	14	15
Architecture	2221	2321	2331	2431	2441	2541	2551	2651	2661
Test error (%)	10.09	8.77	8.53	8.35	7.93	7.65	7.86	7.63	7.77
Receptive field	32	36	44	48	56	60	68	72	80
#parameter	0.57 M	0.61 M	0.76 M	0.80M	0.94M	0.98M	1.13 M	1.16 M	1.31M

In order to resolve representational bottleneck problems [32], a simple but effective way is to double the number of internal outputs or convolutional kernels. The size of the representation is defined as the product of the number of channels and the resolution of each feature map. In general, it should gently decrease from the inputs to the last convolutional layer, which means that one should avoid representational bottlenecks or the loss of features with extreme or sharp compression. Otherwise, it is probably unable to sufficiently represent raw input images. Considering that pooling operations actually reduce resolution of feature maps but make no change to the number of channels, it is likely to lead to representational bottlenecks. For the sake of this, it is necessary to double the number of channels prior to pooling.

Additionally, we need to maintain computational balance for all layers in SAPNet. In a convolutional layer, both the size of incoming feature maps and the number of kernels jointly determine computational cost of this layer. Owing to the fact that the size of incoming feature maps is shrunk through pooling, we should accordingly raise the number of kernels in such convolutional layer so as to keep computational balance for each layer. Moreover, we stack three convolutional layers to form an ensemble for further balancing computational cost.

3.3 Discussions

In this paper, we propose a high-performance SAPNet that combines our SAP method with a generic CNN architecture.

In the SAP method, the pooling areas, instead of the transformed input feature maps in the spatial transformer network [10], are generated using random affine transformation and not optimized over. On such randomly shifted, rotated, and scaled areas, we conduct max pooling operation in the way like that in regular pooling. Essentially, the SAP plays the role of feature-level augmentation rather than data augmentation. Considering that the pooling areas fluctuate in a small range of angle, shift, and scale, it allows significant random image transformation with a little extra computational burden. Although SAP itself is unable to bring either any scale invariance or any rotation invariance, it can provide extensive multi-scale and multi-orientation features to next convolutional layers. In terms of performing max pooling operations merely on randomly transformed areas instead of regular fixed squares, there is almost no difference in efficiency as compared to regular CNNs.

On the other hand, we design a generic CNN architecture with three stacked T-shaped cubes (Figure 2), which can balance computational burden and avoid representational bottlenecks. We use dropout with a small ratio of 0.1 after each convolutional layer. It is quite different from the others, where dropout is often employed only for fully-connected layers or the last few convolutional layers. We do believe that the use of dropout with a small ratio and modified BN are capable of avoiding overfitting effectively. Interestingly, with the same BN and dropout, identical CNN architecure, e.g., SAPNet-32, is able to have superior performances even if applied on different datasets. It seems to be generic, stable, and efficient.

The experimental results given in Table 2 also illustrate that the selection of model depth depends on the size of receptive field. We could provide an inexact explanation below. If we would ignore nonlinearity and normalization, the entire CNN only contains convolutional and pooling operations. We could multiply all convolutional kernels to get a total of kernel until receptive field is enlarged to larger than visual object to be classified. If one continues to increase the depth or multiply more kernels, it is helpless to further improve recognition accuracy.

4 Experimental Results

4.1 Overall Parameter Settings

SAPNet is implemented based on the framework of caffe [11]. All the experiments are conducted on two GPUs with data parallelism. We test our SAPNets on four benchmark datasets: CIFAR-10, CIFAR-100, MNIST, and SVHN. For each dataset, three networks with different parameters are tested for comparison. As an exceptional case, we accomplish more experiments on CIFAR-10 so as to validate effectiveness and efficiency of our method.

In all the experiments, we train our model by stochastic gradient descent (SGD) algorithm with batch size of 96. The initial learning rate is set to 0.01 and is dropped by a constant factor of 10 whenever the loss begins to reach an apparent plateau. We repeatedly decrease the learning rate 3 times, until it arrives at 1e-5. A momentum of 0.9 is adopted during the entire training process to make SGD stable and fast. All the learnable parameters exploit the same weight decay of 0.004. Following each convolutional layer, dropout with a small ratio of 0.1 is performed. We set the standard deviations of parameters $\sigma_\theta = 5^\circ$, $\sigma_c = 0.25d$ and $\sigma_s = 0.01$ for the SAPs. The maximum fluctuations may take 2-3 times of them.For all the datasets, image samples are pre-processed only with removing pre-pixel means.

4.2 CIFAR-10

We begin our experiments with CIFAR-10 [12] and pay more attention on it, in order to verify our method. CIFAR-10 dataset consists of 60,000 32x32 color images with 10 classes. The dataset is divided into five training batches and one test batch, each with 10,000 images.

Table 3. Comparison with other pooling methods on CIFAR-10.

Model	#parameter	Test error (%)
Max pooling	0.76 M	8.53
Average pooling	0.76 M	8.13
Stochastic pooling	0.76 M	8.50
SAPNet-32	0.76 M	**7.56**

As shown in Table 3, we first compare the SAP with the other pooling methods. In the experiments, we exploit the same three stacked T-shaped network architecture as SAPNet-32 for the sake of comparison. Surprisingly, all the test errors in Table 3 surpass the existing state of the art results listed in Table 5. It should be attributed to generic CNN architecture deliberately designed in Section 3.2. In fact, the generic CNN architecture is able to avoid representational bottleneck. Specifically, an appropriate depth like 9 layers makes the network generalize well. In addition, using the dropout layer with a small ratio just after each convolutional layer is also critical. It is demonstrated to be capable of preventing overfitting effectively, especially when implementing it with BN regularization.

From Table 3, it is easy to see that SAP achieves superior performance when using the SAP in place of regular pooling in standard CNN architectures. The reason is that the use of random pooling areas can be viewed as feature-level augmentation. It may perturb feature extractions and thus enhances the generalization ability of CNN. This is also the motivation of constructing stochastic pooling area.

Table 4. Comparison with max pooling for regular fixed square on CIFAR-10.

Model	#params	Test error (%)
MaxNet-32	0.76 M	8.53
MaxNet-48	1.71 M	7.69
MaxNet-64	3.03 M	7.01
SAPNet-32	0.76 M	**7.56**
SAPNet-48	1.71 M	7.02
SAPNet-64	3.03 M	6.61

To give an intuitive insight into performance improvements, we investigate several models with different parameter sizes, as presented in Table 3. Apparently, SAPNet performs better than MaxNet in terms of the same parameter size. One interesting thing is that SAPNet-32 performs as excellent as MaxNet-48, while SAPNet-48 is able to rival MaxNet-64. On the basis of SAPNet, a small model with fewer parameters may result in nearly the same performance as a big model. Actually, SAPNet helps save about half of parameters without any obvious loss of accuracy. In other words, SAPNet exploits much smaller models to reach similar accuracy, because it significantly improves generalization ability through feature-level augmentation.

We then make comparison of SAPNet with the state of the art models on CIFAR-10 (Table 5). Three SAPNet models with different parameter sizes are tested. All of the three SAPNets outperform the existing models. Even for SAPNet-32 with the smallest size, it only has 0.76 million parameters and still exceeds the existing state of the art Tree+Max-Avg [16]. SAPNet-64 breaks record once again and achieves a test error of 6.36% on CIFAR-10 without any data augmentation. Note that SAPNet-64 is repeatedly tested 5 times so as to ensure that the result is reliable.

In order to keep consistent with the previous work, we also test our SAPNet models on CIFAR-10 using data augmentation of translation and horizontal flipping. We randomly crop a portion of 24x24 pixels from original images and flip it horizontal randomly. Five 24x24 crops from four corners and one center w.r.t.their horizontal flipping crops are tested. The final test result is given using average of all the ten crops's outputs. Using nearly the same parameters, SAPNet-48 is superior to Tree+Max-Avg [16]. Most importantly, SAPNet-64 yields a test error of 5.57%. The best result on CIFAR-10 is obtained by Fractional MP [7]. But it employs extreme data augmentation, including randomized mix of translation, rotation, reflection, stretching, and shearing operations.

Table 5. Comparison with existing models on CIFAR-10 with and without data augmentation.

Model	#parameter	Test error (%)
Without data augmentation		
Maxout [6]	>5 M	11.68
Prob maxout [27]	>5 M	11.35
DasNet [30]	>5 M	9.22
NIN [19]	0.97 M	10.41
DSN [17]	0.97 M	9.69
RCNN [18]	1.86 M	8.69
ALL-CNN [26]	1.3 M	9.08
Tree+Max-Avg [16]	1.85M	**7.62**
SAPNet-32	0.76 M	**7.56**
SAPNet-48	1.71 M	**7.02**
SAPNet-64	3.03 M	**6.36(6.50±0.14)**
With data augmentation		
Maxout [6]	>5 M	9.38
Prob maxout [27]	>5 M	9.39
dasNet [30]	>5 M	9.22
DropConnect [33]	5 networks	9.32
NIN [19]	0.97 M	8.81
DSN [17]	0.97 M	7.97
RCNN [18]	1.86 M	7.09
Highway network [29]	2.3 M	7.54(7.72±0.16)
ALL-CNN [27]	1.3 M	7.25
ResNet [8]	1.7M	6.43(6.61±0.16)
Fitnet4-LSUV [20]	2.5M	6.06
Tree+Max-Avg [16]	1.85M	**6.05**
Tuned CNN [24]	1.29M	6.37
SAPNet-32	0.76 M	6.77
SAPNet-48	1.71 M	**5.92**
SAPNet-64	3.03 M	**5.57(5.66±0.09)**
Extreme data augmentation		
Large ALL-CNN [25]	-	4.41
Fractional MP [7]	-	**3.47**

4.3 CIFAR-100

CIFAR-100 [12] is just like CIFAR-10, excluding 100 classes involved. They have the same sizes of training and test datasets as CIFAR-10. In this case, each class in CIFAR-100 only contains 1/10 samples compared to CIFAR-10. Using the same parameter settings as that on CIFAR-10, we test three SAPNet models on CIFAR-100 without any data augmentation, as listed in Table 6. It is readily observed that SAPNet-64, which achieve a test error of 27.59%, is merely inferior to ELU-Network [2] if there is no any data augmentation applied, ranking the second place. Note that SAPNet-64 only exploits 8% of parameters compared to ELU-Network.

4.4 MNIST

MNIST is a handwritten digits dataset with dights from 0 to 9 [15]. It contains a training dataset of 60,000 samples and a test dataset of 10,000 ones. The digits have been centered and size-normalized to 28x28 grayscale images. MNIST is much simpler compared to CIFAR-10. As a result, even small models with fewer parameters can also work well. On such a benchmark, we test two small models of SAPNet-16 and SAPNet-32 without data augmentation. All parameter settings keep the same as CIFAR-10. The comparison with other existing CNN models is listed in Table 7. With 0.19 million parameters, SAPNet-16 obtains the same test error as RCNN-96 that requires 0.67 million parameters [18]. Furthermore, SAPNet-32 with 0.76 million parameters achieves a test error of 0.29%, which breaks

Table 6. Comparison with existing models on CIFAR-100.

Model	#parameter	Test error (%)
Without data augmentation		
Maxout [6]	>5 M	38.57
Prob maxout [27]	>5 M	38.14
DasNet [30]	>5 M	33.78
Tree based priors [28]	-	36.85
NIN [19]	0.98 M	35.68
DSN [17]	0.98 M	34.57
RCNN [18]	1.87 M	31.75
ALL-CNN [27]	1.30 M	33.71
Tree+Max-Avg [16]	1.76 M	32.37
ELU-Network [2]	39.32 M	**24.28**
SAPNet-32	0.76 M	32.22
SAPNet-48	1.71 M	29.32
SAPNet-64	3.03 M	**27.59**
With data augmentation		
Highway Network [29]	2.30 M	32.24
Fitnet4-LSUV [20]	2.5 M	27.66
Tuned CNN [24]	1.29 M	27.40
Extreme data augmentation		
Fractional MP [7]	-	26.39

record on MNIST if not considering any data augmentation. In general, a test error of 0.21% on MNIST is regarded as the best result [33]. But it apparently attributes to complex data augmentation and model averaging of up to 5 nets. Actually, performance of single network in DropConnect is worse than 0.29%, which is a test error yielded using our SAPNet-32.

Table 7. Comparison with existing models on MNIST.

Model	#parameter	Test error (%)
Without data augmentation		
Maxout [6]	0.42 M	0.45
NIN [19]	0.35 M	0.47
DSN [17]	0.35 M	0.39
RCNN [18]	0.67 M	**0.31**
Tree+Max-Avg [16]	1.85 M	**0.31**
FitNet-LSUV-SVM [20]	0.03 M	0.38
SAPNet-16	0.19 M	0.31
SAPNet-32	0.76 M	**0.29**
With data augmentation		
DropConnect [33]	5 networks	0.21
MCDNN [1]	35 networks	0.23

4.5 SVHN

SVHN [21] is a real-world image dataset, which is acquired from house numbers in Google street view images. Among two formats provided by such a dataset, we only use the second format. In fact, SVHN is much more difficult than MNIST, since multiple digits may visibly occur within each image. It contains 630,420 color images of size 32x32, divided into 73,257 images for training, 26,032 digits for testing, and an extra 531,131 additional somewhat less difficult samples for training. When multiple digits exist in one image, we only need to recognize the center one.

We follow training and test procedures given by [6]. In the experiment, this paper randomly selects 400 training samples per class from the training dataset and 200 validation samples per class from the extra dataset. As described before, we employ the same parameter settings and have no extra data pre-processing or augmentation.

Table 8. Comparison with existing models on SVHN.

Model	#parameter	Test error (%)
Without data augmentation		
Maxout [6]	>5 M	2.47
Prob maxout [27]	>5 M	2.39
NIN [19]	1.98 M	2.35
DSN [17]	1.98 M	1.92
RCNN [18]	2.67 M	1.77
Tree+Max-Avg [16]	4.00M	**1.69**
SAPNet-32	0.76 M	1.87
SAPNet-48	1.71 M	1.75
SAPNet-64	3.03 M	**1.71**
With data augmentation		
Multi-digit number recognition [5]	>5 M	2.16
DropConnect [33]	5 networks	1.94

On SVHN benchmark, we test three SAPNet models that have different convolutional kernel sizes. Our experimental results illustrate that SAPNet-64 yields a test error of 1.71% on SVHN, which is only 0.02% less than the state of the art model Tree+Max-Avg [16]. It is the second ranked CNN model.

5 Conclusion

In this paper, we propose a new SAPNet model. First, a SAP pooling method is presented, where pooling area is randomly transformed. It expands feature map space and thus greatly decreases model parameters. Second, a generic CNN architecture that looks like three stacked T-shaped cubes is designed. In such architecture, the number of kernels in convolutional layer preceding any pooling layer is doubled and all learnable weight layers are combined with batch normalization and dropout with a small ratio. Finally, the experimental results show that our SAPNet requires fewer parameters than classic CNN models. Our SAPNet-64 yields state of the art test error of 5.57% on CIFAR-10. On CIFAR-100, the SAPNet-64 achieves a test error of 27.59%. On MNIST, the SAPNet-32 with 0.76 million parameters receives the best result of 0.29%. Our SAPNet-64 obtains a test error of 1.71% on SVHN benchmark, which is ranked second in all the existing machine learning models.

ACKNOWLEDGEMENTS

The authors would like to be grateful to the anonymous reviewers for their valuable comments that considerably contributed to improve this paper. This work was supported in part by the National Science Foundation of China (NSFC) under Grant Nos. 91420106, 90820305, and 60775040, and by the National High-Tech R&D Program of China under Grant No. 2012AA041402.

REFERENCES

[1] Dan Ciresan, Ueli Meier, and Jürgen Schmidhuber, 'Multi-column deep neural networks for image classification', in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pp. 3642–3649. IEEE, (2012).

[2] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter, 'Fast and accurate deep network learning by exponential linear units (elus)', *arXiv preprint arXiv:1511.07289*, (2015).

[3] Kunihiko Fukushima, 'Cognitron: A self-organizing multilayered neural network', *Biological cybernetics*, **20**(3-4), 121–136, (1975).

[4] Kunihiko Fukushima, 'Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position', *Biological cybernetics*, **36**(4), 193–202, (1980).

[5] Ian J Goodfellow, Yaroslav Bulatov, Julian Ibarz, Sacha Arnoud, and Vinay Shet, 'Multi-digit number recognition from street view imagery using deep convolutional neural networks', *arXiv preprint arXiv:1312.6082*, (2013).

[6] Ian J Goodfellow, David Warde-farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio, 'Maxout networks', in *ICML*, pp. 1319–1327, (2013).

[7] Benjamin Graham, 'Fractional max-pooling', *arXiv preprint arXiv:1412.6071*, (2014).

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, 'Deep residual learning for image recognition', *arXiv preprint arXiv:1512.03385*, (2015).

[9] Sergey Ioffe and Christian Szegedy, 'Batch normalization: Accelerating deep network training by reducing internal covariate shift', *arXiv preprint arXiv:1502.03167*, (2015).

[10] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al., 'Spatial transformer networks', in *Advances in Neural Information Processing Systems*, pp. 2008–2016, (2015).

[11] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross B Girshick, Sergio Guadarrama, and Trevor Darrell, 'Caffe: Convolutional architecture for fast feature embedding.', in *ACM Multimedia*, volume 2, p. 4, (2014).

[12] Alex Krizhevsky, *Learning multiple layers of features from tiny images*, Master's thesis, Department of Computer Science, University of Toronto, 2009.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, 'Imagenet classification with deep convolutional neural networks', in *Advances in neural information processing systems*, pp. 1097–1105, (2012).

[14] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel, 'Backpropagation applied to handwritten zip code recognition', *Neural computation*, **1**(4), 541–551, (1989).

[15] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner, 'Gradient-based learning applied to document recognition', *Proceedings of the IEEE*, **86**(11), 2278–2324, (1998).

[16] Chen-Yu Lee, Patrick W Gallagher, and Zhuowen Tu, 'Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree', *arXiv preprint arXiv:1509.08985*, (2015).

[17] ChenYu Lee, Saining Xie, Patrick Gallagher, Zhengyou Zhang, and Zhuowen Tu, 'Deeply-supervised nets', in *AISTATS*, pp. 562–570, (2015).

[18] Ming Liang and Xiaolin Hu, 'Recurrent convolutional neural network for object recognition', in *CVPR*, pp. 3367–3375, (2015).

[19] Min Lin, Qiang Chen, and Shuicheng Yan, 'Network in network', in *ICLR*, (2014).

[20] Dmytro Mishkin and Jiri Matas, 'All you need is a good init', *arXiv preprint arXiv:1511.06422*, (2015).

[21] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng, 'Reading digits in natural images with unsupervised feature learning', in *NIPS*, p. 5, (2011).

[22] Yunchen Pu, Xin Yuan, Andrew Stevens, Chunyuan Li, and Lawrence Carin, 'A deep generative deconvolutional image model', *arXiv preprint arXiv:1512.07344*, (2015).

[23] Karen Simonyan and Andrew Zisserman, 'Very deep convolutional networks for large-scale image recognition', *arXiv preprint arXiv:1409.1556*, (2014).

[24] Jasper Snoek, Oren Rippel, Kevin Swersky, Ryan Kiros, Nadathur Satish, Narayanan Sundaram, Mostofa Patwary, Mr Prabhat, and Ryan Adams, 'Scalable bayesian optimization using deep neural networks', in *Proceedings of the 32nd International Conference on Machine Learning (ICML-15)*, pp. 2171–2180, (2015).

[25] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller, 'Striving for simplicity: The all convolutional net', *arXiv preprint arXiv:1412.6806*, (2014).

[26] Jost Tobias Springenberg, Alexey Dosovitskiy, Thomas Brox, and Martin Riedmiller, 'Striving for simplicity: The all convolutional net', in *ICLR*, (2015).

[27] Jost Tobias Springenberg and Martin Riedmiller, 'Improving deep neural networks with probabilistic maxout units', *arXiv preprint arXiv:1312.6116*, (2013).

[28] Nitish Srivastava and Ruslan R Salakhutdinov, 'Discriminative transfer learning with tree-based priors', in *NIPS*, pp. 2094–2102, (2013).

[29] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber, 'Training very deep networks', in *NIPS*, pp. 2368–2376, (2015).

[30] Marijn F Stollenga, Jonathan Masci, Faustino Gomez, and Jürgen Schmidhuber, 'Deep networks with internal selective attention through feedback connections', in *NIPS*, pp. 3545–3553, (2014).

[31] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich, 'Going deeper with convolutions', in *CVPR*, pp. 1–9, (2015).

[32] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna, 'Rethinking the inception architecture for computer vision', *arXiv preprint arXiv:1512.00567*, (2015).

[33] Li Wan, Matthew Zeiler, Sixin Zhang, Yann L Cun, and Rob Fergus, 'Regularization of neural networks using dropconnect', in *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pp. 1058–1066, (2013).

[34] Matthew D Zeiler and Rob Fergus, 'Stochastic pooling for regularization of deep convolutional neural networks', *arXiv preprint arXiv:1301.3557*, (2013).

The Storage And Analytics Potential Of HBase Over The Cloud: A Survey

Georgios Drakopoulos and **Andreas Kanavos** and **Vasileios Megalooikonomou**[1]

Abstract. Apache HBase, a mainstay of the emerging Hadoop ecosystem, is a NoSQL key-value and column family hybrid database which, unlike a traditional RDBMS, is intentionally designed to scalably host large, semistructured, and heterogeneous data. Prime examples of such data are biosignals which are characterized by large volume, high volatility, and inherent multidimensionality. This paper reviews how biomedical engineering has recently taken advantage of HBase, with an emphasis over cloud, in order to reliably host cardiovascular and respiratory time series. Moreover, the deployment of offline biomedical analytics such as the Hilbert-Huang transform over HBase is explored.

1 INTRODUCTION

Biosignal analysis constitutes an integral part of biomedical engineering. In a typical scenario, long cardiovascular and respiratory time series from a large number of subjects are stored in a database system, possibly along with other relevant information such as virtual patient model, electronic health records, sensor recordings from smart homes, daily activity reports from smart clothes, and biochemical blood examination results. Although the above data may already have been passed as input to a lightweight online analytics system, the bulk of the analysis will be conducted in an offline system.

This holds especially true in group analysis settings with a large number of subjects, where computational requirements for even basic analytics can easily become overwhelming. For instance, in order to obtain empirical estimates of the variance and the kyrtosis coefficient in a set of n time series each of length p the magnitudes of the number of operations τ_v and τ_k respectively are

$$\tau_v = \mathrm{O}\left(\binom{n}{2}p\right) = \mathrm{O}\left(pn^2\right) \tag{1}$$

and

$$\tau_k = \mathrm{O}\left(\binom{n}{3}p\right) = \mathrm{O}\left(pn^3\right) \tag{2}$$

The direct computation of even τ_v for either a large p or n is prohibitively expensive. Although efficient sublinear methodologies which can become the algorithmic cornerstone of a number of analytics have been proposed [2][21], it is clear that the storage and analysis infrastructure should be able to scale with p and n. This includes not only the algorithmic cost but also the actual implementation complexity such as memory and disk requirements, network protocols and the associated stack size, as well as redundancy and fault tolerance [41][39][30][31][33].

[1] University of Patras, Rion Campus Building B, Patras 26500 Hellas, email: {drakop,kanavos,vasilis}@ceid.upatras.gr

Two interrelated challenges common in biomedical signal processing are the number and the patterns of missing values and outliers, both in their own way important when forming data mining and machine learning algorithms for biomedical data. Although random missing values can be expected, especially in manual data input or annotation, systematic ones may well indicate equipment malfunction or even a methodological error. On the other hand, outliers, provided they can be safely attributed to the subject under study, may be generated by an unknown or rare physiological state worth investigating. Therefore, depending on the context, outliers might carry a significant semantic weight. Moreover, both missing values and outliers play an important role to biosignal compression, representation, and modelling [14][32][39][40][4][42].

The primary contribution of this survey is the exploration of applications of HBase in bioengineering and specifically in the offline analysis of large cardiovascular and respiratory time series. Moreover, key-value and column family databases are briefly overviewed. Finally, the scientific literature for analytics for offline biosignal processing is summarized.

The remainder of this survey is organized as follows. Section 2 briefly overviews scientific literature regarding NoSQL databases and HBase. Fundamental cloud properties are explained in section 3. Section 4 outlines the characteristics of cardiovascular and respiratory biosignals. Finally, research directions are discussed in 5.

2 ACID or BASE?

Although the heading might seem as a chemistry question, it is essentially a choice over database operating requirements. Recently, at least partly due to the advent of the Internet of Things (IoT), the Semantic Web, and pervasive computing, new database paradigms have been developed resulting in a database family collectively known as NoSQL databases whose key points are summarized in properties 1 and 2. Table 1 summarizes the four primary NoSQL technologies, namely the key-value, column family, document, and graph databases [16][15][18]. HBase is considered a hybrid between key-value and column family databases.

Database	Data type
Graph	Linked data and conceptual graphs
Key-value	Associative or key-value array
Document	JSON or BSON documents
Column family	Wide and recursively nested tables

Table 1. NoSQL data types.

Property 1 *[15] NoSQL databases are schemaless.*

Property 2 *[26][34] The characteristics of NoSQL databases are:*

- ***B**asic **A**vailability. The database is operational most of the time. The percentage of downtime depends on the local operational requirements.*
- ***S**oft state. The database does not have to be write consistent. Also replicas do not have to be mutually consistent sometimes.*
- ***E**ventual consistency. Replicas may be inconsistent temporarily.*

Theorem 1 *Brewer CAP theorem [9][8][7]: A database system can at most possess simultaneously two of the following three properties: Consistency, Atomicity, and Partition tolerance.*

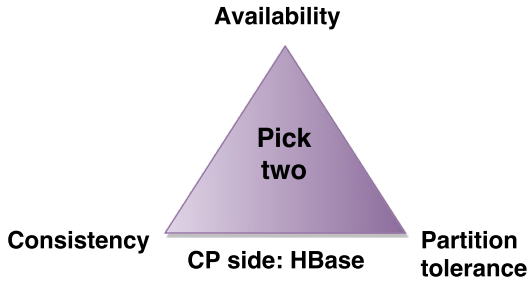


Figure 1. CAP theorem and HBase.

HBase is a sparse, distributed, persistent multidimensional sorted map database management system running on top of a distributed file system, typically HDFS. It is an ongoing, top level Apache project and an integral part of the Hadoop ecosystem. Its primary characteristic is that it is open source, distributed, and non-relational. According to the CAP theorem, it is a CP database. Also, in the NoSQL taxonomy it is a combination of key-value and column family databases [27][24].

The primary data structure of HBase, at least conceptually, is the associative array. As such, design principles for HBase differ from those of the relational world. HBase shares the same data model with Google BigTable [11][28]. In HBase tables comprise of rows and columns, with data rows having a sortable key and an arbitrary number of columns. Tables are sparsely stored, so that rows in the same table can have widely-varying columns, if so deemed by the application. HBase is tuned for sparse data stored in a few wide rows which can be used as input or output of MapReduce jobs over Hadoop [22]. Internally the data are placed in indexed StoreFiles that exist on HDFS for high-speed lookups [28].

An HBase table comprises of a set of regions, each stored in a region server [10]. Regions form the basic data building blocks of HBase and as such they are heavily employed by the underlying Hadoop framework [36]. The region contains store objects that correspond to column families as well as MemStore, an in-memory write cache [10][6][1][13]. Region rebalancing is a critical performance issue handled by the HMaster component.

Delving deeper, HBase relies on Bloom filters in order to index the sparse data [25]. A critical requirement is that rows should be sorted according to the key value. Therefore, rows with keys which are close, lexicographically or otherwise depending on the key nature, are very likely to be physically stored to the same machine. Therefore, choosing a row key convention is of paramount importance. For example, in a table whose keys are domain names it makes the most sense to list these names in reverse order notation so that rows about a subdomain will be near the parent domain row.

Finally, Apache project Phoenix attempts to bridge HBase with the relational world by allowing the formulation of SQL queries and the use of OLTP analytics over Hadoop for low latency applications. Thus, existing SQL and JDBC APIs with full ACID transaction capabilities can be used over an HBase backbone.

3 THE CLOUD AND THE *aaS MODELS

The cloud is an efficient way of distributing and abstracting computational resources [23]. Besides massive scalability, it offers efficient and decentralized resource management including sharing. Thus, asset utilization is typically high. The cloud comes at three major service abstraction levels, namely IaaS, PaaS, and SaaS, depending on the operational requirements.

Infrastructure as a Service (IaaS) model offers the least resource abstraction layer. Still, there is a broad range of virtual services including computing hardware, location, data partitioning, scaling, security, and backup [28]. Besides these basic services, IaaS clouds often offer additional resources such as a virtual-machine disk-image library, raw block storage, file or object storage, firewalls, load balancers, IP addresses, virtual local area networks (VLANs), and software bundles. A hypervisor or, in rare cases, even a powerful mainframe acts as the host to a number of guest virtual machines. Pools of hypervisors within the cloud OS can support large numbers of guests as well as an impressive scaling capability [13]. IaaS is the model of choice mainly for network and database administrators. A prime example is Amazon Elastic Compute Cloud, an IaaS cloud platform available to both industry and academia [5][12].

Platform as a Service (PaaS) aims at higher abstraction and less resource management at the expense of reduced customization flexibility. It is commonly selected to provide a reliable and integrated working environment to application developers. As such, PaaS places emphasis on virtual hardware diversity for facilitating testing purposes as well on software tools like dynamic OS libraries, development and execution environments, databases, and Web servers. PaaS end users do not directly manage the underlying cloud infrastructure such as network interface cards, servers, or storage. PaaS is further divided to Integration PaaS (iPaas) and Data PaaS (dPaaS), where the former facilitates rapid software development and the latter is oriented towards data intensive application creation and testing. PaaS examples include Microsoft Azure and Google App Engine.

Software as a Service (SaaS) model, finally, is used primarily in order to access and manage application software and databases. Under SaaS users have less flexibility compared to PaaS in order to achieve maximum standardization and ease of installation and maintenance. Consequently, typical users include IT and devops personnel and small business administrators, back end operators, or power users. Under SaaS the cloud framework is less transparent and configurable, as only a few software components are directly controlled, but at the same time it is more reliable, especially for novice users.

4 BIOSIGNALS IN HBase

HBase has been the database of choice or a number of applications where biosignals had to be saved and processed. For instance, the efficient processing of clinical signal data was a vital step towards multivariate analysis in order to develop better ways of describing the clinical status of a patient in [28]. Additionally, in [36], a novel

approach for storing and processing clinical signal data, including ECG, based on the Apache HBase distributed column-store and the MapReduce programming paradigm with an integrated web-based data visualization layer is presented. In this work, authors proposed a system where the computation was parallelized, thus trying to resolve the problem of increasing numbers of sensors and resulting data.

Authors in [29] present the notion of Health Monitoring as a Service (HMaaS) in a cloud-based personal health sensor data management platform. The proposed framework will act as a host to an ecosystem of scientists, medical practitioners, developers, as well as professionals with the aim of publishing their data analysis models as utilities in the cloud. Authenticated users can access those services and utilize their collected sensor data without any expert knowledge. The feasibility of this approach is supported by an ECG-based health monitoring service application deployed in addition to the main system.

Finally, in [20] a quick method for regularizing long ECG signals based on finite differences was proposed. This method is based on a tradeoff between two factors, one measuring the distance of the desired regularized version from the raw data and the other measuring the smoothness of the regularized data version. As noted, this approach is almost linearly scalable and HBase over cloud can provide a viable framework for such a scaling. This infrastructure would be also beneficial to the spectral analysis proposed in [38] for ECG signals, as the performance HBase can be fine-tuned to efficiently answer queries regarding the spectral content of the original ECG data.

5 DISCUSSION

This survey serves as a starting point for describing how HBase over the cloud can serve as a viable alternative for storing and analysing offline large biosignals with an emphasis to cardiovascular and respiratory time series. A number of examples has been described while the infrastructure description was intentionally kept on a minimum, as it is transparent to the end user.

Future research directions include the deployment of HBase as part of multimodal medical information processing architectures such as the one proposed in [19], where database interoperability and transparency plays a crucial role to data fusion. HBase over the cloud can be a scalable means for storing medical documents once they have been retrieved by specialized information retrieval systems such as [17], where a tensor-based retrieval system utilizing MeSH was proposed for document search in PubMed. Also, biomedical association rules mining outlined in [35] can be converted to be compatible with NoSQL operating parameters instead of relational ones and stored in such a system.

As a concluding remark, signs from computer fields are encouraging regarding the adoption of HBase. In social media analysis a cloud-based architecture was proposed in [3]. Authors aim at creating a sentiment analysis tool for Twitter data based on Apache Spark cloud framewor. There tweets are classified tweets using supervised learning techniques. Also, some pre-processing steps are introduced for achieving better results in sentiment analysis, whereas the classification algorithms are used for both binary and ternary classification. The proposed system was trained and validated with real data crawled by Twitter and in following results are compared with the ones from real users. In addition, in [37], authors present a novel method for Sentiment Learning in the Spark framework; the proposed algorithm exploits the hashtags and emoticons inside a tweet, as sentiment labels, and proceeds to a classification procedure of diverse sentiment types in a parallel and distributed manner.

REFERENCES

[1] Amitanand S Aiyer et al., 'Storage infrastructure behind Facebook messages: Using HBase at scale', *IEEE Data Eng. Bull.*, **35**(2), 4–13, (2012).

[2] Brain Babcock, Mayur Datar, Rajeev Motwani, and Liadan O'Callaghan, 'Maintaining variance and k-medians over data stream windows', in *Proceedings of the twenty-second ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '03, pp. 234–243, (2003).

[3] Alexandros Baltas, Andreas Kanavos, and Athanasios Tsakalidis, 'An apache spark implementation for sentiment analysis on twitter data', in *Algorithmic Aspects of Cloud Computing - Second International Workshop, ALGOCLOUD 2016*, (2016).

[4] Andrew Barron, Jorma Rissanen, and Bin Yu, 'The minimum description length principle in coding and modeling', *IEEE Transactions on Information Theory*, **44**(6), 2743–2760, (1998).

[5] Vijayalakshmi Bhupathiraju and Ravi Prasad Ravuri, 'The dawn of big data-HBase', in *2014 Conference on IT in Business, Industry and Government (CSIBIG)*, pp. 1–4, (2014).

[6] Dhruba Borthakur et al., 'Apache Hadoop goes realtime at Facebook', in *ACM SIGMOD International Conference on Management of data*, pp. 1071–1080, (2011).

[7] Eric Brewer, 'A certain freedom: Thoughts on the CAP theorem', in *29th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing PODC*, pp. 335–335, (2010).

[8] Eric Brewer, 'CAP twelve years later: How the "rules" have changed', *Computer*, **45**(2), 23–29, (2012).

[9] Eric A Brewer, 'Towards robust distributed systems', in *19th ACM SIGACT-SIGOPS Symposium on Principles of Distributed Computing*, volume 7 of *PODC '00*, (2000).

[10] Andrei Broder and Michael Mitzenmacher, 'Network applications of bloom filters: A survey', *Internet mathematics*, **1**(4), 485–509, (2004).

[11] Fay Chang et al., 'Bigtable: A distributed storage system for structured data', *ACM Transactions on Computer Systems*, **26**(2), (2008).

[12] Max Chevalier, Mohammed El Malki, Arlind Kopliku, Olivier Teste, and Ronan Tournier, 'Benchmark for olap on nosql technologies comparing nosql multidimensional data warehousing solutions', in *2015 IEEE 9th International Conference on Research Challenges in Information Science (RCIS)*, pp. 480–485, (2015).

[13] TK Das and P Mohan Kumar, 'Big data analytics: A framework for unstructured data analysis', *International Journal of Engineering Science and Technology*, **5**(1), 153, (2013).

[14] Anna MR Dixon, Emily G Allstot, Daibashish Gangopadhyay, and David J Allstot, 'Compressed sensing system considerations for ECG and EMG wireless biosensors', *IEEE Transactions on Biomedical Circuits and Systems*, **6**(2), 156–166, (2012).

[15] Georgios Drakopoulos and Aikaterini Baroutiadi, 'Implementing centrality measures for Neo4j', in *Proceedings of the 8th ECESCON*. ECESCON, (2015).

[16] Georgios Drakopoulos, Aikaterini Baroutiadi, and Vasileios Megalooikonomou, 'Higher order graph centrality measures for Neo4j', in *Proceedings of the 6th International Conference on Information, Intelligence, Systems, and Applications IISA*, (2015).

[17] Georgios Drakopoulos and Andreas Kanavos, 'Tensor-based document retrieval over Neo4j with an application to PubMed mining', in *Proceedings of the 6th International Conference of Information, Intelligence, Systems, and Applications*, IISA 2016.

[18] Georgios Drakopoulos, Andreas Kanavos, Christos Makris, and Vasileios Megalooikonomou, 'On converting community detection algorithms for fuzzy graphs in Neo4j', in *5th International Workshop on Combinations of Intelligent Methods and Applications*, CIMA 2015, (2015).

[19] Georgios Drakopoulos and Vasileios Megalooikonomou, 'A graph framework for multimodal medical information processing', in *Proceedings of the Digital World 2016*, MATH 2016, (April 2016).

[20] Georgios Drakopoulos and Vasileios Megalooikonomou, 'Regularizing large biosignals with finite differences', in *Proceedings of the 6th International Conference of Information, Intelligence, Systems, and Applications*, IISA 2016, (2016).

[21] Vishal Gaur, Saravanan Kesavan, Ananth Raman, and Marshall L Fisher, 'Estimating demand uncertainty using judgmental forecasts',

Manufacturing and Service Operations Management, **9**(4), 480–491, (2007).
[22] Lars George, *HBase: The definitive guide*, O'Reilly Media, Inc., 2011.
[23] Dan Han and Eleni Stroulia, 'A three-dimensional data model in HBase for large time-series dataset analysis', in *2012 IEEE 6th International Workshop on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA)*, pp. 47–56, (2012).
[24] Shan Huang, Botao Wang, Jingyang Zhu, Guoren Wang, and Ge Yu, 'R-hbase: A multi-dimensional indexing framework for cloud computing environment', in *2014 IEEE International Conference on Data Mining Workshop*, pp. 569–574, (2014).
[25] P. C. Ivanov, M. G. Rosenblum, C.-K. Peng, J. Mietus, S. Havlin, H. E. Stanley, and A. L. Goldberger, 'Scaling behaviour of heartbeat intervals obtained by wavelet-based time-series analysis', *Nature*, **383**, 323–327, (1996).
[26] Stavros Kontopoulos and Georgios Drakopoulos, 'A space efficient scheme for graph representation', in *26th International Conference on Tools with Artificial Intelligence ICTAI*, pp. 299–303, (2014).
[27] Tuan Dinh Le, Seong Hoon Kim, Minh Hoang Nguyen, Daeyoung Kim, Seung Young Shin, Kyung Eun Lee, and Rodrigo da Rosa Righi, 'Epc information services with no-sql datastore for the internet of things', in *2014 IEEE International Conference on RFID (IEEE RFID)*, pp. 47–54, (2014).
[28] Li Li and Yaqi Song, 'Distributed storage of massive RDF data using HBase', *Journal of Communication and Computer*, **8**(5), 325–328, (2011).
[29] Yang Li, Li Guo, and Yike Guo, 'Enabling health monitoring as a service in the cloud', in *2014 IEEE/ACM 7th International Conference on Utility and Cloud Computing*, UCC '14, pp. 127–136, (2014).
[30] Yang Li, Li Guo, Chao Wu, Chun-Hsiang Lee, and Yike Guo, 'Building a cloud-based platform for personal health sensor data management', in *IEEE-EMBS International Conference on Biomedical and Health Informatics (BHI)*, pp. 223–226. IEEE, (2014).
[31] Sheng Liang and Yang Yang, 'Towards performance evaluation of HBase based multidimensional cloud index', in *2015 4th International Conference on Computer Science and Network Technology (ICCSNT)*, volume 1, pp. 629–632, (2015).
[32] Hossein Mamaghanian, Nadia Khaled, David Atienza, and Pierre Vandergheynst, 'Compressed sensing for real-time energy-efficient ECG compression on wireless body sensor nodes', *IEEE Transactions on Biomedical Engineering*, **58**(9), 2456–2466, (2011).
[33] Patrick E McSharry, Gari D Clifford, Lionel Tarassenko, and Leonard A Smith, 'A dynamical model for generating synthetic electrocardiogram signals', *IEEE Transactions on Biomedical Engineering*, **50**(3), 289–294, (2003).
[34] Jim Melton and Alan R Simon, *Understanding the new SQL: A complete guide*, Morgan Kaufmann, 1993.
[35] Rosa Meo, Giuseppe Psaila, and Stefano Ceri, 'A new SQL-like operator for mining association rules', in *VLDB*, volume 96, pp. 122–133, (1996).
[36] Andrew V Nguyen, Rob Wynden, and Yao Sun, 'HBase, MapReduce, and integrated data visualization for processing clinical signal data', in *AAAI Spring Symposium: Computational Physiology*, volume 2011, (2011).
[37] Nikolaos Nodarakis, Spyros Sioutas, Athanasios K. Tsakalidis, and Giannis Tzimas, 'Large scale sentiment analysis on twitter with spark', in *Proceedings of the Workshops of the EDBT/ICDT 2016 Joint Conference, EDBT/ICDT Workshops 2016.*, (2016).
[38] Pontus B Persson, 'Spectrum analysis of cardiovascular time series', *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, **273**(4), R1201–R1210, (1997).
[39] P Vignesh Raja and E Sivasankar, 'Modern framework for distributed healthcare data analytics based on Hadoop', in *Information and Communication Technology-EurAsia Conference*, pp. 348–355, (2014).
[40] Jorma Rissanen, 'Hypothesis selection and testing by the MDL principle', *The Computer Journal*, **42**(4), 260–269, (1999).
[41] Kerk Chin Wee and Mohd Soperi Mohd Zahid, 'Auto-tuned Hadoop MapReduce for ECG analysis', in *2015 IEEE Student Conference on Research and Development (SCOReD)*, pp. 329–334, (2015).
[42] Zhaomin Zhang and Daming Wei, 'A new ECG identification method using Bayes teorem', in *TENCON 2006-2006 IEEE Region 10 Conference*, pp. 1–4, (2006).

Experimental and causal view on information integration in autonomous agents

Philipp Geiger[1] and **Katja Hofmann**[2] and **Bernhard Schölkopf**[3]

Abstract. The amount of digitally available but heterogeneous information about the world is remarkable, and new technologies such as self-driving cars, smart homes, or the internet of things may further increase it. In this paper we examine certain aspects of the problem of how such heterogeneous information can be harnessed by autonomous agents. After discussing potentials and limitations of some existing approaches, we investigate how *experiments* can help to obtain a better understanding of the problem. Specifically, we present a simple agent that integrates video data from a different agent, and implement and evaluate a version of it on the novel experimentation platform *Malmo*. The focus of a second investigation is on how information about the hardware of different agents, the agents' sensory data, and *causal* information can be utilized for knowledge transfer between agents and subsequently more data-efficient decision making. Finally, we discuss potential future steps w.r.t. theory and experimentation, and formulate open questions.

1 Introduction

Increasing amounts of heterogeneous information are recorded and connected, and this trend is likely to continue in the light of new technology such as self-driving cars, smart homes with domestic robots, or the internet of things. Intuitively, it makes sense to design autonomous agents in a way that they automatically integrate all relevant and well-structured information on their environment that is available. Various aspects of the problem of designing such agents have been investigated previously. In this paper we approach the problem from two directions which, to our knowledge, have not been (exhaustively) examined yet: using sophisticated simulated experiments, on a practical level, and causal models, on a more theoretical level. The complexity of the problem allows us only to take small steps.

1.1 Main contributions

The main contribution of this paper consists of two investigations:

- In Section 5 we use a simulated *experimentation platform Malmo* to obtain a *better understanding* of the problem of integrating heterogeneous information. More specifically, we present a simple agent that harnesses video data from a different agent, and implement and evaluate a version of it.
- In Section 6 we investigate how detailed information on the *hardware of different agents* (we consider self-driving cars as example), their sensory data, and physical or *causal information* can be utilized for knowledge transfer between them and subsequent more data-efficient decision making.

The common structure of both investigations is that we start with a description of a scenario that captures certain core aspects of the general problem, in particular containing a variety of heterogeneous information sources, and then sketch a method to perform information integration and subsequent decision making in these scenarios. After experimentally evaluating the method, or illustrating it based on a toy example, we conclude both investigations with a discussion of the advantages and limitations of the respective methods.

A reoccurring theme in our investigations is that we try to treat as much information (including models) as possible explicitly as input to algorithms instead of implicitly encoding it into algorithms. Our hope is that this sheds a better, more explicit light on the problem.

1.2 Structure of the paper

The paper is structured as follows: We introduce the experimentation platform and basic concepts in Section 2. In Section 3, we formulate the problem. In Section 4, investigate potentials and limitations of existing approaches for the problem. In Sections 5 and 6, we present our two main investigations. In Section 7, we discuss future directions and pose open questions. We conclude with Section 8.

2 Preliminaries

Here we introduce the concepts, models and the experimentation platform we will use in the paper.

Autonomous agents. By an *(autonomous) agent* we mean a mechanism which, at each time point t, takes some input from the *environment*, in particular its sensory data which we refer to as *observation*, and outputs some *action* that influences the environment. Moreover, by an *intelligent (autonomous) agent* we mean an autonomous agent which is successful in using its inputs and outputs for given *tasks*, i.e., specific goals w.r.t. the environment, often encoded by a *reward* or *utility* function.

Note that in this paper we do not define a clear boundary between agent and environment. Usually, we consider the hardware platform of an agent (e.g., the car) as part of the agent. This particularly has to be kept in mind when we talk of several agents in the "same environment": the hardware of the agents may still differ. (It is almost a philosophical problem to define what precisely the "same environment" means. Here we simply suggest to interpret this notion as if it

[1] Max Planck Institute for Intelligent Systems, Tübingen, Germany; email: philipp.geiger@tuebingen.mpg.de
[2] Microsoft Research Cambridge, Cambridge, United Kingdom; email: katja.hofmann@microsoft.com
[3] Max Planck Institute for Intelligent Systems, Tübingen, Germany; email: bs@tuebingen.mpg.de

were used in an everyday conversation. We pose a related question in Section 7.)

When we consider some agent A w.r.t. which some task is given and for which we want to infer good actions, while information may come from (sensory data of) a collection C of other agents, then we refer to A as *target agent* and the agents in C as *source agents*.

Experimentation platform "Malmo". For the experiments in Section 5 we will use the software *Malmo*, a simulated environment for experimentation with intelligent agents, that was introduced recently [4]. Malmo is based on "Minecraft", which is an open-ended computer game where players can explore, construct, collaborate, and invent their own "games within the game" or tasks. The Malmo platform provides an abstraction layer on top of the game through which one or more agents observe the current state of the world (observations are customizable) and interact with it through their specific action sets (or actuators). The advantage of Malmo is that it reflects important characteristics of the problem instances we will introduce in Section 3. To illustrate the platform, three sample observations of an agent in three different maps in Malmo will be depicted in Figure 1.

Causal models. Mathematically, a *causal model* [10, 14] M over a set V of variables consists of a directed acyclic graph (DAG) G with V as node set, called *causal diagram* or *causal DAG*, and a conditional probability density $p_{X|PA_X=pa_X}$ (for all pa_X in the domain of PA_X) for each $X \in V$, where PA_X are the parents of X in G. Given a causal model M and a tuple of variables Z of M, the *post-interventional causal model* $M_{\mathrm{do}\ Z=z}$ is defined as follows: drop the variables in Z and all incoming arrows from the causal diagram, and fix the value of variables in Z to the corresponding entry of z in all remaining conditional densities. Based on this, we define the *post-interventional density of* Y *after setting* Z *to* z, denoted by $p_{Y|\mathrm{do}\ Z=z}$ or $p_{Y|\mathrm{do}\ z}$, by the the density of Y in $M_{\mathrm{do}\ Z=z}$.

On a non-mathematical level, we consider M to be a correct causal model of some part of reality, if it correctly predicts the outcomes of *interventions* in that part of reality (clearly there are other reasonable definitions of causation). Keep in mind that in this paper, in particular Section 6, will use causal models and causal reasoning in a more intuitive and sometimes less rigorous way, to not be limited by the expressive power of the current formal modeling language.

Note that we will use expressions like $p(x|y)$ as shorthand for $p_{X|Y}(x|y)$.

3 Problem formulation

Let us describe the problem we consider in this paper in more detail:

- *Given:* a task T w.r.t. some partially unknown environment E, and additional heterogeneous but well-structured information sources H (e.g., in the form of low-level sensory data, or in the form of high-level descriptions).
- *Goal:* design an agent A that automatically harnesses as much relevant information of H as possible to solve T; more specifically, it should use H to either improve an explicit model of the effects[4] of its actions, which then guides its actions, or let its actions directly be guided by H.

[4] In this sense, at least the target of the information integration is clear: modeling the dynamics or causal structure of the agent in the environment.

Note that alternatively, one could also formulate the problem by letting A only be an actuator, and not a complete agent, and include the agent's sensors into H. This might be a more precise formulation, however, for the sake of an intuitive terminology, we stick to the definition based on A being an agent.[5]

To illustrate the general problem, in what follows, we give three concrete examples of desirable scenarios in which agents automatically integrate heterogeneous information. Ideally, agents would be able to *simultaneously integrate information sources from all three examples*.

3.1 Example 1: sharing information between different self-driving cars

Consider self-driving cars. It is desirable that as much information about the environment can be shared amongst them. By such information we mean up-to-date detailed street maps, traffic information, information on how to avoid accidents etc. For instance, assume that one self-driving car leaves the road at some difficult spot due to some inappropriate action, since, for instance, the spot has not been visited by self-driving cars before (or newly appeared due to say some oil spill or rockfall). If we only consider other self-driving cars of the same hardware, this experience could directly be transferred to them by enforcing them not to perform the very action at the very spot. (I.e., for all cars of the same hardware one could treat the experience as if it was their own and make them "learn" from it in the usual reinforcement learning (RL) way.) However, if we assume that there are self-driving cars of different types, then it is not possible to transfer the experience, and thus avoid further accidents, in this straight-forward way.

3.2 Example 2: observing another agent

Consider domestic robots. A domestic robot may, with its sensor, observe humans how they handle doors, windows, light switches, or kitchen devices. It should be possible that domestic robots learn from such experience. For instance, one could imagine a robot to reason that, if it is able to operate the door knob in a similar way as a human did before, this would also open the door and and thus allow the robot to walk into the other room (to achieve some task).

3.3 Example 3: integrating high-level information

Consider an agent that arrives in a city it has never been to before. The goal is to get to a certain destination, say to the town hall. A resident may be able to explain the way in a simple language, with words such as "... follow this street until you come to a church, then turn right ...". Or a resident could provide a map and mark directions on the map. One could imagine that an autonomous agent could combine such a description with a model of the "local" (or "low-level") dynamics that is shared by most environments (which is closely related to the laws of physics). The model of the "local" dynamics could have been either hard-coded, or inferred based on exploration in other (related) environments. In principle it should be possible that

[5] Note that the problem we formulate here does not coincide with developing ("strong") artificial intelligence (AI), as defined, e.g., by the Turing test or simply based on human-level intelligence. We restrict to sources of information that are more or less well-structured - either quantitative measurements with a simple and clear relation to the physical world, or information in a language much more restrictive than natural language. Nonetheless, the formulated problem can be seen as one step from say RL into the direction of AI.

such a combination allows the agent to successfully navigate to the destination city hall.

4 Related work: potentials and limitations

Various research directions exist that address major or minor aspects of the problem formulated in Section 3. Here we discuss the most relevant such directions we are aware of, highlighting their potentials and limitations w.r.t. the problem. Keep in mind that Sections 5.4 and 6.4 contain additional discussions on the advantages of our approaches over these directions.

4.1 Reinforcement learning

One of the most powerful approaches to shaping intelligent autonomous agents is *reinforcement learning (RL)* [16]. Instead of explicitly hard-coding each detail of an agent, for each environment and objective individually, the idea is to take an approach which is more modular and based on learning instead of hard-coding: the supervisor only determines the reward function and then the agent ideally uses exploration of the unknown environment and exploitation of the gained experience (sensory data) to achieve a high cumulative reward.

Regarding the problem we consider in this paper, RL plays a key role for integration of information in the form of *recordings of an agent's own past*, or of an agent with the same hardware. However, as mentioned in Section 3, in contrast to RL, here we consider the problem of integrating information beyond such recordings, such as sensory data from agents with different hardware, or higher level information such as maps.

4.2 Learning from demonstrations

According to [1], in *learning from demonstrations (LfD)*, some "teacher" performs a trajectory which is recorded, and the goal is that a "learner" agent, based on this recording, infers and imitates (or utilizes) the teacher's "policy" (or the dynamics of the environment, or both). Central notions that [1] uses to analyze and distinguish various types of LfD problems are the *record mapping*, i.e., what aspects of the teacher's demonstration are measured and recorded, and the *embodiment mapping*, i.e., if the recorded actions can directly be implemented by the "learner" and lead to similar observations as the recoded ones, or if the recordings first have to be transformed "to make sense" for the learner.

Our problem formulation can be seen as a generalization of LfD. Based on this, while a significant part of the problem we consider can be addressed by LfD methods, others are beyond the scope of these methods: Instead of hand-crafting, e.g., the embodiment mapping for each agent individually, we aim at (semi-)automating the inference of the mapping from recordings of "source" agents to actions of a "target" agent. In particular, we propose to do such a (semi-)automation based on additional information sources on the hardware specifications of the agents involved (Section 6).[6]

[6] Note that there is some work on learning from observations only (not actions) of a "teacher" [12]. However, this approach does not allow to integrate information such as the map in Example 3. Note that a difference to our method in Section 5 is that, e.g., an estimate of the complete transition probability is necessary, while our method only requires an idea of the "low-level" dynamics.

Generally, we aim at integrating information from many different sources simultaneously (e.g., many other self-driving cars in Example 1 and many different forms of information as described in Example 1 through Example 3. In particular, we aim at learning from databases that contain desirable as well as undesirable trajectories (e.g., avoid similar accidents as in Example 1).

Clearly, we do not present methodology that fully tackles the above shortcomings of LfD methods in this paper. Rather, we make first steps towards such methodology in Sections 5 and 6.

4.3 Multi-agent systems

In multi-agent systems, collections of agents acting in a shared environment are studied [15]. One important task is collaboration between agents [9]. A common approach is to model the collection of agents again as a single agent, by considering tuples of actions and observations as single actions and observations. Learning-based methods have been extensively studied [5].

While multi-agent systems approaches often allow to share and transfer information between agents, regarding the problem we formulated in Section 3 they have certain limitations: Similar as LfD, they usually do not integrate higher-level information sources (as the map in Example 3) or explicit hardware specifications of the agents (which we do in Section 6). Furthermore, if the mapping from some source agents sensory data to a target agents action is learned via modeling all agents as a single one, then it seems difficult to add agents to an environment, while our preliminary investigation in Section 6 in principle allows for adding agents more easily. Also note that the task of collaboration between agents is rather external to the problem we consider.

4.4 Transfer learning for agents

The problem we consider is related to transfer learning for agents. For instance, [17] consider an example where, in the well-known mountain car example, experience should be transferred although the motor of the car is changed. This comes close to transferring experience between self-driving cars as we suggest in Example 1. However, the scope of methods reviewed in [17] is on transferring observation-action recordings or things such as policies, value functions etc. using an appropriate mapping, while the goal we pursue is to also integrate information which is usually not expressible in these terms (e.g., the map or natural language description in Example 3, or the observation of another agent in Example 2). Furthermore, in this paper we aim at integrating *many heterogeneous sources* of information, while in transfer learning, even though several sources of information may be considered, they are usually homogeneous.

4.5 Further related areas

Other related directions include the following. Recently, the experimentation with intelligent agents in platforms based on computer games has become popular [8]. To our knowledge, the current work is the first one to use such platforms to study the problem of information integration, or related problems such as LfD (Section 4.2).

The general integration and transfer of data (not focused on intelligent agents) using causal models has been studied by [11, 3]. The idea of integrating higher-level information (again not for intelligent agents though) has been studied, e.g., by [18]. The relation between intelligent agents and causal models has been studied from a more philosophical perspective, e.g., by [19].

Another related areas is computerized knowledge representation [13]. Compared to general approaches to knowledge representation, our focus is on knowledge about the *physical* world.

Another relevant area is integration of human knowledge [7].

5 Investigation 1: integration of "non-subjective" information, evaluated in a simulated environment

In this section we aim to shed light on the following aspects of the problem formulated in Section 3:

- Generally, what experiments, in particular in simulated environments, can be performed to better understand the problem?
- How can experimentation (exploration) help an agent to translate "non-subjective" experience not recorded by itself into its own "coordinate system" and use it for (successful) decision making?
- How can partial information on the dynamics, such as a controller that is known to work locally, be merged with "higher-level" information such as hints on the path to some goal position?
- How can we quantify the efficiency gain from additional information sources?

The investigation is structured as follows: in Section 5.1 we describe the scenario, in particular the available heterogeneous information sources, in Section 5.2 we sketch an information-integrating agent for that scenario, then, in Section 5.3, we evaluate an adapted version of the agent in a simulated environment, and last, in Section 5.4 we discuss the advantages of our method over those that do not use additional information, and some further aspects.

5.1 Scenario

Task. An agent A starts in some unknown landscape and the task is to get to some visually recognizable goal position as quickly as possible.

Available heterogeneous information. We assume the following information sources to be available:

- the agent's own sensory input in the form of images y_t and position signal q_t (which can be seen an "interactive information" source since the agent can "query" this source via its actions),
- the controller ctl, which can be seen as a summary of the agent A's past subjective experience regarding the invariant local "physical laws" of a class of environments[7],
- a video trajectory $y^*_{0:L}$ that is a first-person recording of another agent with similar (but not necessarily identical) hardware that runs to the goal in the same environment.

Relation to the problem formulated in Section 3. On the one hand, this scenario can be seen as a (very) simplistic version of the scenario described in Example 1: A is a self-driving car that is supposed to get to some marked goal in an unknown environment, and the video $y^*_{0:L}$ comes from other cars that have a similar video-recording device but different hardware (engine etc.).

On the other hand, this scenario can be seen as a simplistic version of the scenario described in Example 3: the unknown landscape is some unknown city A arrived in, and instead of a description of the way to the destination in simple natural language, it gets a sequence of photos that describe the path it has to take.

[7] Specifically, we assume that if the distance between position q_1 and q_2 is small, then ctl successfully steers from q_1 to q_2. Alternatively, ctl could be a local model of the dynamics which induces such a controller.

Algorithm 1 Agent that integrates first-person video of other agent

input: Controller ctl, video $y^*_{1:L}$.
for i = 1, . . . , L **do**
  Use local controller ctl, optimization method opt and interaction with the environment to search locally around the current position for the next $q_i = \arg\min_q dist(y^*_i, \mathbb{E}(Y|Q = q))$.
  Use ctl to go to q_i.
end for

Algorithm 2 Proof-of-concept of Algorithm 1 for Malmo

input: Controller ctl, video $y^*_{1:L}$.
set $r_0 =$ current position, once the mission starts
for i = 1, . . . , L **do**
  use ctl, opt and teleportation to locally search around position r_{i-1} for the next $r_i = \arg\min_r \mathbb{E}(dist(y^*_i, Y)|Q = r)$
end for
restart the mission
set $i := 0$.
while $i < L$ **do**
  use ctl to steer to r_i
  if current position is close to r_i **then**
    set $i := i + 1$
  end if
end while

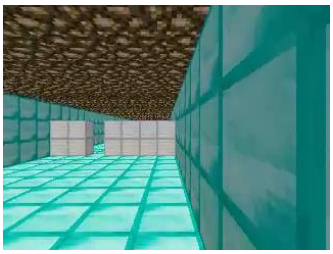
5.2 Method

First we sketch a general method, i.e., "software" for A, in Algorithm 1, assuming a (stochastic) optimization method opt and an image distance $dist$ as given (for concrete examples, see below). Note that in Algorithm 1 we denote by $\mathbb{E}(Y|Q = q)$ the mean image Y observed at position $Q = q$. The basic idea is that the agent uses local experimentation, based on prior knowledge of the local dynamics, to map the video $y^*_{1:L}$ into information (and eventually actions) that directly describes its own situation.

Although Algorithm 1 is in principle applicable to the experimental setup we consider in Section 5.3 below, we will evaluate Algorithm 2 instead, which is a simplified proof-of-concept implementation of it, making use of "teleportation", allowing the agent to directly jump to other positions without needing to navigate there. For Algorithm 2, as optimization method opt, we use simple grid search. (Note that instead one could use gradient descent or Bayesian optimization techniques.) Furthermore, we define the image distance $dist$ using Gaussian blur as follows: $dist(u, v) := \|N * (\bar{u} - \bar{v})\|$, where $\bar{u}$ is the normalization of u (i.e., subtraction by mean and division by standard deviation over the single pixels), N is a bivariate Gaussian with hand-tuned variance and $*$ is the convolution in both image dimensions.

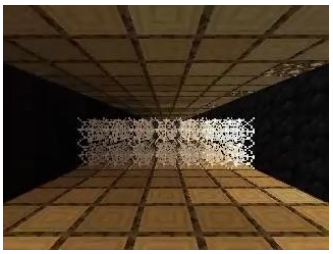
5.3 Empirical evaluation in a simulated environment

To evaluate Algorithm 2, we consider three simple "Parkours" missions in the experimentation platform Malmo, described in Section 2. These missions consist of simple maps that have a special, visually recognizable, position which is defined as goal. A short description



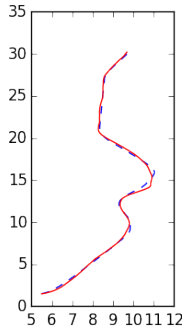
(a) Mission 1

(b) Mission 2

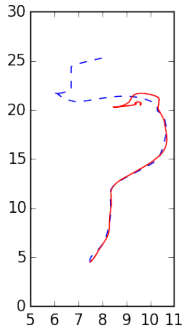


(c) Mission 3

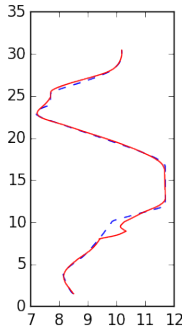
Figure 1: First frames of the respective missions, for illustration purposes.



(a) Mission 1



(b) Mission 2



(c) Mission 3

Figure 2: The "ground truth" position trajectory of the demonstrator $q^*_{0:L}$ (blue dashed line), and the position trajectory of Algorithm 2 $\hat{q}_{0:K}$, (red solid line) from top view (x- and y-axis correspond to x- and y-coordinate in the map. While Algorithm 2 fails in Mission 2 due to the repetitive structure of some wall, it succeeds in Missions 2 and 3 in spite of its simplicity.

Table 1: For each mission (row) a short description (column 2), and the outcome of Algorithm 2 applied to it (columns three and four).

Mission	**Description and image**	U	$\hat{q}_{0:K}$ **versus** $q^*_{0:L}$
1	Figure 1a. Two passages have to be passed.	success	Figure 2a
2	Figure 1b. With (mortal) lava next to the path.	fail	Figure 2b
3	Figure 1c. With spider webs that occlude vision and lead to slow motion.	success	Figure 2c

of the three missions is given in column two of Table 1. We generally restrict the possible actions to $[-1, 1]^2$, where the first dimension is moving forth and back, and the second is strafing (moving sideways). The *task* is to get to the goal position within 15 seconds in these maps.

For each mission, we record one trajectory performed by a human demonstrator, which solves the task. More specifically, we record positions, which we denote by $q^*_{0:L}$, and observations (video frames), denoted by $y^*_{0:L}$.

We run Algorithm 2 with inputs $y^*_{0:L}$ and a simple proportional controller [2] (for ctl), where we tuned the proportional constant manually in previous experiments (but without providing $q^*_{0:L}$ or the actions the human demonstrator took). Let $\hat{q}_{0:K}$ denote the trajectory of positions that Algorithm 2 subsequently takes in the map. Furthermore, let $U \in \{\text{fail}, \text{success}\}$ denote whether the position tracking while-loop of Algorithm 2 (line 8 and following) gets to the goal within 15 seconds. (This is a significantly weaker evaluation metric than considering the runtime of the complete Algorithm 2 of course.)

The outcome of the experiment is given in columns three and four of Table 1 and Figure 2.

5.4 Discussion

5.4.1 *Discussion of the experiment*

Outcome. As shown by Table 1 and Figure 2, Algorithm 2 is successful for Missions 1 and 3. It fails in Mission 2 due to the repetitive structure of a wall that fills the complete image that is observed at some point during $y^*_{1:L}$. This wall makes the mapping from position to observation (video frame) (locally) non-injective which makes the algorithm fail. Note that this problem could quite easily be overcome by using prior assumptions on the smoothness of $q^*_{0:L}$ together with considering more than one minimum of $dist$ as the potential true position (using, e.g., Bayesian optimization) or by searching for position sequences longer than 1 that match $y^*_{1:L}$.

Limitation of the experiment. A clear limitation of the experiment is that the human demonstrator that produced $y^*_{1:L}$ used the same (simulated) "hardware" as Algorithm 2, while the overall goal of this paper is to integrate heterogeneous information. However, we hope that this experiment can form the basis for more sophisticated ones in the future.

5.4.2 *Theoretical analysis and insight*

Efficiency gain from harnessing $y^*_{1:L}$. Assuming there are at most N positions which the demonstrator can reach within one time step, Algorithm 1 takes only about $O(L \cdot N)$ steps to get to the goal. Note that this theoretical analysis is supported by the empirical evaluation: Algorithm 2's trajectories - visualized in Figure 2 - are roughly as long as $y^*_{1:L}$ (note that the visualization does not show the local search of length N).

This has to be contrasted with an agent that does not integrate the information $y^*_{1:L}$, and therefore, in the worst case, has to search *all* positions in the map, a number which is usually is much higher than $O(L \cdot N)$ (roughly $O(L^2)$).

Comparison to LfD. The task we study is closely related to LfD. However, note that usually in LfD, the target agent ("learner") has access to the demonstrator's actions, which is not necessary for our method. Furthermore, in our method, in some sense, the target agent can be seen as translating $y^*_{1:L}$ into its own "coordinate system" itself, while this mapping is usually hand-crafted in LfD.

Some insights during the development of the method. An interesting insight during the development of Algorithm 5.2 was that while the low-pass filter only led to minor improvements, what really helped was to visit each position say three times and then optimize the distance over the *averaged* images. Furthermore, it was was surprising how well the simple Euclidean distance we used worked.

Limitations. Note that the method is limited to environments similar to landscapes where some stochasticity and variation may be contained, but not too much. For instance, if the environment varies to strongly in the dependence on the time an agent spends in the environment, the proposed method most likely fails since the tracking of $y^*_{1:L}$ usually takes longer than the original performance of it.

6 Investigation 2: integrating sensory data, hardware specifications and causal relations

In this section we aim to shed light on the following aspects of the problem formulated in Section 3:

- How can information on the the hardware specifications of various agents be used for knowledge transfer between them?
- To what extent can *causal models* help, e.g., for integrating those hardware specifications (i.e., information on the "data producing mechanisms")?[8]
- How can information from the *"subjective perspective"* of an agent (i.e., on the relation between its sensory measurements and its actions) be merged with information from an "outside perspective" (i.e., that of an engineer which sees the hardware specifications of an agent).

The investigation is structured as follows: in Section 6.1 we describe the scenario, in particular the available heterogeneous information sources, in Section 6.2 we outline an information-integrating agent for that scenario, then, in Section 6.3, we give an intuitive toy example of scenario and method, and last, in Section 6.4 we discuss advantages and limitations of our approach.

Keep in mind the definition of causal models in Section 2. It needs to be mentioned that at certain points in this section we will allow ourselves some extent of imprecision (in particular in the treatment of the (causal) model M and how it is inferred), since we aim at going beyond what current rigorous modeling languages allow.

6.1 Scenario

Task. We consider a scenario where a collection C of autonomous agents, think of self-driving cars, operates in a shared environment. (For simplicity we assume that the number of agents is small compared to the size of the environment, such that they do not affect each other.) We assume that while some hardware components of the agents differ, others are *invariant* between them. We assume that for each car a task (e.g., to track some trajectory) is given and fixed.

Available heterogeneous information. Note that we could allow C to vary over time, e.g., to account for the fact that new cars get on the road every day, however, for the sake of a simple exposition, we leave it fixed here. We assume the following information sources to be available at time t:

[8] Another reason why causal reasoning could help is that in the end we are interested in the *causal effects* of an agent on its environment, and not just correlational information.

Algorithm 3 Integration and control algorithm for agent j

1: **input:** Time point t, description D, specifications $(spec^k)_{k\in C}$, experiences $(e^k_t)_{k\in C}$.
2: Initialize a causal model M by the causal diagram implied by the description D over the set of factors in $(spec^k)_{k\in C}$ and $(e^k_t)_{k\in C}$.
3: Update the "belief" over the mechanisms in M using all values of variables contained in $(spec^k)_{k\in C}$ and the experiences $(e^k_t)_{k\in C}$ (possibly based on additional priors).
4: From the updated M, calculate M^j, the implication of M for agent j.
5: Find action $u(t)$ that is optimal w.r.t. the given task, under M^j.
6: **output:** $u(t)$

- specifications $spec^k$ of the hardware of each agent k
- past experiences (i.e., actions and observations) e^k_t of all agents $k \in C$, consisting of observations $y^k(t)$ and control outputs $u^k(t)$, i.e., $e^k_t = (u^k(0), y^k(0), \ldots, u^k(t), y^k(t))$.
- a description D (e.g., a physical or causal model or collection of such models) consisting of (1) a set of *independence statements*[9] and (2) a set of dependence statements, possibly including specific information on the shape of the dependence, w.r.t. the factors contained in the specifications $spec^k$ and the experiences e^k, for $k \in C$. Potentially, the various independence and dependence information pieces could come from different sources, say targeted experiments as well as general prior knowledge. We assume the dependence structure (including the precise shapes of the dependences) to be time-invariant.

Relation to general problem. This scenario captures certain aspects of Example 3 in that some higher-level information in the form of the description D is available. While here, as a first step, we only consider mathematical models, in the future one could also imagine to include informal but well-structured models and descriptions (in simple natural language), possibly translating them into formal models as an intermediate step (using, e.g., machine learning). Furthermore, the scenario captures important aspects of Example 1, since we consider the integration of information from certain source self-driving cars for the decision making of a given target car.

6.2 Sketch of a method

We sketch a method for the described scenario in Algorithm 3. It first derives a "global" causal diagram - applying to all agents - from the potentially heterogeneous description D (line 2). Then (line 3) the causal conditionals of M, which are not determined by D, are inferred from the given hardware specifications as well as the experiences gathered by all agents up to time t. Last, based on the hardware specifications of agent j, the implications of M for agent j are calculated (line 4) and the optimal action under these implications is performed (line 5).

6.3 A toy example

Let us illustrate how the method proposed in Section 6.2 works in a concrete toy scenario. The core intuition is that while some details

[9] Clearly, independence assertions are central to integration of information: only based on statements of the form "Y is more or less independent of all factors that potentially will be included, except for this and this small set" it seems possible to rigorously (automatically) reason about integration.

of the dynamics of self-driving cars may vary between different cars, they can still share information on say the road conditions (friction, drag, etc.) at certain positions y, or the like.

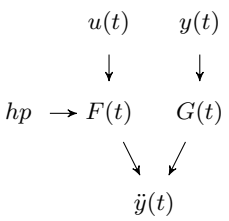


Figure 3: Sketch of the causal diagram H. The power hp influencing only $F(t)$ implies that knowledge on the mechanism f_G for $G(t)$ can be transferred between two cars even if they differ in hp.

Specific scenario. We consider two simplified self-driving cars, i.e., $C = \{1, 2\}$, where we assume the observation $y^k(t)$ (or $y(t)$ if we refer to a model for both cars) to be the car's position. We consider the following concrete instances of the information sources listed in Section 6.1. (Note that, for the sake of simplicity, we assume all mechanisms to be deterministic, such that we can model them by functions $f_{...}$ instead of conditional distributions $p(\dots|\text{pa}_{...})$, although the latter would be more general, see Section 2.)

- The specification for car k is given by its power (e.g., measured in horse powers), i.e., $spec^k = hp^k$.
- The experience of car k consist of all past position-action pairs of both cars, i.e., $e_t^k = (u^k(0), y^k(0), \dots, u^k(t), y^k(t))$.
- The description D consists of three elements:
  - an engineer contributes the equation $F(t) = f_F(u(t), hp)$ for the engine of the cars[10], where $F(t)$ is the force produced by the engine (incl. gears), and f_F a known function,
  - a physicist contributes the equation $\ddot{y}(t) = \frac{1}{m}(F(t)+G(t))$ for the acceleration $\ddot{y}(t)$ of the cars, where $G(t)$ are other forces that affect the cars, such as friction and drag, and m is the known mass (which we assume to be the same for both cars),
  - another physicist produces a set of additional independence assertions, such that altogether the description D implies the causal diagram H depicted in Figure 3.[11]

Implementation of our method. We suggest the following concrete implementation of the crucial part of our method, i.e., line 3 and 4 in Algorithm 3, based on the concrete instances of information available in this toy scenario. Keep in mind that f_G, the function that maps a position y to the corresponding force G at that position and thus models the generating mechanism for $G(t)$, is the only unknown part of M after initializing it by D.

- Line 3: Use the experience $(e_t^k)_{k\in C}$, to infer the function f_G on all positions y visited by either of the cars, based on the equation

$$m\ddot{y} - f_F(u, hp^k) = f_G(y)$$

for all $k \in C$, and the fact that the l.h.s. of this equation as well as y are known for all positions (and accelerations) visited by either of the cars.
- Line 4: calculate "$p(\ddot{y}|do(u), y, hp^j)$", i.e., the effect of control action u of agent j at position y, *for all positions that were visited before by either of the cars*.

6.4 Discussion

The toy example in Section 6.3 shows how in principle the integration of heterogeneous information could help some "target" self-driving car for better decision making in situations not visited by it but by different "source" self-driving cars. It is important to note that all listed information sources were necessary for this: the hardware specifications are necessary to understand $F(t)$, the experience is necessary to infer f_G, and the independence knowledge (hp not affecting $G(t)$) is necessary to transfer the knowledge about the force $G(t)$ on various positions y between the cars. Note that the above scenario cannot be tackled by standard RL approaches since we transfer knowledge between agents of different hardware. Furthermore, methods like LfD or transfer learning (see Sections 4.2 and 4.4) usually do not automatically harness information on hardware specifications of agents.

Based on our preliminary investigation above, it seems that causal models are helpful in that they provide a language in which one can express relevant assumptions and reason about them. However, from a practical perspective, it is not clear if the necessary calculations could not be genuinely done e.g. in classical probabilistic models.

An important question is whether the method sketched in Section 6.2 can be generalized to dependence statements in more natural - but still well-structured - language than equations and causal diagrams.

7 Outlook: future directions and open questions

Here we sketch a potential agenda for future investigations and pose interesting open questions.

7.1 Potential future directions

"Universal representation of physical world". An interesting subject-matter of future research would be a "universal representation" of the physical world - a rich representation to which each information source could be translated, and from which each agent could derive the implications for its specific sensor and actuator configurations. Such a representation would be more efficient than handcrafting mappings for each (new) pair of source and agent individually (as is usually done in e.g. LfD, see Section 4.2), reducing the number of necessary mappings from n^2 to n, where n is the number of agents. One starting point would be representations that are already used to integrate laser or radar scanner data on the one hand with (stereo) video camera data on the other hand in self-driving cars [6]. Another starting point would be the global positioning system (GPS) which is a successful universal representation of position with clear, hardware-independent semantics.

Investigation and classification of the "integration mapping". Another important concept for integration of heterogeneous information could be the *mapping* that transforms a collection of pieces of well-structured heterogeneous information into a model of the current situation or even directly into action recommendations. The study of such a mapping could build on the investigation of related

[10] In particular, the mechanism specification implies non-influence by all other relevant factors.

[11] A better way to describe the physical forces causally might be to replace $\ddot{y}(t)$ in the equations and in D by expressions based on $v(t + \Delta t)$, $v(t)$, and Δt, where $v(t)$ denotes the velocity.

mappings in LfD and transfer learning for agents (see Sections 4.2 and 4.4). Furthermore, parts of such a mapping could be learned (which would be related to machine-learning-based multi-agent systems, see Section 4.3. Generally, it would be interesting to examine the basic conditions under which the integration of heterogeneous information can be beneficial. Note that one way to *classify and order* various sources of information (and the mappings that are necessary to integrate them) would be from "closest to the agent A", i.e., it's own past observations and actions to "most distant", e.g., agent-independent descriptions of the world in simple natural language, as exemplified in Section 3.

Further experiments. Further experiments, with a gradually increasing difficulty (e.g. along the ordering proposed in the previous paragraph), could be performed, e.g. using the platform Malmo (Section 2) to gain a better understanding of integration of information:

1. Agents can observe other agents from a third-person perspective, enabling Example 2 in Section 3.
2. Higher level observations can be provided in the form of natural language (typed chat or external information in the form of natural language), or through artifacts such as maps, sign-posts, symbolic clues, etc.

7.2 Open questions

It would also be interesting to investigate how the following questions could be answered:

- One of the main question which guided our investigation in Section 6 can be cast as follows. While the information relevant to an agent is usually in the form of effects of its actions in certain situations, a lot of knowledge is formulated in non-causal form: for instance street maps at various granularities for self-driving cars. How are these two forms of information related? Is there a standard way to translate between them? Stated differently, how can various forms of information be translated into a model of the *dynamics* of the agent in the world.
- Where is the boundary between additional heterogeneous information and prior knowledge?
- How can the need for information integration be balanced with *privacy* restrictions? For instance, one may imagine cases where the mapping from a source agent's experience to a target agent's action is rather simple in principle, but information collected by the source agent cannot or should not be transmitted to the other, at least not in full.
- How can big databases of information be filtered for useful information, i.e., the information which is correct and relevant for the current environment and task?
- To what extent is the problem of information transfer between two different agents in the "same" environment just a special case of transfer between different environments (by considering the hardware of an agent as part of the environment)?
- How can we reason without having a *"global" model* such as Figure 3? What about interfaces to build global from *"local" models*, describing only say the engine?
- Generally, what are potential theoretical limitations of automated information integration, e.g. in terms of computability?

8 Conclusions

In this paper, we considered the problem of designing agents that autonomously integrate available heterogeneous information about their environment. We investigated how experimentation in simulated environments on the one hand, and causal models on the other, can help to address it. A next step would be to perform more sophisticated experiments, ideally guided by specific problems e.g. from the area of self-driving cars.

9 Acknowledgments

The authors thank Mathew Monfort, Nicole Beckage, Roberto Calandra, Matthew Johnson, Tim Hutton, David Bignell, Daniel Tarlow, Chris Bishop and Andrew Blake for helpful discussions, and the anonymous reviewers for useful hints.

REFERENCES

[1] Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning, 'A survey of robot learning from demonstration', *Robotics and Autonomous Systems*, **57**(5), 469 – 483, (2009).
[2] Karl Johan Aström and Richard M Murray, *Feedback systems: an introduction for scientists and engineers*, Princeton university press, 2010.
[3] Elias Bareinboim and Judea Pearl, 'Causal inference from big data: Theoretical foundations and the data-fusion problem', Technical report, CALIFORNIA UNIV LOS ANGELES DEPT OF COMPUTER SCIENCE, (2015).
[4] David Bignell, Katja Hofmann, Tim Hutton, and Matthew Johnson, 'The Malmo platform for artificial intelligence experimentation', in *IJCAI (to appear)*, (2016).
[5] Lucian Buşoniu, Robert Babuška, and Bart De Schutter, 'Multi-agent reinforcement learning: An overview', in *Innovations in Multi-Agent Systems and Applications-1*, 183–221, Springer, (2010).
[6] Andreas Geiger, Julius Ziegler, and Christoph Stiller, 'Stereoscan: Dense 3d reconstruction in real-time', in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, pp. 963–968. IEEE, (2011).
[7] Andreas Holzinger, 'Interactive machine learning for health informatics: when do we need the human-in-the-loop?', *Brain Informatics*, **3**(2), 119–131, (2016).
[8] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, et al., 'Human-level control through deep reinforcement learning', *Nature*, **518**(7540), 529–533, (2015).
[9] Reza Olfati-Saber, Alex Fax, and Richard M Murray, 'Consensus and cooperation in networked multi-agent systems', *Proceedings of the IEEE*, **95**(1), 215–233, (2007).
[10] J. Pearl, *Causality*, Cambridge University Press, 2000.
[11] Judea Pearl and Elias Bareinboim, 'Transportability of causal and statistical relations: A formal approach', in *In Proceedings of the Twenty-Fifth National Conference on Artificial Intelligence. AAAI Press, Menlo Park, CA*, pp. 247–254, (2011).
[12] Bob Price and Craig Boutilier, 'Accelerating reinforcement learning through implicit imitation', *J. Artif. Int. Res.*, **19**(1), 569–629, (December 2003).
[13] John F Sowa, 'Knowledge representation: logical, philosophical, and computational foundations', (1999).
[14] P. Spirtes, C. Glymour, and R. Scheines, *Causation, prediction, and search*, MIT, Cambridge, MA, 2nd edn., 2000.
[15] Peter Stone and Manuela Veloso, 'Multiagent systems: A survey from a machine learning perspective', *Autonomous Robots*, **8**(3), 345–383, (2000).
[16] Richard S Sutton and Andrew G Barto, *Reinforcement learning: An introduction*, MIT press, 1998.
[17] Matthew E Taylor and Peter Stone, 'Transfer learning for reinforcement learning domains: A survey', *The Journal of Machine Learning Research*, **10**, 1633–1685, (2009).
[18] Vladimir Vapnik and Akshay Vashist, 'A new learning paradigm: Learning using privileged information', *Neural Networks*, **22**(5), 544–557, (2009).
[19] James Woodward, *Making things happen: A theory of causal explanation*, Oxford University Press, 2005.

Analysis of Swarm Communication Models

Musad Haque[1], **Christopher Ren**[1], **Electa Baker**[1], **Douglas Kirkpatrick**[2], and **Julie A. Adams**[1]

Abstract. The biological swarm literature presents communication models that attempt to capture the nature of interactions among the swarm's individuals. The reported research derived algorithms based on the metric, topological, and visual biological swarm communication models. The evaluated hypothesis is that the choice of a biologically inspired communication model can affect the swarm's performance for a given task. The communication models were evaluated in the context of two swarm robotics tasks: search for a goal and avoid an adversary. The general findings demonstrate that the swarm agents had the best overall performance when using the visual model for the search for a goal task and performed the best for the avoid an adversary task when using the topological model. Further analysis of the performance metrics by the various experimental parameters provided insights into specific situations in which the models will be the most or least beneficial. The importance of the reported research is that the task performance of a swarm can be amplified through the deliberate selection of a communications model.

1 INTRODUCTION

Animals that live in groups gain reproductive advantages, benefit from reduced predation risks, and forage efficiently through group hunting and the distribution of information amongst group members [18]. The collective behavior of these biological systems, for instance, trail-forming ants, schooling fish, and flocking birds, display tight coordination that appears to emerge from local interactions, rather than through access to global information or a central controller [7]. Numerical simulations based solely on local interaction rules can recreate coordinated movements of biological systems living in groups [2, 10, 15, 17, 24, 29].

Proposed communication models for group behavior in animals include the metric [9], the topological [1, 3], and the visual models [28]. The metric model is directly based on spatial proximity: two individuals interact if they are within a certain distance of one another [9]. Ballerini *et. al.*'s [3] topological model requires each individual to interact with a finite number of nearest group members. The visual model, which is based on the sensory capabilities of animals, permits an individual to interact with other agents in its visual field [28]. The communication model is an important element in collective behavior, because it reveals how information is transferred in the group [28].

The development of communication networks is described as "one of the main challenges" in swarm robotics [16]. Bio-inspired artificial swarms inherit desirable properties from their counterparts in nature, such as decentralized control laws, scalability, and robustness [6]. Robustness in the context of this paper implies that the failure of one agent does not lead to the failure of the entire swarm. Despite the beneficial properties, a poorly designed communication network to an artificial swarm can lead to undesirable consequences, such as the swarm fragmenting into multiple components [16].

The evaluated hypothesis is that the three communication models – metric, topological, or visual – when used by a tasked artificial swarm will affect the swarm's performance. The evaluation analyzes how the communication models impact swarm performance for two swarm robotics tasks: searching for a goal and avoiding an adversary. The findings demonstrate that there is a significant impact of the communication model on task performance, which implies that the task performance of a deployed artificial swarm is amplified through performance-based selection of communication models.

Section 2 provides related work. Section 3 describes the coordination algorithms derived from the biological models. Experiments are presented in Sections 4 and 5, and an overall discussion with concluding remarks is provided in Section 6.

2 RELATED WORK

Comparative evaluations of the different swarm communication models can be grouped into three fields: biology [3, 28], physics [4, 26], and computer science [14].

Prior research compared the communication models to identify which model best explains the propagation of information within biological species. Stranburg-Peshkin *et al.* [28] reported that for golden shiners, *Notemigonus crysoleucas*, the visual model best predicts information transfer within the school. The Metric and topological models were compared for flocks of European starlings, *Sturnus vulgaris* [3], and the topological model most accurately described the starlings' information network. The experiment compared the cohesion of simulated swarms using the topological and metric models, and the topological model generated more cohesive swarms [3].

Physics-based approaches compared the metric and topological models and presented the resulting system properties. Specifically, Shang and Bouffanais [26] presented results on the probability of reaching a consensus. Barberis and Albano [4] analyzed the difference in group orders (alignment and moment) that arise when using the metric and topological models.

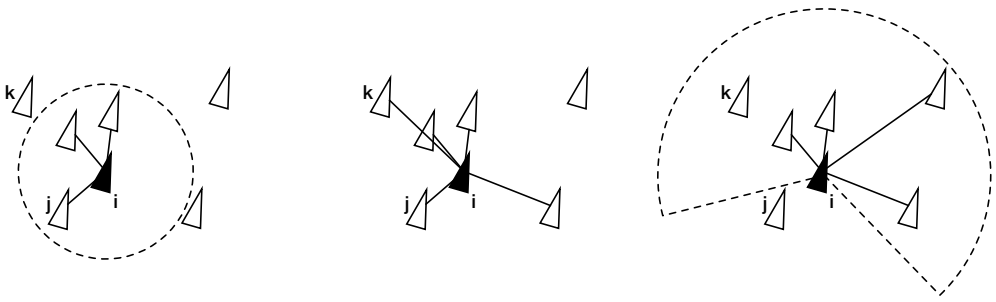
Computer science results include evaluating the metric and topological models in the context of human-swarm interaction [14]. The human steered the swarm by manipulating a leader agent that directly influenced other swarm members. It was determined that a human can more easily manage a swarm using the topological model.

The presented evaluation appears to be the first to compare the metric, topological, and visual models for tasks on artificial agents.

3 COORDINATION ALGORITHMS

The agents are modeled as $2D$ self-propelled particles. A self-propelled particle is controlled through updates to its velocity heading, which in turn affects the particle's position [10, 13, 29].

[1] Vanderbilt University, USA, email: musad.a.haque@vanderbilt.edu
[2] Michigan State University, USA



(a) Metric (b) Topological (c) Visual

Figure 1. Agents (triangles) are shown in relation to the focus agent (filled triangle), labeled i. The communication links from agent i to its neighbors are represented with lines. (a) Metric: Agent k is at a distance greater than d_{met} (dashed circle) from agent i; (b) Topological: n_{top} is set to 5; (c) Visual: The visual range of agent i is shown (dashed sector), where agent j is in agent i's blindspot and agent k is occluded from agent i by another agent.

The artificial agents are indexed 1 through N, where N is the number of agents in the swarm. If there is a communication link from agent $i \in \{1, \ldots, N\}$ to agent $j \in \{1, \ldots, N\}$, where $i \neq j$, agent j is a neighbor of agent i. The neighbor set of agent i, denoted by $\mathcal{N}_i(t)$ is the collection of all the neighbors of agent i at time t.

The coordination of the swarm agents is designed through a multi-level coordination algorithm. At the higher level of abstraction, an agent's neighbors are determined by the communication model. Thus, for agent i, the communication model constructs the set $\mathcal{N}_i(t)$ at each time t. At the lower abstraction level, agents only interact with their neighbors and the nature of this interaction is governed by three rules: repulsion, orientation, and attraction. The rules are based on Reynolds's rules for boids (see [24]), which are similar to the biological swarm literature (e.g., [2]).

Each agent's zones of repulsion, orientation, and attraction are centered at the agent's position and are parameterized through the radii r_{rep}, r_{ori}, and r_{att}, respectively, where $r_{rep} < r_{ori} < r_{att}$. The zones are represented as circles in the $2D$ case.

The heading of each agent $i \in \{1, \ldots, N\}$ is updated as follows: 1) Veer away from all agents in $\mathcal{N}_i(t)$ within a distance r_{rep}, 2) Align velocity with all agents in $\mathcal{N}_i(t)$ that are between a distance of r_{rep} and r_{ori}, and 3) Remain close to all agents $j \in \mathcal{N}_i(t)$ that are between a distance of r_{ori} and r_{att} [19, 24].

3.1 Communication models

The metric model uses a single parameter d_{met} that represents a distance measure. All agents within a distance d_{met} from agent i are i's neighbors, as shown in Figure 1(a). Due to the symmetric nature of the model, if $j \in \mathcal{N}_i(t)$, then $i \in \mathcal{N}_j(t)$. A stochastic version of this model was developed to analyze starling data [5]. The analyzed models assign neighbors in a deterministic manner.

The topological model is characterized by n_{top}, measured in units of agents. $\mathcal{N}_i(t)$ is the set containing the n_{top} nearest agents from agent $i \in \{1, \ldots, N\}$. Zebrafish, *Danio rerio*, have $3-5$ topological neighbors [1], and starlings coordinate, on average, with the nearest $6-7$ birds [3]. Figure 1(b) depicts the neighbors of agent i, with n_{top} set to 5.

A sensing range, a blindspot, and occlusion are used to describe the visual model [28]. Agent j is a neighbor of agent i, if three conditions are met: 1) The distance between the two agents is less than d_{vis}, 2) Agent j is not in agent i's blindspot, and 3) The line-of-sight between the agents is not occluded by another agent or object in the environment. A blindspot emerges because the agent's sensing range is characterized by an angle $\pm\phi$ from its heading [10, 13]. Figure 1(c) depicts agent i's sensing range, with ϕ set to $2\pi/3$ radians.

The particular choices made for the values of d_{met}, n_{top}, d_{vis}, and ϕ can be characterized as inheriting from the "descriptive agenda" of multi-agent learning [20, 27]. The goal in the descriptive agenda is to model the underlying phenomenon from the social sciences (biological swarm communication models). The biological swarm literature provides parameter values that are used to compare the different communication models on tasked artificial swarms. d_{met} was set to r_{att} for metric model experiments (e.g., [2, 9]). The visual model experiments set an agent's d_{vis} to half the size of the diagonal of the world with $\phi = 2\pi/3$ radians [10, 28]. $n_{top} \in \{5, 6, 7, 8\}$ for the topological experiments, allowing some variability, while remaining close to what was observed in nature [3].

The novelty is the comparative evaluations of the different communication models that are *solely* based on the biological swarm literature; hence, strictly inheriting from a descriptive agenda. Traditional artificial swarm communication models do not typically mimic the three communication models (e.g., [8]). Although, perception-based models that rely on line-of-sight communication, such as a swarm of foot-bots responding to light sensors, is a variant of the visual model [12]. As such, one potential application is to serve as a guide for hardware selection.

4 THE SEARCH FOR A GOAL EXPERIMENT

4.1 Experimental design

All experiments were conducted using the `Processing` open-source programming language on a 8GB, 2.6GHz Intel Core i5 Macbook Pro. The body length, BL, of each agent was set to 2 pixels. The size of the world was 600×600 pixels.

The communication model is the primary independent variable: metric, topological, and visual. Additional independent variables were: the number of agents, the number of obstacles, the radius of repulsion, the radius of orientation, and the radius of attraction. The experiment combined each of the primary independent variables with each of the additional independent variables. The resulting pair-wise

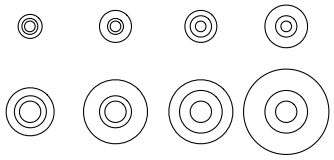


Figure 2. The eight possible interaction zone configurations. The inner-most, middle, and outer-most circles represent the zones of repulsion, orientation, and attraction, respectively.

combinations offers a more comprehensive analysis of the effect of the communication models.

The number of *agents*, N, was 50, 100, and 200. The tuple (r_{rep}, r_{ori}, r_{att}) describes an agent's repulsion, orientation, and attraction zones. The radius of repulsion, r_{rep}, was set to either $5 \times BL$ or $10 \times BL$. The radius of orientation, r_{ori}, was assigned to either $1.50 \times r_{rep}$ or $2.00 \times r_{rep}$, and the radius of attraction, r_{att}, was given a value of either $1.50 \times r_{ori}$ or $2.00 \times r_{ori}$. Designing the interaction zones in this manner results in 2^3 possible tuples with varying (relative) zone sizes, as illustrated in Figure 2.

The search for a goal task included *environmental* obstacles. The number of obstacles, N_{obs}, was 0%, 10%, or 20% of N.

The objective of the artificial swarm during the *search for a goal* is to locate a single goal location[3], the star in Figure 3. The goal area's size is scaled to ensure the swarm is able to fit within the goal area. The world is bounded by a wall that exerts a repulsive force. An agent can sense the goal if it is within r_{att} of the goal area's location. Once an agent locates the goal, it can communicate the location to its neighbors. Agents aware of the goal's location update their headings by equally weighing the desire to travel to the goal and the desire to follow the interaction rules, which was employed by Couzin *et al.* [9] and Goodrich *et al.* [14]. The simulation runs for $1,000$ iterations.

The *percent reached*, denoted by R, determines the number of agents that reached the goal area, expressed as a percentage of the swarm's size, N, at the end of the task.

The *latency*, L, measures the rate of information transfer in the swarm during the task. Specifically, latency represents the number of iterations required for the swarm to transition from a state where at least one agent knows the goal's location to all agents being aware of the goal's location. Degenerate cases are processed by setting the latency to the maximum possible duration, $1,000$ iterations, Based on this definition, the simulator did not influence this metric.

The clustering coefficient is the fraction of pairs of a swarm agent's neighbors that are neighbors with each other [11]. The coefficient ranges from 0, where none of the swarm agent's neighbors are neighbors with each other, to 1, where all pairs of a swarm agent's neighbors are neighbors with each other. The *swarm clustering coefficient*, denoted by SCC, averages the clustering coefficients of all swarm agents. A high swarm clustering coefficient implies a dense communication network and redundant information passing between the agents. While calculating the swarm clustering coefficient, the asymmetric nature of the communication links that resulted from the topological and visual models were ignored. Strandburg-Peshkin *et al.* [28] performed the same treatment on directed links when comparing this metric across different communication models for fish data. This metric permits comparison to prior findings.

The three hypotheses for this task are:

1. H_{sg1}: $R_V > R_T > R_M$,
2. H_{sg2}: $L_V < L_T < L_M$, and,
3. H_{sg3}: $SCC_V < SCC_T < SCC_M$.

The subscripts associated with the performance metrics indicate the metric (M), the topological (T) and the visual (V) models.

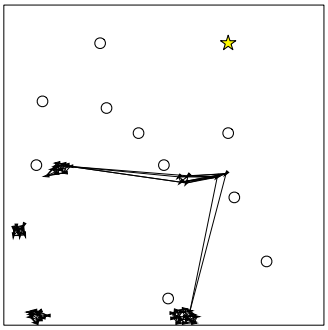


Figure 3. An artificial swarm performing the search for a goal task using the topological model. The center of the goal area is represented by a star, circles represent obstacles, agents are filled triangles, and the lines denote communication links. The trial parameters were: $N = 50$, $N_{obs} = 0.20N$, $r_{rep} = 20$, $r_{ori} = 40$, $r_{att} = 60$, and $n_{top} = 6$.

Hypothesis H_{sg1} assumes that a greater percentage of agents will reach a goal using the visual model and that the metric model will have the lowest percentage reached. The hypothesis is based on the *potentially* long-range sensing capabilities associated with the visual model. Agents favorably oriented and not occluded by obstacles or other agents have a higher chance of communicating with an agent that has located the goal. Moreover, fewer stragglers may arise with the visual and topological models, thus increasing the percent reached. H_{sg1} further assumes that the limit on n_{top}, compared to the range of d_{vis}, allows a greater percentage of agents to arrive at a goal using the visual model, compared to the topological model.

Establishing long-range communication between two agents in the visual model depends on the orientation of the agents and occluding factors. The range d_{vis} may not be a limiting factor in identifying neighbors when positioned in the interior of the swarm. However, any occurrence, regardless of how infrequent, of a long-range link in the network can act as a short-cut for transferring information. As such, H_{sg2} states that information diffuses faster in swarms using the visual and topological models, than with the metric model.

Hypothesis H_{sg3} states that the swarm clustering coefficient will be the highest in the metric model and the lowest in the visual model. Communication links in the metric and topological models are not affected by occlusions, a factor that is expected to yield sparser networks for the visual model.

A trial is defined as a single simulation run for a given selection of parameters, $(N, N_{obs}, r_{rep}, r_{ori}, r_{att})$. Twenty-five trials for each parameter selection were completed. The total number of trials for the search for a goal task was $10,800$: $1,800$ trials for each of the metric and visual models, and $7,200$ trials for the topological model ($1,800$ trials for each of the four values of n_{top}).

[3] Videos of example trials can be found at http://eecs.vanderbilt.edu/research/hmtl/wp/index.php/research-projects/human-swarm-interaction/emulating-swarm-communications/

4.2 Results

The Anderson-Darling test for normality indicated that all performance metrics: percent reached (A=431.01, p<0.001), latency (A=621.88, p<0.001), and swarm clustering coefficient (A=162.72, p<0.001) were distributed normally. An analysis of variance (ANOVA) by n_{top} did not find a significant difference for the topological model's performance. Without loss of generality, the topological trials with $n_{top} = 7$ are used in the reported ANOVAs.

The topological and visual models had virtually identical mean **percent reached**, as reported in Table 1. The ANOVA found that model type had a significant impact on the percent reached (F(2,5398)=83.91, p<0.001). A Fisher's LSD test investigated the pair-wise differences. There was no significant difference between the visual and topological models, and the metric model had a significantly lower percent reached compared to the other models.

Table 1. The search for a goal task descriptive statistics by models. The best means are in bold. (The percent reached, latency, and swarm clustering coefficient are represented by R, L, and SCC, respectively.

Model	Statistic	R	L	SCC
Metric	Mean	27.68	637.79	0.95
	Median	0.00	1000.00	0.95
	Std. Dev.	41.60	471.73	0.03
Topological	Mean	**39.08**	864.99	0.62
	Median	34.00	1000.00	0.62
	Std. Dev.	31.75	290.20	0.06
Visual	Mean	**41.10**	**438.73**	**0.31**
	Median	22.00	31.00	0.33
	Std. Dev.	42.56	487.99	0.07

All data was further analyzed by the number of agents, number of obstacles, and the radii of repulsion, orientation, and attraction. ANOVAs showed significant interactions between the communication models and the number of agents (F(2,5398)=11.26, p<0.001), the number of obstacles (F(2,5398)=8.85, p<0.001), and the radius of attraction (F(2,5398)=2.52, p=0.043). No significant interactions were found for the radii of orientation and repulsion.

Fisher's LSD test showed that for N=50, there was no significant difference between the visual and topological models. The mean percent reached was the highest when N=100 using the topological model. The visual model had the highest mean percent reached for N=200. The percent reached for the metric model was significantly different compared to the other models across all values of N.

The mean percent reached for all the models decreased as additional obstacles were included, as shown in Figure 4(a).

At r_{att}=22.50 there was no significant difference in percent reached for the metric and visual models, see Figure 4(b). The metric model's mean percent reached was significantly higher at r_{att}=22.50 compared to r_{att}=80.

The means are susceptible to the influence of outliers, thus the median values are also reported as a central tendency measure to better assess the performance of the communication models. Further, the interquartile ranges provide additional insights beyond the means.

The median for the metric model's percent reached was 0 for the overall results (see Table 1), which was much lower than the mean. The metric model's median was 0 for most of the parameters and their associated values. The exception being the largest value of N, the obstacle-free trials, the smallest values for the radii of repulsion and orientation, and the two smallest values for the radius of attraction. The median was typically below 10 for those cases, and less than 40 for the obstacle-free trials.

The visual model's third quartiles were at least 95 and mostly 100, except for when N=100, N_{obs}=0.2, and for the smallest values of the radii of orientation and attraction. The high third quartiles indicates that the fourth quartile, or the top 25% of the visual model trials, and at least one of the third quartile trials, had *all* agents reaching the goal area. The metric model's interquartile ranges had much larger variability than the topological model. Across the various parameters, there was at least one parameter value for which the metric model's third quartile was 100%.

Overall, the visual model's mean **latency** was the lowest, whereas the topological model had the highest mean latency, as presented in Table 1. ANOVA showed that a significant difference existed by communication model (F(2,5398)=449.26, p<0.001). Moreover, pairwise testing with Fisher's LSD test found that latency for all three models were significantly different from each other.

ANOVA found significant interactions by model and the number of agents (F(2,5398)=45.70, p<0.001), number of obstacles (F(2,5398)=40.60, p<0.001), radii of repulsion (F(2,5398)=66.96, p<0.001), orientation (F(2,5398)=28.59, p<0.001), and attraction (F(2,5398)=11.15, p<0.001).

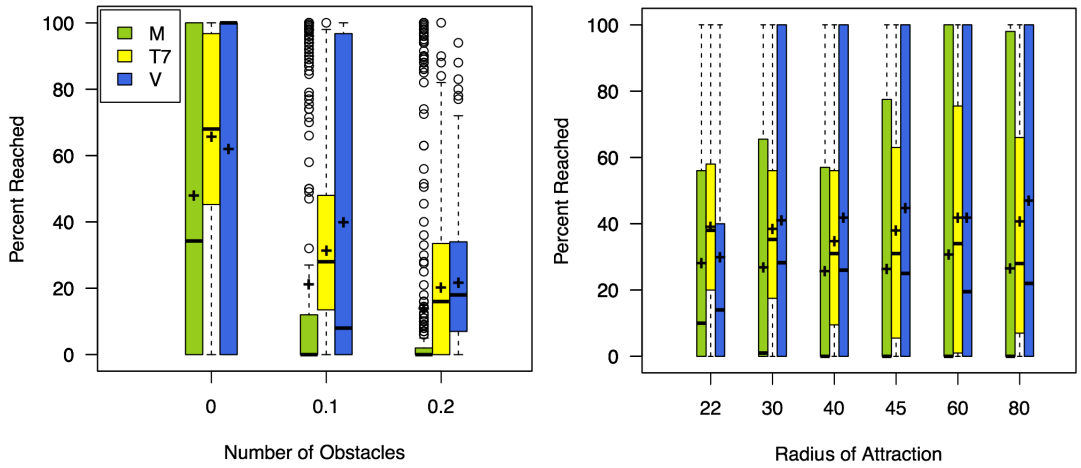
Fisher's LSD test showed that the visual model latency at r_{att}=22.50 was significantly lower than the metric and topological models. At r_{att}=80, the analysis found a significant difference across each of the models, with the metric model's mean latency being lowest (see Figure 4(c)). An identical trend occurs for the lowest and highest radius of orientation.

The metric model's median latency was 1000, for most cases across the number of agents, number of obstacles, and the radii of repulsion, orientation, and attraction. The exceptions occurred for the largest value of the radius of repulsion, the two largest values of the radius of orientation, and the two largest values of the radius of attraction, as shown in Figure 4(c). The median latency was typically 0 for those exceptional cases. Similarly, the topological model's median latency was 1000 across the variables. Additionally, the first quartile of the topological model's latency was 1000 in most cases, and in certain cases, it was at least greater than 400 (see Figure 4(c)). The visual models' median latency was lower than the mean, and was 0 for the largest value of the number of agents, the largest value of the radius of repulsion, and the two largest values of the radii of orientation and attraction (see Figure 4(c)).

The mean **swarm clustering coefficient** was lowest in the visual model and highest in the metric model. An ANOVA showed a significant difference by model (F(2,5398)=1810, p<0.001). Fisher's LSD test found that all the models had significantly different means.

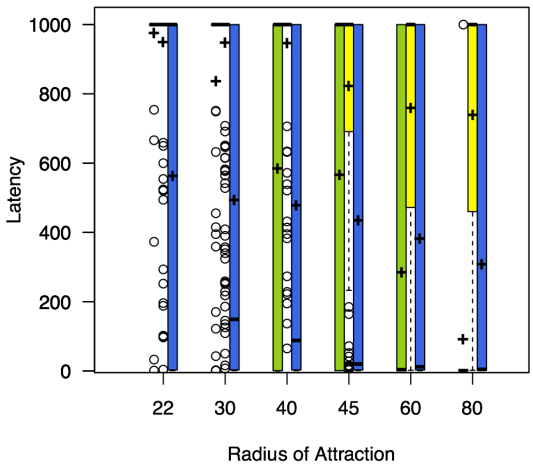
Results from ANOVA showed that for the swarm clustering coefficient, there were significant interactions by model and the number of agents (F(2,5398)=631.50, p<0.001), the number of obstacles (F(2,5398)=2132.00, p<0.001), the radii of repulsion (F(2,5398)=320.90, p<0.001), orientation (F(2,5398)=144.40, p=0.03), and attraction (F(2,5398)=166.40, p<0.001). The results of Fisher's LSD test found a significant pair-wise difference between the models across all variables and associated values.

The median swarm clustering coefficients for all communication models were generally close to the means across all parameters and associated values. The interquartile ranges were typically tight, with only a few cases where the maximum value of one model overlapped with the minimum value of another. Those cases were the smallest number of agents (see Figure 4(d)), the smallest radii of repulsion, orientation, and attraction.

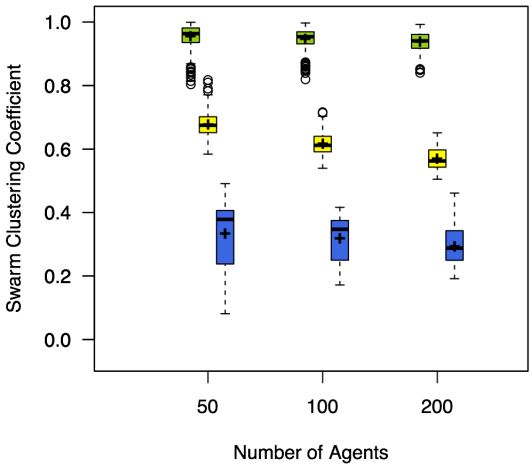


(a) Percent reached.

(b) Percent reached.



(c) Latency



(d) Swarm clustering coefficient.

Figure 4. The search for a goal task performance metrics. Each box plot denotes the first and third quartile of data. The horizontal lines indicate the medians, the crosses represent the means, and the circles show the outlying data. The legend for the plots (b)-(d) can be found in (a), where M, $T7$, and V denote the metric, topological (with $n_{top} = 7$), and visual models, respectively.

4.3 Discussion

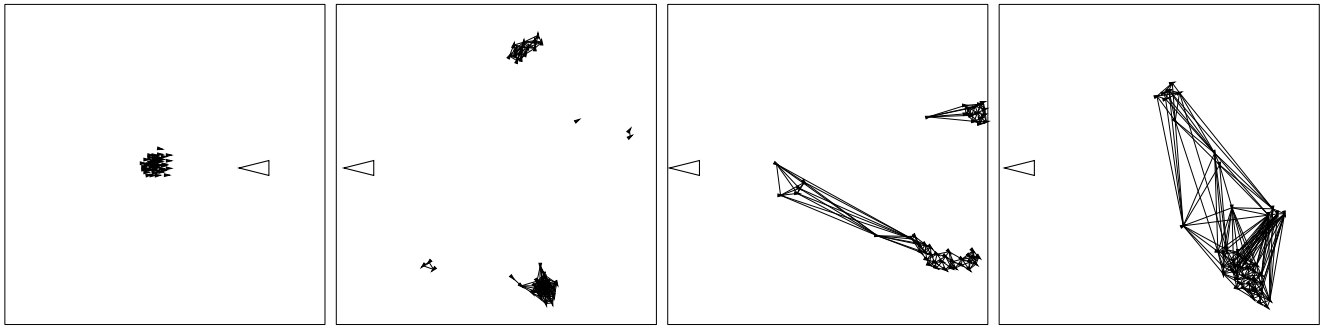
H_{sg1} was partially supported. The topological and visual models outperformed the metric model in reaching the goal area, yet there was no clear difference between the visual and topological models.

The visual model latency was substantially lower than the topological and metric models; however, the metric model outperformed the topological model in terms of the transfer of information. As such, H_{sg2} was also only partially supported. The metric model's bidirectional communication links possibly allowed information to spread faster through the network, compared to the topological model.

Similar to Strandburg-Peshkin *et al.*'s [28] results for fish, the swarm clustering coefficient was lowest with the visual model. The clustering coefficient for fish with the topological model was higher than the metric model, contrary to the findings presented in Table 1. One possible reason for this difference can be attributed to the difference in using collective motion experimental data as opposed to modeling through self-propelled particles.

Based on the general findings, the visual communication model is the best for artificial swarms completing a search for a goal task when fewer redundant connections are desired, because it resulted in virtually the best percent reached, the lowest latency, and the lowest swarm clustering coefficient. A low swarm clustering coefficient can be disadvantageous in noisy environments, which can benefit from redundant communication links. The metric and topological models are preferred for such environments, because of their high swarm clustering coefficients. Furthermore, given a noisy environment and a requirement for only a few agents to reach the goal, then the metric model is preferred. Given the same noisy environment, but a high percentage of agents needed to reach the goal, then the topological model can be used. The tradeoff is the model's high latency.

The analysis by the radius of attraction, which was the value of d_{met}, revealed that the metric and visual models are fundamentally different from one another and the difference does not stem from the visual model's larger communication range. Overall, the visual-based swarms performed better than the metric-based swarms. However, at the lowest value of the radius of attraction (d_{met}=22.50), the metric and visual models had comparable mean percent reached.



(a) Initial configuration (b) Metric at $t = 181$ (c) Topological at $t = 187$ (d) Visual at $t = 183$

Figure 5. An artificial swarm performing the avoid an adversary task under all three communication models. The adversary is denoted by a triangle and swarm's agents are represented by filled triangles. The lines between agents denote communication links. The trial parameters were: (b)-(d) $N = 50$, $r_{rep} = 10$, $r_{ori} = 15$, $r_{att} = 30$; (b) $d_{met} = 30$; (c) $n_{top} = 6$; (d) $d_{vis} = 425$.

Furthermore, for the highest value of the radius of attraction, or d_{met}=80, the latency of the metric model was shown to be significantly lower than the visual, which used a range of d_{vis}=425.

5 THE AVOID AN ADVERSARY EXPERIMENT

5.1 Experimental design

This experiment was performed using the same machine and the experimental parameters, other than N_{obs}, were identical. No obstacles were included in this experiment.

The swarm is required to avoid a predator-like agent[4] during the *avoid an adversary* task, which is modeled through a repulsive force exerted by the adversary on the swarm agents [3]. The swarm (dark mass in Figure 5(a)) is initially aligned facing the predator (triangle in Figure 5(a)). The predator is the same size as the swarm agents and can occlude the visual communication between agents. For illustrative purposes, the rendering of the predator has been increased. The predator (moving in a predefined path) and swarm travel toward each other and when the swarm agents are within r_{att} of the adversary, the predator's repulsive forces affect the swarm agents' heading. Agent positions are initially distributed in an area that is proportional to the swarm's size, N. The predator's starting position is horizontally offset, such that the predator and swarm travel the same distance to meet, regardless of the swarm's size. The effects of the adversary on the swarm are isolated by removing the environmental obstacles and negating the wall's repulsive forces. Each trial runs for 200 iterations.

Dispersion, denoted by D, is measured as the percentage increase of the average agent to agent distance from the start to the end of the trial. The average agent to agent distance has significance in the biological literature and is one of eleven parameters considered when characterizing the emergent properties of fish [23].

A connected component is defined as the largest collection of agents in which any two agents are either connected directly by a communication link or indirectly via neighbors [11]. The *number of connected components*, CCO, is calculated at the end of a trial, and is 1 at the start of a trial.

The *percent isolated components*, represented by I, is the percentage of swarm agents that have no neighbors.

[4] Videos of example trials can be found at http://eecs.vanderbilt.edu/research/hmtl/wp/index.php/research-projects/human-swarm-interaction/emulating-swarm-communications/

The three hypotheses for this task are:

1. H_{aa1}: $D_V < D_T < D_M$.
2. H_{aa2}: $CCO_V < CCO_T < CCO_M$.
3. H_{aa3}: $I_T < I_V < I_M$.

The subscripts indicate the communication models.

Hypothesis H_{aa1} assumes that the metric model will generate swarms with the highest dispersion due to fragmentation. Additionally, the topological and visual models are expected to attract outlying agents back into the main swarm after the adversary's attack, reducing the swarm's dispersion.

Hypothesis H_{aa2} states that swarms using the visual model will fragment into fewer connected components compared to the topological swarms, which will fragment less than the metric-based swarms. The hypothesis is based on the metric model's limited sensing range.

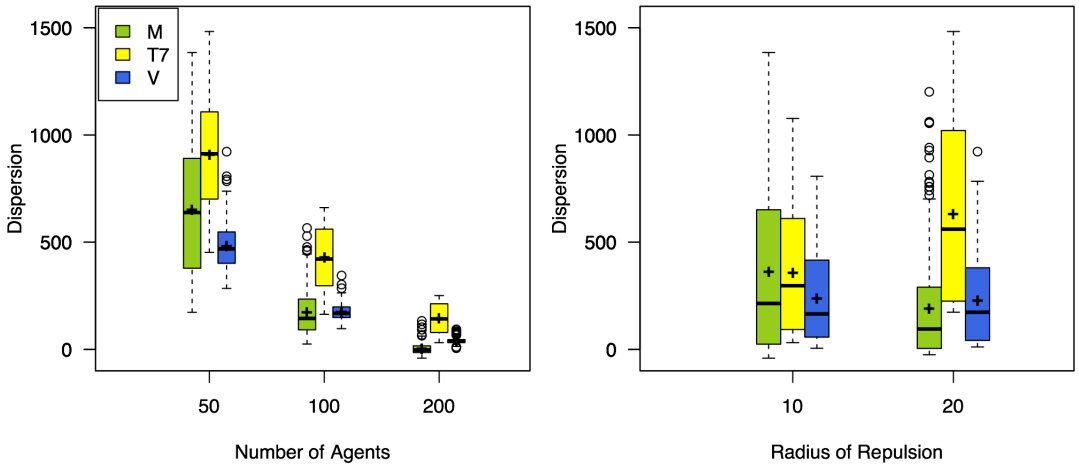
By definition, the topological model does not produce any isolated agents for any $n_{top} \geq 1$. H_{aa3} in relation to the visual and metric models follows the same reasoning underlying H_{aa2}: the metric model's limited sensing range will lead to a higher percentage of isolated agents than the visual model.

The avoid an adversary task experiments were specified similarly to the search for a goal task. The total number of trials for the avoid adversary task was $3,600$: 600 trials for the metric and the visual models, and $2,400$ trials for the topological model (600 trials for each of the four values of n_{top}).

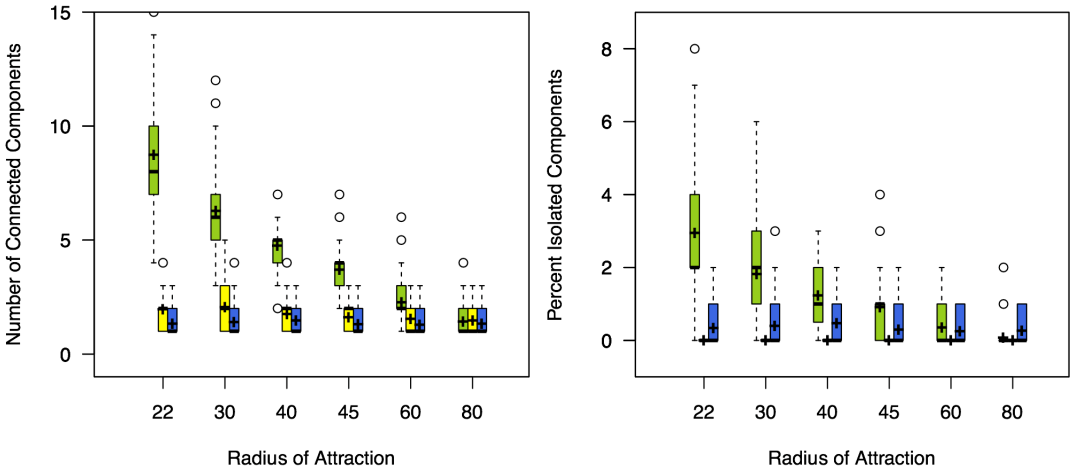
5.2 Results

Dispersion (A=70.16, p<0.001), number of connected components (A=179.90, p<0.001), and percent isolated components (A=296.44, p<0.001) were distributed normally according to the Anderson-Darling test. Similar to the prior experiment, n_{top} was set to 7, as the ANOVA found no significant interactions across the metrics by the topological number. Unlike the previous experiment, a detail account of the medians and quartile ranges are not reported as the medians were generally quite close to the means. Furthermore, the interquartile ranges were tight (see Figure 6).

Overall, **dispersion** was the highest with the topological model and the lowest with the visual model (see Table 2). An ANOVA showed that model type had a significant impact on dispersion (F(2,5398)=562.49, p<0.001). Fisher's LSD test found the mean dispersions to be significantly different across the three models.



(a) Dispersion. (b) Dispersion.



(c) Number of connected components. (d) Percent isolated components.

Figure 6. The avoid an adversary task performance metrics. The legend for the plots (b)-(d) can be found in (a), where M, $T7$, and V denote the metric, topological (with $n_{top} = 7$), and visual models, respectively.

Table 2. The avoid an adversary task descriptive statistics by models. Dispersion, the number of connected components, and the percent isolated components are denoted by D, CCO, and I, respectively.

Model	Statistic	D	CCO	I
Metric	Mean	275.61	4.46	1.19
	Median	144.71	4.00	1.00
	Std. Dev.	334.85	2.78	1.38
Topological	Mean	**493.92**	1.75	**0.00**
	Median	421.50	2.00	0.00
	Std. Dev.	356.32	0.79	0.00
Visual	Mean	232.03	**1.35**	0.33
	Median	168.68	1.00	0.00
	Std. Dev.	196.64	0.58	0.54

ANOVAs revealed that the communication models had significant interactions for the number of agents ($F(2,5398)$=118.32, p<0.001), the radius of repulsion ($F(2,5398)$=363.27, p<0.001), the radius of orientation ($F(2,5398)$=26.15, p<0.001), and the radius of attraction ($F(2,5398)$=9.98, p<0.001).

Dispersion using the topological model was significantly higher compared to the metric and the visual models for all values of N. Fisher's LSD tests showed that the visual model dispersion was significantly lower compared to the metric model at N=50. However, no significant difference between the metric and visual model dispersions was found for the other values of N (see Figure 6(a)).

Fisher's LSD test found that the mean dispersion for the visual model was significantly lower than the metric model at r_{rep}=10, but significantly higher than the metric model at r_{rep}=20, as shown in Figure 6(b). Similarly, as the values of the radii of orientation and attraction increased, the metric model's dispersion decreased to a value significantly lower than the visual model.

ANOVA determined that model type had a significant impact on the **number of connected components** ($F(2,5398)$=1776.23, p<0.001). This metric was significantly different between each of

the communication models, as indicated by the Fisher's LSD test. The visual model had the lowest number of connected components, while metric had the highest, as shown in Table 2.

The ANOVAs found significant interactions by the number of agents ($F(2,5398)=5.25$, $p<0.01$), and the radii of repulsion ($F(2,5398)=772.53$, $p<0.001$), orientation ($F(2,5398)=133.89$, $p<0.001$), and attraction ($F(2,5398)=53.72$, $p<0.001$).

Fisher's LSD test showed that the number of connected components was significantly different between all three models across the number of agents. Visual had the lowest number of connected components, whereas metric had the highest, for all values of N.

The metric model generated fewer connected components as the radius of attraction increased (see Figure 6(c). At $r_{att}=80$, there was no significant difference between the metric and visual models in the number of connected components.

The visual model produced values for the **percent isolated components** that were typically lower than the metric model (Table 2). ANOVA found a significant difference across the communication models ($F(2,5398)=489.78$, $p<0.001$), and Fisher's LSD test found that the models had significantly different means from each other.

ANOVAs indicated that communication models had significant interactions with the number of agents ($F(2,5398)=8.14$, $p<0.001$), the radius of repulsion ($F(2,5398)=232.23$, $p<0.001$), the radius of orientation ($F(2,5398)=32.24$, $p<0.001$), and the radius of attraction ($F(2,5398)=29.28$, $p<0.001$).

Similar to the connected components evaluations, the metric model's percent isolated components decreased as the size of the radii of repulsion, orientation, and attraction (see Figure 6(d)) increased. At, $r_{att}=80$, the metric model's percent isolated components was significantly lower than the visual model.

5.3 Discussion

The topological model produced the highest dispersion compared to the other models. H_{aa1} was only partially supported due to the topological's higher dispersion compared to the metric model.

H_{aa2} was fully supported, as the visual model produced the smallest number of connected components, whereas the metric model generated the highest number of connected components. The visual model's percent isolated components was lower than the metric model, which fully supports H_{aa3}.

A high dispersion in some biological species may serve to confuse a predator from singling out a particular swarm agent [3]. Thus, if a higher dispersion is preferred, the general findings indicate that the topological communication model is the best for the avoid an adversary task, because it offers the highest dispersion, paired with low connected components, and no isolated components. A high dispersion can be disadvantageous if environmental features physically constrain the swarm's movement. The metric and the visual models are preferred for such environments, as they provide a lower dispersion. However, if a task requires a low percentage of isolated components, then the visual model is preferred, otherwise, the metric communication model will suffice.

The results across independent variables did not find the visual model's relatively larger communication range to provide an unfair advantage over the metric model. At the highest radius of attraction, or $d_{met}=80$, there was no significant difference in number of connected components between the metric and visual models, despite d_{vis} being 425.

6 DISCUSSION AND CONCLUSION

The presented research focuses on a general hypothesis that the selection of communication model impacts the swarm's task performance. The general findings demonstrated that there was a significant impact of model type on task performance. Further, the results show that the visual model resulted in the best overall task performance for the search for a goal task, while the best overall performance was achieved with the topological model for the avoid an adversary task. The relevance of this outcome is that the intelligence of a remotely deployed swarm is amplified through the deliberate selection of a communication model. Additional analysis of typical artificial swarm tasks is necessary to fully support the general hypothesis; however, the presented results provide preliminary evidence that artificial swarm design needs to consider the communication model and task pairing in order to optimize the overall swarm performance.

Based on the presented search for a goal and avoid an adversary task results, connections can be made to the biological swarm literature. Couzin *et. al.* [10] showed that the size of the radius of repulsion did not have an effect on the transitions between different swarm movement patterns. Rather, the relative sizes of the radius of orientation to the radius of repulsion and the radius of attraction to the radius of orientation produces the transitions. For instance, simulated swarms rotate in a torus when the ratio of the radius of orientation to the radius of repulsion is relatively low and the ratio of the radius of attraction to the radius of orientation is relatively high. Presented results for the search for a goal task conform to Couzin *et. al.*'s [10] results in relation to the radius of repulsion. The duration of this task (1000 iterations) resulted in trials that demonstrated swarm movement patterns, as found by Couzin *et. al.*. Similar results were expected for the avoid an adversary task; however, were not found due to the task's short duration (200 iterations).

The scope of the reported research does not follow the so-called prescriptive agenda where the values of the model parameters are free design choices [20, 27]; thus, d_{vis} is not varied. This line of inquiry will become necessary when prescribing the communication models to specific platforms, such as the s-bots, which are equipped with proximity and vision sensors [22]. Analyzing the effects of varying model parameters, such as d_{met} and d_{vis} will also be necessary due to differences in the communication ranges across the platforms that will attempt to adopt the models. For instance, the metric model can be realized with omni-directional antennas, as well as infrared LED sensors. The LED range is considerably smaller (10 cm in Kilobots [25]). Similarly, exploring the effects of different values of n_{top} will be useful. The topological model can be implemented using band-limited communication channels [14], and for infrared-based, band-limited platforms, such as the r-one, n_{top} will be inversely related to the maximum communication range [21].

ACKNOWLEDGEMENTS

This material is based upon research supported by, or in part by, the U.S. Office of Naval Research under award #N000141210987.

REFERENCES

[1] Nicole Abaid and Maurizio Porfiri, 'Fish in a ring: spatio-temporal pattern formation in one-dimensional animal groups', *Journal of the Royal Society Interface*, **7**(51), 1441–1453, (2010).

[2] Ichiro Aoki, 'A simulation study on the schooling mechanism in fish', *Bull. Jap. Soc. Sci. Fish.*, **48**(8), 1081–1088, (1982).

[3] Michele Ballerini, Nicola Cabibbo, Raphael Candelier, Andrea Cavagna, Evaristo Cisbani, Irene Giardina, Vivien Lecomte, Alberto Orlandi, Giorgio Parisi, Andrea Procaccini, Massimiliano Viale, and Vladmiri Zdravkovic, 'Interaction ruling animal collective behavior depends on topological rather than metric distance: Evidence from a field study', *PNAS*, **105**(4), 1232–1237, (2008).

[4] Lucas Barberis and Ezequiel V. Albano, 'Evidence of a robust universality class in the critical behavior of self-propelled agents: Metric versus topological interactions', *Phys. Rev. E*, **89**(1), 9–26, (2014).

[5] Nikolai W. F. Bode, Daniel W. Frank, and A. Jamie Wood, 'Limited interactions in flocks: relating model simulations to empirical data', *J. R. Soc. Interface*, **8**, 301–304, (2011).

[6] Eric Bonabeau, Marco Dorigo, and Guy Theraulaz, *Swarm intelligence: from natural to artificial systems*, Oxford University Press, New York, NY, 1999.

[7] Scott Camazine, Jean-Louis Deneubourg, Nigel R. Franks, James Sneyd, Guy Theraulaz, and Eric Bonabeau, *Self-Organization in Biological Systems*, Princeton University Press, Princeton, NJ, 2003.

[8] Christopher M. Cianci, Xavier Raemy, Jim Pugh, and Alcherio Martinoli, 'Communication in a swarm of miniature robots: The e-puck as an educational tool for swarm robotics', in *Proceedings of the 2nd international conference on swarm robotics*, pp. 103–115, (2007).

[9] Iain D. Couzin, Jens Krause, Nigel R. Franks, and Simon A. Levin, 'Effective leadership and decision-making in animal groups on the move', *Nature*, **433**, 513–516, (2005).

[10] Iain D. Couzin, Jens Krause, Richard James, Graeme D. Ruxton, and Nigel R. Franks, 'Collective memory and spatial sorting in animal groups', *Journal of Theoretical Biology*, **218**(1), 1–11, (2002).

[11] David Easley and Jon Kleinberg, *Networks, Crowds, and Markets : Reasoning About a Highly Connected World*, Cambridge University Press, New York, NY, 2010.

[12] Eliseo Ferrante, Ali Emre Turgut, Nithin Mathews, Mauro Birattari, and Marco Dorigo, 'Flocking in stationary and non-stationary environments: A novel communication strategy for heading alignment', in *Parallel Problem Solving from Nature*, pp. 331–340, (2010).

[13] Michael Goodrich, Sean Kerman, Brian Pendleton, and P.B. Sujit, 'What types of interactions do bio-inspired robot swarms and flocks afford a human?', in *Proceedings of Robotics: Science and Systems*, pp. 105–112, (2012).

[14] Michael A. Goodrich, P.B. Sujit, Sean Kerman, Brian Pendleton, and José Pinto, 'Enabling human interaction with bio-inspired robot teams: Topologies, leaders, predators, and stakeholders', Technical Report BYU-HCMI 2011-1, Computer Science Department, Brigham Young University, Provo, USA, (August 2011).

[15] Andreas Huth and Christian Wissel, 'The simulation of the movement of fish schools', *J. theor. Biol.*, **156**, 365–385, (1992).

[16] Andreas Kolling, Phillip Walker, Nilanjan Chakraborty, Katia Sycara, and Michael Lewis, 'Human interaction with robot swarms: A survey', *IEEE Trans. Human-Machine Systems*, **46**(1), 9–26, (2016).

[17] Allison Kolpas, Michael Busch, Hong Li, Iain D. Couzin, Linda Petzold, and Jeff Moehlis, 'How the spatial position of individuals affects their influence on swarms: A numerical comparison of two popular swarm dynamics models', *PLoS ONE*, **8**(3), e58525, (2013).

[18] Jens Krause and Graeme D. Ruxton, *Living in Groups*, Oxford University Press, New York, NY, 2002.

[19] Catarina Maçãs, Pedro Cruz, Pedro Martins, and Penousal Machado, 'Swarm systems in the visualization of consumption patterns', in *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, pp. 2466–2472, (2015).

[20] Shie Mannor and Jeff Shamma, 'Multi-agent learning for engineers', *Artificial Intelligence*, **171**(7), 417–422, (2007).

[21] James McLurkin, Adam McMullen, Nick Robbins, Golnaz Habibi, Aaron Becker, Alvin Chou, Hao Li, Meagan John, Nnena Okeke, Joshua Rykowski, Sunny Kim, William Xie, Taylor Vaughn, Yu Zhou, Jennifer Shen, Nelson Chen, Quillan Kaseman, Lindsay Langford, Jeremy Hunt, Amanda Boone, and Kevin Koch, 'A robot system design for low-cost multi-robot manipulation', in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, (2014).

[22] Francesco Mondada, Luca Maria Gambardella, Dario Floreano, Stefano Nolfi, Jean-Louis Deneubourg, and Marco Dorigo, 'The cooperation of swarm-bots: Physical interactions in collective robotics', *Robotics & Automation Magazine*, **12**(2), 21–28, (2005).

[23] Julia K. Parrish, Steven V. Viscido, and Daniel Grünbaum, 'Self-organized fish schools: An examination of emergent properties', *Biol. Bull.*, **202**, 296–305, (June 2002).

[24] Craig Reynolds, 'Flocks, herds and schools: A distributed behavioral model', *Computer Graphics*, **21**(4), 25–34, (1987).

[25] Michael Rubenstein, Christian Ahler, and Radhika Nagpal, 'Kilobot: A low cost scalable robot system for collective behaviors', in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, (2012).

[26] Yilun Shang and Roland Bouffanais, 'Consensus reaching in swarms ruled by a hybrid metric-topological distance', *Eur. Phys. J. B*, **87**(294), (2014).

[27] Yoav Shoham, Rob Powers, and Trond Grenager, 'If multi-agent learning is the answer, what is the question?', *Artificial Intelligence*, **171**(7), 365–377, (2007).

[28] Ariana Strandburg-Peshkin, Colin R. Twomey, Nikolai W.F. Bode, Albert B. Kao, Yael Katz, Christos C. Ioannou, Sara B. Rosenthal, Colin J. Torney, Hai Shan Wu, Simon A. Levin, and Iain D. Couzin, 'Visual sensory networks and effective information transfer in animal groups', *Current Biology*, **23**(17), R709–R711, (2013).

[29] Tamás Vicsek, András Czirók, Eshel Ben-Jacob, Inon Cohen, and Ofer Shochet, 'Novel type of phase transition in a system of self-driven particles', *Phys. Rev. Lett.*, **75**(6), 1226–1229, (1995).

Metis: system for early detection and prevention of student failure

Damjan Kužnar[1] and **Matjaž Gams**

Abstract. In this paper we first give an overview presentation of a novel system Metis which aims to improve an existing educational process with automatic prediction of student failure and also provide tools in form of smartphone application to apply preventive measures with aim to mitigate a negative outcome. Metis uses artificial intelligence, more specifically machine learning algorithms on data stored in the school information system to identify students with an increased risk of failing a course. Identified students are then referred to an education professional, which will - on the basis of individual consultations with the student - construct an action plan with appropriate measures to improve student's performance. The action plan program can be followed with smartphone application, which will also serve as an interface for transmitting reminders, praise progress and achievements of the objectives of the action plan. The paper then gives a detailed description on the implementation of student failure prediction module, which is the most important and novel component of the Metis system, and performs an objective evaluation where we report on encouraging results regarding the predictive performance of the prediction module.

1 INTRODUCTION

The research presented in this paper is based on the Slovenian educational process and reflects challenges that are in some respect specific, however, we believe that our solution should be applicable in any country or educational process.

Slovenian educational process is faced with relatively high rate of students that do not progress through the educational program as expected. This is most pronounced at secondary schools, which recorded 11.1 % [13] of failed students on average in years between 2008 and 2012, and even more in tetriary (higher) education [12], where 15 % of first year students failed on average in years between 2008 and 2013 and as high as 40% of students never progress to second year. There are many reasons for this: wrong student choice of the program, learning difficulties, personal problems, parents' expectations, social problems, behavioural problems or low self-esteem. If these problems are detected early during the school year there is usually sufficient time for education professionals to offer assistance and successfully mitigate a negative outcome – a student failure.

Current approach for detecting a student's problems in school is depicted in Figure 1. Learning difficulties are usually detected by teachers when a student starts to get a series of negative grades, which results in a significant drop in his/her performance, however teachers feel that this is not systematic enough leaving every teacher to decide on their own if any measures should be taken to help a student to improve his/her school performance. It is usually expected of students to individually seek help from education professional, however, according to the study presented in [10], this happens in rare cases (67 % of education professionals report this happens rarely) or even never (7 % of education professionals). Similarly, parents who detect learning difficulties of their child also seek assistance in rare cases.

When learning difficulties are detected the student usually receives assistance in form of additional studying help from teachers – he/she get more attention during the class or is referred to additional classes. If the student's performance is not improved by these measures, he/she is then referred to an education professional and if necessary to special external educational services.

Usually, an experienced teacher that allocates enough time to follow his/her students performance and is in constant interaction with them and their parents is capable of detecting learning difficulties in time. However, this is becoming more difficult due to recent changes in educational standards in Slovenia, mostly due to the increase of allowed number of students per class - maximum of 28 in primary school and 32 in secondary school.

Informatization of the educational system in Slovenia has significantly increased in the past several years. The switch to IT solutions has already provided several benefits such as easier storage, access and retrieval of all education related data – such as grades, absences, praises, reproofs and more. However, using machine learning methods the usefulness of the recorded data can be further exploited to provide means of automatic detection and prediction of learning difficulties, which is also the goal of the Metis system.

2 RELATED WORK

There already have been several attempts to predict the student academic performance in the literature, which served as a reference when developing our system. The most similar attempt to student failure prediction is reported in [10, 11], where the authors use a genetic programming algorithm and different data mining approaches for predicting student failure at school using real data about 670 high school students from Zacatecas, Mexico. They also present a novel method for inducing both more comprehensible and accurate IF-THEN rules by using the genetic algorithm. However, their student data was more feature rich, more precisely it contained additional information about the student's socioeconomic status, personal, social, family and school factors which they acquired through surveys and resulted in 77 features. Similar work is also reported in [6], where authors use university students' (as opposed to our work on high school students) internim tests data of Artificial Intelligence course and machine learning to predict the students' final performance in the course. Using the learned decision tree the authors then manually extracted the IF-THEN rules and encoded them in semantic web rule

[1] Jozef Stefan Institute, email: damjan.kuznar@ijs.si

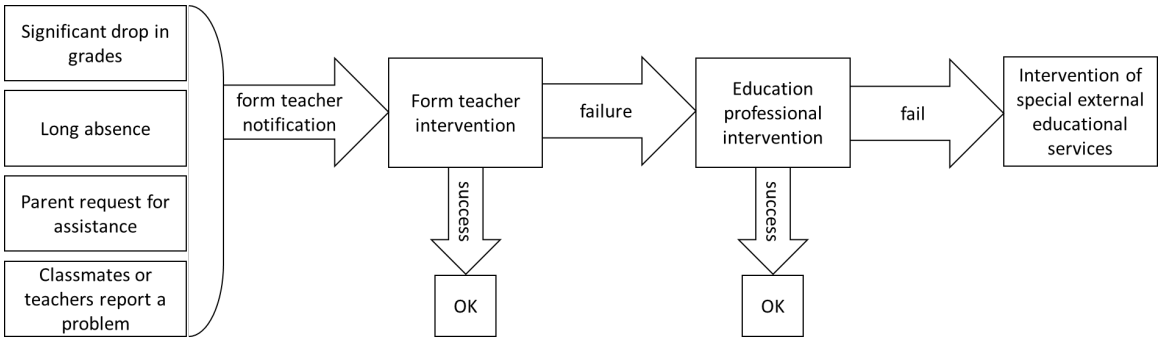


Figure 1. Current process of detection and prevention of student's learning difficulties.

language (SWRL) and used them in their Artificial Intelligence Tutoring System. Similarly, [9] also deals with predicting the academic progress of university student. The authors present a novel three tier architecture that uses educational data mining (EDM) techniques to predict and to identify those who are at dropout risk. When tested on data of three undergraduate engineering courses of one the largest Brazilian public university, they achieved best results using the Naïve Bayes classifier. In [16] authors report on a large-scale study to identify students at risk of not meeting acceptable levels of performance in one state-level and one national standardized assessment in Grade 8 of a major US school district. This significantly differs from our goal of "real time" (on a daily basis) prediction of student's failure. In [17] the authors studied the problem of predicting student performance in an online course. Their goal was to identify at an early stage of the course those students who have a high risk of failing by using the k-nearest neighbor method (KNN) and they report that KNN can predict student performance accurately, and already after the very first lessons. Similar to our approach, they used the developed methods to implement an early warning feature for teachers of the touch-typing course, so they can quickly focus their attention to the students who need help the most.

More general overview of the benefits of using analytics on data gathered by different Learning Management Systems (also called e-learning systems) that are used at several universities in the USA is given in [5]. They report that using learning analytics on LMSs enables faculties to identify at-risk learners and provide interventions, transform pedagogical approaches, and help students gain insight into their own learning. A more comprehensive recent survey on predicting student performance is available in [8].

3 METIS ARCHITECTURE

A high level overview of the Metis system architecture is shown in Figure 2. There are three main components which are depicted as grey boxes. The most important component is the student failure prediction module which is integrated into the existing cloud based school information system, which is used by the majority of the Slovenian schools and has data of nearly 250.000 students (primary and high school). Its main functionalities include extracting student data from the information system, transformation and preprocessing of the data, model training and prediction of student failure on regular basis (e.g. daily or weekly). The results of predictions are sent to the class or form teacher who upon reviewing the prediction results and student's records ultimately decides whether any preventive actions are necessary to avoid the student's class failure. If the teacher decides to act, he/she can use a failure prevention web app that helps in defining and following the action plan that is expected to mitigate the student's negative outcome. The action plan is basically a study agenda that specifies in detail what the student should be doing to improve his/her performance – e.g. on Monday, 21.12.2015, from 15:00 till 16:00 do math homework. This type of help for preventing failure was recognized by education professionals during the system requirements analysis as most helpful for students with learning difficulties, since most of them lack the ability to plan their short and long term studying activities. Once the action plan is defined it can be synchronized with a student's smartphone app (or also tablet) that helps the student to follow the plan on a daily basis. This is mainly achieved through alerts that a planned activity is about to start (by using push notification technology). Once the activity is finished the student is prompted to mark it as completed, which is again synchronized back to the web app for the teacher to follow the student's progress.

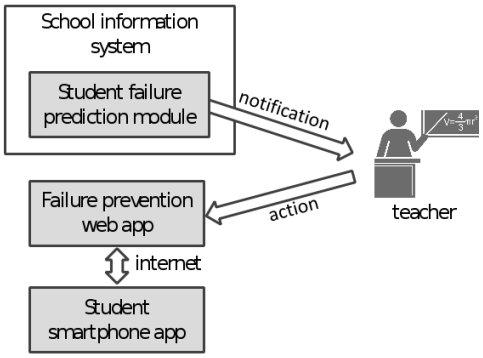


Figure 2. Metis system architecture.

Although the web and smartphone application play the important role in the prevention of the student's failure, the remainder of this paper will focus on the failure prediction module which has the most significant scientific contribution.

4 PREDICTING STUDENT FAILURE

One of the most important requirements of Metis system is the ability to make predictions on a regular (e.g. daily or weekly) basis. This is necessary to detect possible future problems (such as student failure)

early, when there is still time to act. However, this has several implications on how we approached the problem and how the evaluation of the performance was conducted. In general we simulate time dimension when constructing the learning dataset – for each step in the simulation (e.g. each day or week) we made a snapshot of the data in the school information system and constructed a snapshot dataset. After the simulation the snapshot datasets were merged into a single learning dataset, where each instance corresponds to certain point in time.

4.1 Student data

The prediction module has access to student grades throughout the academic year, final semester grades (in Slovenia academic year is split into two semesters) and records of absences. Student grades and absences are basically time series data, therefore we applied feature extraction to obtain features to be used in machine learning. Due to the legal limitation (privacy rights) at the time of writing this paper we had access to data of 692 high school students (all attending the same high school), covering the time span from September 2011 till August 2015. The data contains 107871 individual grades from 18 possible courses, however we decided to focus only on the four most important courses (math, chemistry, Slovene and English) due to the sufficient number of grades to have meaningful evaluation results – they altogether contain 43319 individual grades. With access to all students, we will be able to also include grades from other courses.

4.2 Learning data set

The learning data set is a union of multiple snapshot data sets that are constructed during the simulation of the time dimension – as already noted at the beginning of section 4. Each snapshot dataset therefore has the same set of features that describe data set instances. As noted in subsection 4.1 the data is in form of time series, therefore we extracted several statistics (mean values, standard deviation, etc.) and have done this for the current semester up until the snapshot date and for the entire previous semester. Therefore we have two sets of identical features, once for the current semester and once for the previous semester. We extract the following statistics:

- *grades count* – the number of grades obtained for the given course (separately defined in feature course name)
- *grades maximum* – highest achieved grade
- *grades minimum* – lowest achieved grade
- *grades mean* – average of grades
- *grades std* – standard deviation of grades
- *grades mean ¡ 1.5* – a Boolean value indicating whether the grades mean is lower than 1.5 – in Slovenia, grades are on the scale from 1 (negative) to 5 (the best)
- *negative grades count* – the number of negative grades
- *ratio of negative grades* – defined as negative grades count / sum(grades), where sum(grades) is a sum of all grades (also negative)
- *grades slope* – expresses whether the grades are improving or worsening with respect to the time order of the grades
- *unauthorized absences count* – number of absences that were not authorized by a teacher or school
- *authorized absences count* – number of absences that were authorized by a teacher or school
- *all absences count* – the sum of unauthorized and authorized absences count
- *test absences count* – the number of times the student did not attend class when there was a planned test, either oral or written

In addition to the described statistics we also include six other features which are not semester specific. These include:

- *days remaining* – number of days remaining until the end of the semester
- *grades count difference* – the difference in number of grades between previous and current semester
- *grades average difference* – the difference in grades average between previous and current semester
- *failed previously* – a Boolean value indicating whether the student failed the given course in previous semester
- *course name* – the name of the course to which this instance is referring to
- *student ID* – the anonymized ID of the student, which is not used in learning but is used during the evaluation

The target feature failed is a Boolean value which indicates whether the student has failed the current semester or not. Altogether there are 32 features and a target feature.

To construct the entire data set we used a time interval of one week in our time simulation, which resulted in 256784 instances being generated. Given the nature of the problem, the dataset is heavily unbalanced (as shown in Figure 3), with 11992 (4.67 %) instances have *failure* set to *true* and 244792 (95.33 %) instances have *failure* set to *false*.

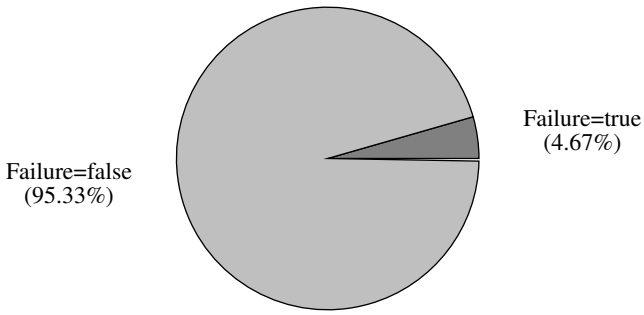


Figure 3. Learning data set imbalance.

5 EVALUATION

During the user requirements analysis the education professionals expressed a requirement related to the performance of the classifier. Their request was that the classifier's precision [15] is greater than some predefined threshold, where the threshold is time dependent. More precisely, this means that the requested threshold increases as we move closer to the semester end. For this purpose the teachers defined two major milestones, one for each semester – 30th November for the 1st semester when their minimum required precision is 30 % and 28th February for the 2nd semester when their minimum required precision is 40 %. Later we added additional milestones for 2nd semester due to insufficient number of recorded grades up until the end of February (2nd semester starts on the 16th January) as can be observed in Figure 4. Summary of all threshold is presented in Table 1.

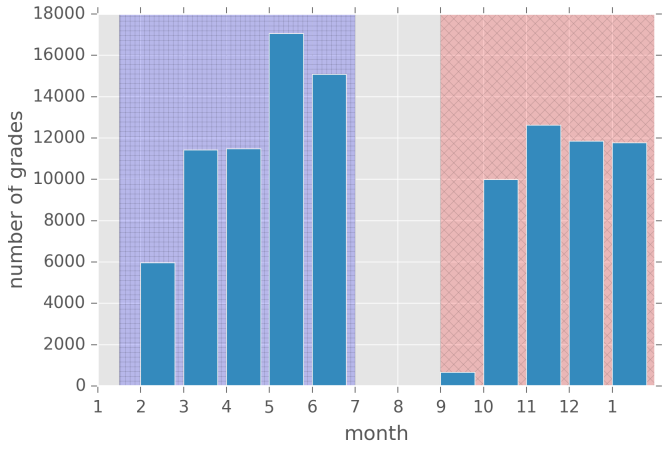


Figure 4. Number of grades by month. Red background represents 1st semester and blue background represents 2nd semester.

Table 1. Time depended mimumum requested precisions thresholds.

	1st Semester		2nd Semester		
Date	30.11	28.2.	31.3.	30.4.	31.5.
Precision	30 %	40 %	50 %	60 %	60 %

5.1 Finding the optimal classifier

Given the minimum threshold requirement the goal was to find a classifier that would have the requested performance. The task of finding it can be formulated as an optimization problem which we tackled with a classifier hyper parameter optimization tool Hyperopt [2], which is implemented in Python and uses Tree of Parzen Estimators [3] algorithm to guide the search. The use of Hyperopt library was mainly directed by the use of Python Scikit-learn [14] library which was used for its implementation of various machine learning methods.

Hyperopt was configured to find classifiers with highest F1 score, which is defined as a geometric mean of precision and recall [1]. The reason for not optimizing on precision is that the precision can usually be easily increased at the expense of a lower recall. Since Hyperopt can optimize only single criteria, whereas our problem is basically a multi-objective one, we decided to optimize F1 score to obtain a variety of classifiers with similar F1 score but different precision/recall values. The outcome of the optimization is therefore a set of non-dominant classifiers [18]. This set allows us to select a classifier which has the best precision/recall tradeoff, more precisely, we can easily select a classifier that has the highest recall rate at the request minimum precision. It also opens up possibilities for setting the threshold for each teacher individually to conform to his/her personal preferences – e.g. some teachers would like higher precision so that they have to deal with less false positives.

5.2 Evaluation methodology

Since the time simulation was used to generate our data set, the information about student grades is shared among data set instances within the same semester. Therefore we cannot use regular cross validation method for classifier evaluation [1] – e.g. if the two instances that correspond to two consecutive simulation steps (snapshots) are the same, since no new grade or absence was recorded, and one instance is placed in train and other in test set then this will not result in fair assessment of performance. To avoid this problem, we implemented a variation of the cross validation method that splits instances based on the student IDs. This means that all instances that correspond to one student are either in test or train data set. Moreover, we ensured that the spits were stratified [7], meaning that train and test data sets contain the same proportion of failed student, where a failed student is defined as a student that had at least one failed course in any semester.

We decided to separate the classifier learning for each of the two semesters due to some particularities. During the data exploration we learned that teachers tend to fail students more often in the 1st semester than in 2nd semester (see Figure 5). We can also see that in 1st semester the teacher mostly give final semester grades to failed student and not to students who do not fail. To overcome this we added additional rule to our simulation for generating data set which assigns failed = false (target feature) if the student does not have a semester grade recorded and grades average value is grater than or equal to 1.5. Figure 5 also shows a trend of increase in overall number of final grades over the time span from 2011 till 2014. This is due to the fact that the school information system became operational in 2011.

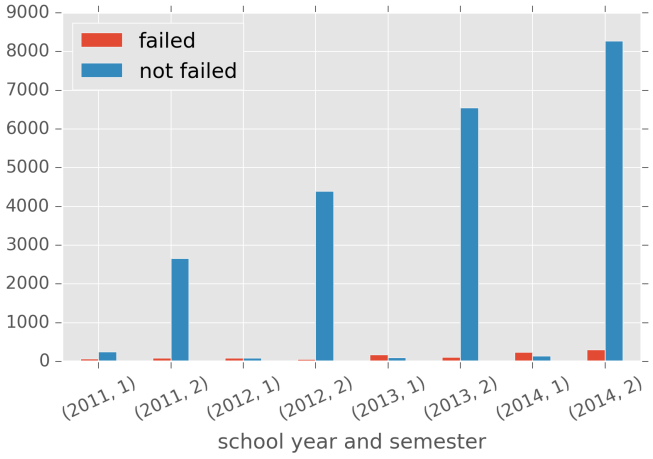


Figure 5. Failed and not failed classes over time.

5.3 Machine learning methods used

The selection of machine learning method was not performed in advance but was rather defined as a part of optimization problem. Therefore we allow for Hyperopt to select among the plethora of machine learning methods that are implemented in the Scikit-learn library, namely: Naïve Bayes, Logistic Regression, Decision Tree, Random Forrest, SVM with different types of kernel (linear, poly,

rbf), Linear Discriminant Analysis (LDA) and Quadratic Discriminant Analysis (QDA). We later removed the SVM classifiers with poly and rbf kernels due to the time complexity of the algorithm and therefore inability to infer a model in reasonable time. For each method we optimized the wide array of parameters, which is too extensive to list them in this paper, however one of the most important that is available to all methods is the *class weight* parameter, which imposes weights to instances based on the value of the target feature. This parameter is very important, due to the unbalanced dataset, and usually significantly impacts the precision/recall trade-off – e.g. setting high weight to class value *true* will result in higher recall and lower precision for the *true* class.

5.4 Results

Results of the hyper parameter optimization can be seen in Figure 6, which is showing a scatter plot of non-dominant classifiers on dimensions of precision and recall. As we can see there is a near linear tradeoff between the precision and recall – increasing one decreases the other. Given the teacher's minimum precision requirement we can select the best classifier from the set – remove the classifiers that do not achieve the minimum required precision and then select the classifier with the highest recall.

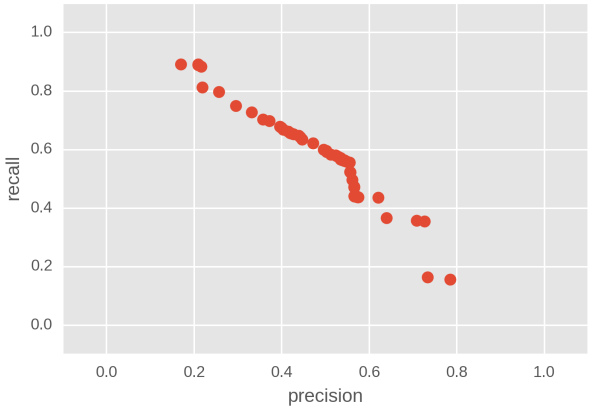


Figure 6. Optimization results – non-dominant classifiers.

Figure 7 gives an insight on how the best classifier that we had chosen from the set of non-dominant classifiers performs over time. We can see that the performance of the classifier (in this particular case a Logistic regression classifier) clearly increases as the predictions come closer to the end of the semester as one would expect. We can also observe a trend that the classifier performance increases over the years as the number of instances increases (especially the ones with failed=true). This increase can be attributed to the fact that machine learning methods are generally capable of inferring better models as the number of instances increases.

In regards to the teacher specified milestones, we can see in Table 2 that we managed to find classifiers that achieve greater than requested precision, while maintaining reasonably high recall and F1 score. The results shown in the table slightly differ from the observed results in Figure 6, since they correspond to a smaller test data set, where only instances from a specified dates are present.

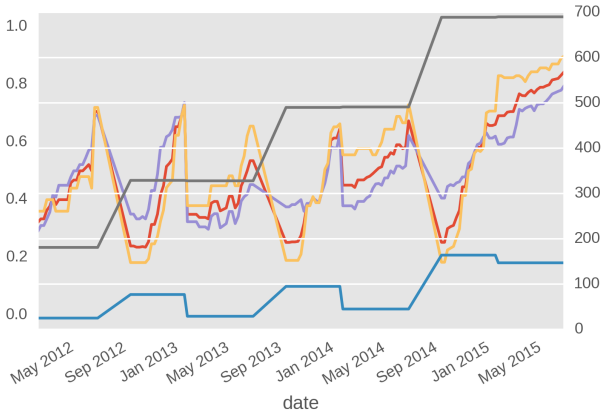


Figure 7. Classifier performance over time. Precision is orange, recall is purple, F1 score is red, total number of instances is gray and number of *failed=true* instances is blue. Precision, recall and F1 score correspond to the left scale and number of instances to the right scale.

Table 2. TABLE II. Achieved precision and recall at defined milestones.

	1st Semester		2nd Semester		
Date	30.11	28.2.	31.3.	30.4.	31.5.
Threshold	30 %	40 %	50 %	60 %	60 %
Precision	54.52 %	56.23 %	58.75 %	61.22 %	60.52 %
Recall	50.30 %	72.81 %	65.68 %	73.71 %	72.81 %

6 CONCLUSION

We presented a novel system Metis for early detection of possible future student failures and prevention through the use of a smartphone application. As demonstrated in section 5 the system performs according to the teacher's needs with respect to the required minimum precision of the detection. The evaluation of the prevention functionality is not in the scope of this paper. Moreover, an objective evaluation will be only possible after extended period of its usage in educational process, after which we will be able to analyze its effect on prevention of student failures.

In our future work we aim to increase the performance of the system detection performance in several ways. Most importantly we need to obtain the data for all the students that are stored in the cloud based school information system. We expect that this alone will increase the performance significantly. More data will also allow us to include other courses into our learning, further increasing the data set size. Another improvement might come from more fine grained separation of the learning problem into smaller problems – e.g. instead of separating the learning into only two semesters we could separate the learning phase into separate months. We also plan to apply Metis system to the higher education institutions in Slovenia, since the methodology presented in this paper should also be applicable to detect problems at university level. However, the type of problems will probably have to be adjusted for the different educational process at Universities - e.g. predicting whether the student will successfully complete the studies.

ACKNOWLEDGEMENTS

This research was funded by the Slovenian Ministry of Education, Science and Sport and the European Union.

REFERENCES

[1] S. Arlot, 'A survey of cross-validation procedures', *Statistics Surveys*, **4**, 40–79, (2010).

[2] James Bergstra, Dan Yamins, and David D. Cox, 'Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms', in *Proceedings of the 12th Python in Science Conference*, pp. 13–20, (2013).

[3] James S. Bergstra, Remi Bardenet, Yoshua Bengio, and Balazs Kegl, 'Algorithms for hyper-parameter optimization', *Advances in Neural Information Processing Systems 24*, 2546–2554, (2011).

[4] European Commision. The bologna process: setting up the european higher education area. http://eur-lex.europa.eu/legal-content/EN/TXT/?uri=URISERV:c11088.

[5] Beth Dietz-Uhler and Janet E. Hurn, 'Using learning analytics to predict (and improve) student success: A faculty perspective', *Journal of Interactive Online Learning*, **12**(1), 17–26, (September 2013).

[6] F. Grivokostopoulou, I. Perikos, and I. Hatzilygeroudis, 'Utilizing semantic web technologies and data mining techniques to analyze students learning and predict final performance', in *Teaching, Assessment and Learning (TALE), 2014 International Conference on*, pp. 488–494, (December 2014).

[7] Ron Kohavi, 'A study of cross validation and bootstrap for accuracy estimation and model selection', in *Proceedings of 14th International Joint Conference on Artificial Intelligence*, pp. 1137–1143, (1995).

[8] A. Dinesh Kumar and Dr. V. Radhika, 'A survey on predicting student performance', *International Journal of Computer Science and Information Technologies*, **5**(5), 6147–6149, (2014).

[9] Laci Mary Barbosa Manhães, Sérgio Manuel Serra da Cruz, and Geraldo Zimbrão, 'Wave: An architecture for predicting dropout in undergraduate courses using edm', in *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, SAC '14, pp. 243–247, New York, NY, USA, (2014). ACM.

[10] Carlos Márquez-Vera, Alberto Cano, Cristóbal Romero, and Sebastián Ventura, 'Predicting student failure at school using genetic programming and different data mining approaches with high dimensional and imbalanced data', *Applied Intelligence*, **38**, 315–330, (April 2013).

[11] Carlos Márquez-Vera, Cristóbal Romero, and Sebastián Ventura, 'Predicting school failure using data mining', in *Proceedings of the 4th Educational Data Mining Conference*, pp. 271–276, (July 2011).

[12] University of Ljubljana. Higher education application and information service archive. http://www.vpis.uni-lj.si/.

[13] Statistical Office of Republic of Slovenia. Demography and social statistics. http://pxweb.stat.si/pxweb/Database/Demographics/Demographics.asp.

[14] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay, 'Scikit-learn: Machine learning in python', *Journal of Machine Learning Research*, **12**, 2825–2830, (October 2011).

[15] D. M. Powers, 'Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation', *Journal of Machine Learning Technologies*, **2**(1), 37–63, (2011).

[16] Ashay Tamhane, Shajith Ikbal, Bikram Sengupta, Mayuri Duggirala, and James Appleton, 'Predicting student risks through longitudinal analysis', in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 1544–1552, (2014).

[17] Tuomas Tanner and Hannu Toivonen, 'Predicting and preventing student failure – using the k-nearest neighbour method to predict student performance in an online course environment', *International Journal of Learning Technology*, **5**(4), 356–377, (2010).

[18] E. Zitzler and L. Thiele, 'Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach', *IEEE Transactions on Evolutionary Computation*, **3**(4), 257–271, (November 1999).

Combining Reinforcement Learning and Quantitative Verification for Agent Policy Assurance

George Mason[1] and **Radu Calinescu**[2] and **Daniel Kudenko**[3] and **Alec Banks**[4]

Abstract. Reinforcement learning (RL) agents converge to optimal solutions for sequential decision making problems based on rewards received after interactions with an uncertain environment. Despite growing success in recent years, RL has limited appeal in applications where unpredictable agent behaviour can have significant unintended negative consequences. To address this limitation, we introduce an *assured reinforcement learning* (ARL) method which combines RL and quantitative verification (QV) to restrict the agent behaviour during and after learning to areas that satisfy safety, reliability and performance constraints specified in probabilistic temporal logic. ARL uses a hierarchical approach that allows verification of the constraints in more complex domains. To this end, ARL builds an abstract Markov decision process (AMDP) that models the problem to solve at a high level, and uses QV to identify a set of Pareto-optimal AMDP policies that satisfy the required constraints. These formally verified *abstract policies* define areas of the agent behaviour space where RL can then take place without constraint violations. We show the effectiveness of our hybrid ARL method through a case study that involved the development of an autonomous agent for a benchmark flag-collection navigation task.

1 Introduction

Reinforcement learning (RL) [39] is a widely-used machine learning technique where an agent explores an initially unknown Markov decision process (MDP) to find an optimal *policy*, i.e. the actions to take in different MDP states in order to maximise the cumulative reward while navigating the MDP. Despite successful adoption in some robotics [35], sensing [25], gaming [6] and control [4] applications, traditional RL cannot be used in mission- and safety-critical applications. In these applications, unpredictable agent actions can lead to mission failure, increased risks to humans or other systems, or violations of legal requirements (e.g. in business domains) [7].

In this paper we present a new RL method that can be used in applications that must satisfy strict safety, reliability and performance constraints. Our *assured reinforcement learning* (ARL) combines traditional reinforcement learning with a formal analysis stage in which the agent exploration is restricted to areas of the original MDP that satisfy the required constraints. ARL carries out this analysis using *quantitative verification* (QV) [21], a mathematically based technique for establishing the reliability, performance and other quality-of-service properties of stochastic systems. Particular advantages of our hybrid ARL are scalability due to a hierarchical approach and convenience of formulating required constraints by using an expressive representation language that has been successfully applied in quantitative verification.

Specifically, ARL supports constraints specified in a variant of probabilistic temporal logic called *probabilistic computation tree logic* (PCTL) [19], and comprises a QV stage and an RL stage. In the QV stage, expert-provided knowledge of the scenario is given in the form of an abstract Markov decision process (AMDP) [30], a common and feasible practice in safety engineering [26, 9, 14]. Compared to the complete MDP to be explored by the RL agent, the AMDP can be assembled with only limited understanding of the problem, and has a significantly reduced state space and a simplified action set [27, 30, 36] that enable efficient analysis which would not have been feasible without this hierarchical approach. Given a set of PCTL constraints, quantitative verification is then used to identify AMDP policies that satisfy all these constraints. Each of these "safe" *abstract policies* resolves some of the nondeterminism of the original MDP, inducing a restricted MDP that the agent explores in the reinforcement learning stage of our ARL method without violating any of the constraints.

As described above, ARL incorporates a set of constraints on the behaviour of a reinforcement learning agent both in the learning process and in the learnt policy. Multiple "safe" abstract policies are typically generated during the QV stage. ARL supports the selection of a suitable abstract policy for the RL stage by retaining only the abstract policies that are *Pareto-optimal* with respect to optimization objectives associated with constraints from the QV stage and/or specified additionally.

Our work contributes to the ongoing research on *safe reinforcement learning* [16]. Thus, ARL complements the existing *constrained optimisation* approaches to safe RL, in which the agent seeks a policy that maximises its obtained reward subject to a set of constraints. To the best of our knowledge, ARL is the first such approach that supports the broad range of safety, reliability and performance constraints that can be formally specified in PCTL [19] extended with rewards [2], and that uses quantitative verification [21] to identify *allowable MDP policies*. In contrast, the existing approaches are typically limited to specifying bounds for the reward obtained by the RL agent or for simple measures related to this reward [1, 8, 11, 17, 33, 34].

The remainder of this paper is organized as follows. Section 2 introduces the technologies that are used in ARL. Section 3 provides an example scenario, based on the benchmark RL flag-collection domain [10] and modified to include an aspect of risk where the application of ARL is necessary. Section 4 outlines the procedure for using ARL, using the example scenario to illustrate the process. Section 5 evaluates ARL through a case study based on the running example.

[1] University of York, UK, email: grm504@york.ac.uk
[2] University of York, UK, email: radu.calinescu@york.ac.uk
[3] University of York, UK, email: daniel.kudenko@york.ac.uk
[4] Defence Science and Technology Laboratory, UK

Section 6 discusses related research, and Section 7 summarizes our results and discusses directions for future work.

2 Background

2.1 Markov Decision Processes (MDPs)

Markov decision processes represent a formalism for modelling sequential decision-making problems [39]. An MDP models an environment in which an autonomous agent can perceive the current state s and select an action a from a set of actions. Performing the selected action a results in the agent transitioning to a new state s' and receiving an immediate numerical reward $r \in \mathbb{R}$. Formally, an MDP is defined as a tuple (S, A, T, R), where:

- S is a set of states;
- A is a set of actions;
- T is a state transition function such that for any $s, s' \in S$ and any action $a \in A$ that is permitted in state s, $T(s, a, s')$ gives the probability of transitioning to state s' when performing action a in state s;
- R is a reward function such that $R(s, a, s') = r$ is the reward received by the agent when action a performed in state s leads to state s'.

A related concept that is central to RL is that of a *policy*. ARL uses *deterministic policies*, i.e. mappings of the form $\pi : S \rightarrow A$ that associate each state $s \in S$ to one of the actions allowed in state s.

When all elements of the MDP are known, the problem can be solved using dynamic programming, e.g. by using the value or policy iteration algorithms. In scenarios where the transition and/or reward functions are unknown a priori, RL needs to be used as described in Section 2.3.

2.2 Quantitative Verification (QV)

QV is a formal verification technique used to establish safety, reliability, performance and other non-functional properties of systems through the analysis of their stochastic models [21, 22]. Unlike techniques like testing and simulation, QV uses efficient algorithms to examine the entire state space of the analysed model, yielding results that are guaranteed to be correct. QV supports the analysis of models including MDPs, Markov chains and probabilistic automata. The analysed properties of these models are specified formally in probabilistic variants of temporal logic. QV is performed using efficient probabilistic model checkers, such as PRISM [24] or MRMC [20].

For the analysis of MDPs, QV labels the model states with *atomic propositions* that specify basic properties of interest that hold in each MDP state, e.g. *success*, *fail* or *retry*. MDPs labelled with atomic propositions enable the QV of properties that express probabilities and temporal relationships between events. For example, QV can verify if the probability of achieving *success* without any *retry* (i.e. of reaching a state labelled *success* without passing through a state labelled *retry*) is at least 0.95. These properties are specified in a probabilistic temporal logic called probabilistic computational tree logic (PCTL) [19]. Given a set of atomic propositions AP, a *state formula* Φ and a *path formula* Ψ in PCTL are defined by the grammar:

$$\begin{array}{l} \Phi ::= true \mid a \mid \neg\Phi \mid \Phi_1 \wedge \Phi_2 \mid \mathrm{P}_{\bowtie p}[\Psi] \\ \Psi ::= \mathrm{X}\Phi \mid \Phi_1 \,\mathrm{U}\, \Phi_2 \mid \Phi_1 \,\mathrm{U}^{\leq k}\, \Phi_2 \end{array}, \qquad (1)$$

where $a \in AP$, $\bowtie \in \{<, \leq, \geq, >\}$, $p \in [0, 1]$ and $k \in \mathbb{N}$; and a PCTL *reward state formula* [23] is defined by the grammar:

$$\Phi ::= \mathrm{R}_{\bowtie r}[\mathrm{I}^{=k}] \mid \mathrm{R}_{\bowtie r}[\mathrm{C}^{\leq k}] \mid \mathrm{R}_{\bowtie r}[\mathrm{F}\Phi] \mid \mathrm{R}_{\bowtie r}[\mathrm{S}], \qquad (2)$$

where $r \in \mathbb{R}_{\geq 0}$. State formulae include the logical operators $\wedge$ and $\neg$, which allow the formulation of disjunction ($\vee$) and implication ($\Rightarrow$).

The semantics of PCTL are defined with a satisfaction relation $\models$ over the states and paths of an MDP (S, A, T, R). Thus, $s \models \Phi$ means Φ is satisfied in state s. For any state $s \in S$, we have: $s \models true$; $s \models a$ iff s is labelled with the atomic proposition a; $s \models \neg\Phi$ iff $\neg(s \models \Phi)$; and $s \models \Phi_1 \wedge \Phi_2$ iff $s \models \Phi_1$ and $s \models \Phi_2$. A state formula $\mathcal{P}_{\bowtie p}[\Psi]$ is satisfied in a state s if the probability of the future evolution of the system satisfying Ψ satisfies $\bowtie p$. For an MDP path $s_1 s_2 s_3 \ldots$, the "next state" formula X Φ holds iff Φ is satisfied in the next path state (i.e. in state s_2); the *bounded until* formula $\Phi_1 \,\mathrm{U}^{\leq k}\, \Phi_2$ holds iff before Φ_2 becomes true is some state s_x, $x < k$, Φ_1 is satisfied for states s_1 to s_{x-1}; and the *unbounded until* formula Φ_1 U Φ_2 removes the constraint $x < k$ from the bounded until. For instance, the PCTL formula $\mathrm{P}_{\geq 0.95}[\neg retry \,\mathrm{U}\, success]$ formalises the constraint 'the probability of reaching *success* without *retry* is at least 0.95' from the earlier example.

The notation $\mathrm{F}^{\leq k}\Phi \equiv true\mathrm{U}^{\leq k}\Phi$, and $\mathrm{F}\Phi \equiv true\mathrm{U}\Phi$ is used when the first part of a bounded until, and until formula, respectively, are $true$. The reward state formulae (2) express the expected cost at timestep k ($\mathcal{R}_{\bowtie r}[\mathrm{I}^{=k}]$), the expected cumulative cost up to time step k ($\mathcal{R}_{\bowtie r}[\mathrm{C}^{\leq k}]$), the expected cumulative cost to reach a future state that satisfies a property Φ ($\mathcal{R}_{\bowtie r}[\mathrm{F}\Phi]$), and the expected steady-state reward in the long run ($\mathcal{R}_{\bowtie r}[\mathrm{S}]$).

Finally, probabilistic model checkers also support PCTL formulae in which the bounds '$\bowtie p$' and '$\bowtie r$' are replaced with '$=?$', to indicate that the computation of the actual bound is required. For example, $\mathrm{P}_{=?}[\mathrm{F}^{\leq 20} success]$ expresses the probability of succeeding (i.e. of reaching a state labelled *success*) within 20 time steps.

2.3 Reinforcement Learning (RL)

An RL agent starts with no knowledge of the environment, and must learn about it by *exploration*, i.e. by selecting initially arbitrary actions while moving from one state of the unknown MDP to another. By receiving rewards after each state transition, the agent learns about the quality of its action choices. The agent stores this knowledge it gains about the quality of a state-action pair (s, a) in the form of a *Q-value*, denoted $Q(s, a)$. Updates to Q-values are done using a temporal difference learning algorithm, such as Q-learning [38]. The Q-learning algorithm has the update formula:

$$Q(s, a) \leftarrow Q(s, a) + \alpha[r + \gamma \max_{a'} Q(s', a') - Q(s, a)], \qquad (3)$$

where α is the learning rate and γ is the discount factor. Through these updates, information about rewards in the environment are propagated over the state-action pairs.

With these Q-values the agent can exploit the knowledge it has already learned: when revisiting a state, instead of randomly selecting an action to perform, it can select an action based on a pre-defined policy. An example of such a policy is the *ϵ-greedy policy*, where with probability ϵ the agent will act randomly and with probability $1-\epsilon$ if will select the highest-value action it knows about [39].

Provided that each state in the MDP is visited an infinite number of times and there are conditions on α the algorithm is guaranteed to converge to an optimal policy. In practice, a finite number of learning iterations, known as learning *episodes*, is typically sufficient to obtain a policy that is sufficiently close to the optimal policy.

2.4 Abstract MDPs (AMDPs)

An AMDP is a high-level representation of an MDP in which multiple states of the MDP are aggregated (e.g. based on their similarity [27]). Additionally, the low-level actions of the MDP are replaced by temporally abstract *options* [36]. For example, instead of an agent performing a sequence of stepwise movements to transition through a series of Cartesian coordinates from location A to enter location B, in an AMDP each location would be a single state and the option would simply be to "move" from one location to the other. An AMDP is orders of magnitude smaller than its MDP counterpart, can often be assembled with significantly less knowledge about the environment, and can be solved and reasoned about much faster [30].

Given an MDP (S, A, T, R) and a function $z : S \to \bar{S}$ that maps each state $s \in S$ to an abstract state $z(s) \in \bar{S}$ such that $z(S) = \bar{S}$, an AMDP can be formally defined as a tuple $(\bar{S}, \bar{A}, \bar{T}, \bar{R})$, where:

- $\bar{S}$ is the set of abstract states;
- $\bar{A}$ is the set of options;
- $\bar{T}$ is a state transition function such that $\bar{T}(\bar{s}, o, \bar{s}') = \sum_{s \in S, z(s)=\bar{s}} w_s \sum_{s' \in S, z(s')=\bar{s}'} P(s'|s, o)$ for any $\bar{s}, \bar{s}' \in \bar{S}$ and any option $o \in \bar{A}$;
- $\bar{R}$ is a reward function such that, for any $\bar{s} \in \bar{S}$ and any $o \in \bar{A}$, $\bar{R}(\bar{s}, o) = \sum_{s \in S, z(s)=\bar{s}} w_s R(s, o)$,

where w_s is the weight of state s, calculated based on the expected frequency of occurrence of state s in the abstract state $z(s)$ [30].

A *parameterised* AMDP uses parameters to specify which option to perform in each AMDP state [40]. An *abstract policy* fixes the values of all these parameters, and thus resolves the non-determinism of the AMDP, essentially transforming it into a Markov chain since there is a fixed, single option for each state.

3 Running Example

We will illustrate the application of ARL using an extension of the benchmark RL flag-collection mission from [10]. In the original mission, an agent learns to navigate a series of rooms and hallways in order to find and collect flags scattered throughout a building. In our extension, the building is augmented with security cameras that monitor certain doorways between areas. The detection of the agent by a camera results in the capture of the agent and the termination of its flag-collection mission. An illustration of this environment is shown in Figure 1.

The agent is camouflaged, yet there is still a probability that it can be detected. *Unknown to the agent*, the detection effectiveness of the cameras decreases towards the boundary of their field of vision. We represent this by sectioning the camera-monitored doorways into three areas: direct view by the camera, partial view, and hidden. These three areas are associated with decreasing probabilities of detection (Table 1).

Table 1. Detection probabilities when transitioning between areas.

	View Detection Probabilities		
Area Transitions	Direct	Partial	Hidden
HallA ↔ RoomA	0.18	0.12	0.06
HallB ↔ RoomB	0.15	0.1	0.05
HallB ↔ RoomC	0.15	0.1	0.05
RoomC ↔ RoomE	0.21	0.14	0.07

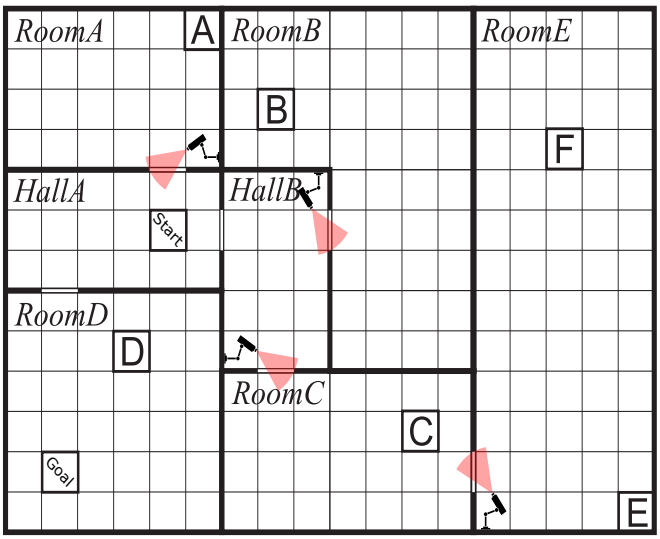


Figure 1. Augmented flag-collection domain, showing the locations of the security cameras and their scope of vision, start and goal positions for the agent and flags to collect A-F.

The only information available to the agent is a list of the rooms, hallways and doorways in the building, and conservative estimates of the detection probabilities for the doorways equipped with a camera. These conservative estimates correspond to the probability of direct-view detection from Table 1.

Suppose that in the real world, where the agent is actually a physical vehicle of some value, the owners of the vehicle wish for the safe return of it. However, they do not want it to behave "too safely" or it will not collect enough flags. Therefore, they specify the constraints from Table 2 for the agent. In this way, the right level of risk can be selected for each instance of the mission. Note that formulating the constraints C_1 and C_2 into a reward function and using standard RL to solve the problem is not possible because an RL agent aims to maximize its reward rather than to maintain it in a specified range.

Table 2. Constraints for the flag-collection mission

ID	Constraint	PCTL
C_1	The probability that the agent reaches the 'goal' area should be at least 0.75	$\mathrm{P}_{\geq 0.75}[\mathrm{F}\ goal]$
C_2	The agent should cumulate a reward greater than 2 when it reaches the 'goal'	$\mathrm{R}_{>2}[\mathrm{F}\ goal]$

4 Method

As shown in Figure 2, ARL takes as input a description of the problem to solve that comprises: (a) incomplete knowledge about the environment (i.e. problem); and (b) the set of constraints $\mathcal{C} = \{C_1, C_2, \ldots, C_n\}$ that must be satisfied by the policy obtained by the RL agent. The incomplete knowledge must contain sufficient information for the *conservative* QV analysis of the environment properties associated with the $n > 0$ constraints. For instance, given the constraint C_1 from Table 2, it is sufficient to know a conservative lower bound for the detection probabilities of the cameras from the

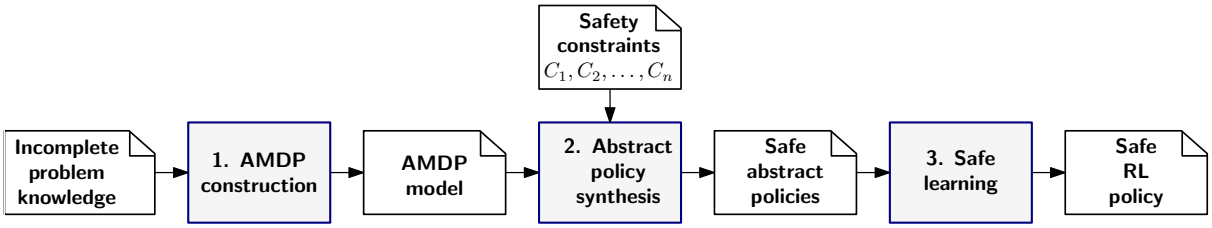


Figure 2. Stages of the ARL method.

flag-collection mission in our running example. Note that the incomplete knowledge about the environment assumed by ARL is necessary: no constraints could be ensured during RL exploration in the absence of any information about the environment.

Under the assumptions detailed above, our ARL method yields a policy that satisfies the constraints $\mathcal{C}$. To this end, ARL employs a process that integrates quantitative verification and reinforcement learning, and comprises the following stages:

1. **AMDP construction** – This ARL stage devises a parameterised AMDP model of the RL problem that supports the QV of the constraints $C_1, C_2, \ldots, C_n$.
2. **Abstract policy synthesis** – This stage generates AMDP policies (i.e. *abstract policies*) that satisfy the problem constraints. These "safe" policies are used to assemble an approximate Pareto-front of abstract policies. The optimisation objectives used to establish the Pareto dominance between different abstract policies are derived directly from the constraints (as described later in Section 4.2) or specified manually.
3. **Safe learning** – In this ARL stage, a suitable abstract policy from the Pareto-front is selected, and translated into state-action constraints for the exploration of the environment by the RL agent. Accordingly, the RL agent obtains an optimal policy that complies with the problem constraints.

4.1 Stage 1: AMDP Construction

In this ARL stage, all features that are relevant for the problem constraints must be extracted from the available incomplete knowledge about the RL environment. This could include locations, events, rewards, actions or progress levels. The objective is to abstract out the features that have no impact on the solution attributes that the constraints $\mathcal{C}$ refer to, whilst retaining as much detail as possible about the key features that these attributes depend on. This ensures that the constructed AMDP is sufficiently small to be analysed using quantitative verification, while also containing the necessary details to enable the analysis of all constraints.

In our running example, the key features are the locations and connections of rooms and halls, the detection probabilities of the cameras and the progress of the flags collected. Instead of having each (potentially unknown) Cartesian coordinate within a room or hall as a separate state, the room or hall as a whole is considered a single state in the AMDP. Also, we only consider the conservative detection probability per camera (which allows a conservative verification of constraint C_1 from Table 2), since the probabilities from Table 1 are unknown to the agent at this stage.

These abstractions reduce the size of the RL MDP, which is unknown to the agent and contains 14,976 states, to just 448 states for the associated AMDP. Note that the number of AMDP states is larger than the number of locations (i.e. rooms and halls) because different AMDP states are used for each possible combination of a location and a number of flags collected so far.

The actions of the full RL MDP should be similarly abstracted. For example, instead of having the cardinal movements at each location of the building from our running example, abstract actions (i.e. *options* – cf. Section 2.1) are specified as simply the movement between locations. Thus, instead of the four possible actions for each of the 14,976 MDP state, the 448 AMDP states have only between one and four possible options each. The m options that are available for an AMDP state correspond to the $m \geq 1$ passageways that link the location associated with that state with other locations, and can be encoded using a *state parameter* that takes one of the discrete values $1, 2, \ldots, m$. The parameters for AMDP states with a single passageway (corresponding to rooms A, B and E from Figure 1) can only take the value 1 and are therefore discarded. This leaves a set of 256 parameters that correspond to approximately 4×10^{99} possible abstract policies.

4.2 Stage 2: Abstract Policy Synthesis

In this ARL stage, a heuristic is used to find abstract policies that satisfy the constraints $\mathcal{C}$ for the AMDP constructed in Stage 1. The process is made easy by the use of the state parameters proposed in the previous section. Thus, each abstract policy can be obtained by assigning suitable values to these parameters. Fixing these parameter values in the AMDP resolves all nondeterminism, and the resulting model (which is a *Markov chain*) can be verified using QV, to establish if the abstract policy satisfies each constraint from $\mathcal{C}$. If it does, the policy is deemed "safe", and is considered for inclusion in an approximate Pareto-optimal set of abstract policies. This set consists of abstract policies that *Pareto-dominate* each other according to a number of optimisation objectives such as probability of success or cumulated reward, where a policy π_A is said to Pareto-dominate another policy π_B iff π_A gives superior results to π_B for at least one objective, and for all other objectives π_A it is at least as good as π_B [28]. A Pareto-optimal set and the associated *Pareto-front* of objective values allow an acceptable trade-off between objectives to be determined a posteriori.

The optimisation objectives used to assess if either of two abstract policies Pareto-dominates the other can be specified manually or can be derived automatically from the constraints $\mathcal{C}$. In the former case, additional PCTL formulae need to be formulated. In the latter case, the PCTL formula for each constraint C_i that specifies a lower bound for an attribute of the RL problem is interpreted as an attribute whose value should be maximised. Conversely, attributes for which upper

bounds are specified in the problem constraints are considered attributes whose value should be minimised.

In our running example, the two constraints for the flag-collection problem specify lower bounds both for the probability that the agent reaches the 'goal' area and for the reward cumulated by the RL agent. Therefore, using automated selection of optimisation objectives for the running example yields an approximate Pareto-front corresponding to these two attributes being maximized.

Pareto-front generation is orthogonal to our method; an effective approach that uses multi-objective optimisation genetic algorithms to automate the ARL abstract policy synthesis for very large search spaces has recently been proposed in [18], and can be directly employed by ARL. Alternatively, simple random search can be used for the same purpose, as in the case study that we describe later in the paper. Both our case study and the approach from [18] use the probabilistic model checker PRISM [24] for the quantitative verification of the PCTL-encoded constraints, so switching to using the approach from [18] is straightforward.

4.3 Stage 3: Safe learning

The last ARL stage exploits the approximate Pareto-optimal set of abstract policies synthesised in Stage 2. A policy is selected from this set by taking into account the trade-offs that different policies achieve for the optimisation objectives used to assemble the set. The selected abstract policy is then used to ensure that the RL agent achieves the required constraints by removing the low-level MDP actions that do not correspond to *options* from the abstract policy. For instance, assume that the selected abstract policy for our running example requires the agent to never enter RoomA. In this case, should the agent be at Cartesian coordinates (5,9) (i.e. the position immediately to the North of the Start position), the action to move North and thus to enter RoomA is removed from the agent's action set, for this specific state. By disallowing the actions that are not associated with options permitted by the abstract policy, the RL agent's learning and learnt low-level behaviours are guaranteed to satisfy the problem constraints, as illustrated in the next section for two case studies.

Abstract policies intentionally reduce agent autonomy to prevent unsafe actions, but do not preclude it completely. For example, in the running example the agent must learn the flag locations within the rooms as well as the doorway areas safest to cross, information which is not contained within the abstract policies. Whilst abstract policy constraints may yield suboptimal RL policies with respect to the RL model in its entirety, this key feature guarantees safety and speeds learning.

5 Evaluation

5.1 Experimental Setup

We evaluated the effectiveness and generality of our ARL approach by applying it to a case study based on the navigation task described in Section 3, where the learning agent must navigate a guarded environment comprising hallways and rooms in order to collect flags distributed throughout.

For this case study we conducted a set of four experiments. The first of these experiments did not involve the use of our ARL approach, and was a standard RL implementation of the case study. This experiment serves as a baseline which we contrast with the ARL experiments in order to determine the effects of our method.

For all experiments we use a discount factor $\gamma = 0.99$ and a learning rate $\alpha = 0.1$ which decays to 0 over the learning run. For the baseline experiment we use an exploration probability $\epsilon = 0.8$ and for the ARL experiments $\epsilon = 0.6$, both decaying to 0 over the learning run. These values of ϵ have been chosen empirically in line with standard RL practice. As is standard practice when evaluating stochastic processes, we repeated each experiment multiple times (i.e. five times) and we evaluated the final policy for each experiment many times (i.e. 10,000 times) in order to ensure that the results are suitably significant [3].

Learning evaluation is done after each learning episode during a run, whilst we only perform five learning runs per experiment, error bars for the standard error of the mean show the statistical significance of the learning (Figures 4 and 5). Policy evaluations were done once learning had finished (Table 4).

5.2 Case Study

This case study is based on the scenario described in Section 3 and referred to throughout Section 4. In the interest of brevity, the details presented in these two previous sections will not be repeated here.

In our RL implementation, the agent receives a reward of 1 for each flag it collects and an additional reward of 1 for reaching the 'goal' area of the building. If the agent is captured the agent receives a reward of -1, regardless of any flags already collected.

We used the AMDP constructed during the first ARL stage as described in Section 4.1. In the second ARL stage, we generated 10,000 abstract policies with parameter values (i.e. state to action mappings) drawn randomly from a uniform distribution. Out of these abstract policies, QV using the probabilistic model checker PRISM identified 14 policies that satisfied the two constraints from Table 2. Figure 3 shows the QV results obtained for these 14 abstract policies, i.e. their associated probability of reaching the 'goal' area and expected number of flags collected. The approximate Pareto-front depicted in this figure was obtained using the two optimization objectives described in Section 4.2, i.e. maximizing the expected number of flags collected and the probability of reaching the 'goal' area of the building.

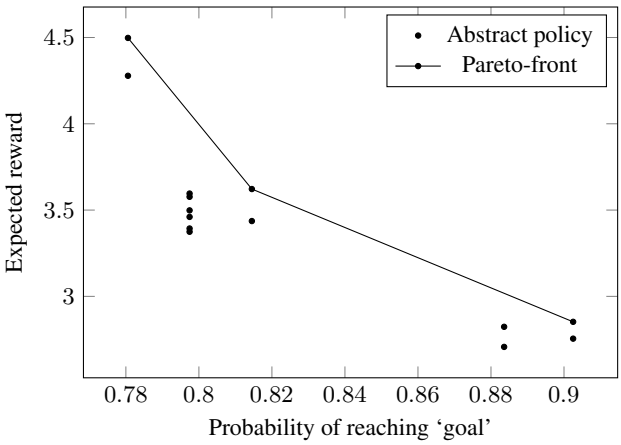


Figure 3. Pareto-front of abstract policies that satisfy the constraints from Table 2.

From this Pareto-front we selected three abstract policies to use in different experiments during the safe learning ARL stage, as explained in Section 4.3. The properties of these three abstract policies are shown in Table 3.

Table 3. Selected abstract policies to use for ARL in the guarded flag-collection.

Abstract Policy	Probability of Reaching 'goal'	Expected Reward
A	0.9	2.85
B	0.81	3.62
C	0.78	4.5

The baseline experiment (which did not use ARL) performed 2×10^7 learning episodes, each with 10,000 steps. This did not, however, reach a global optimum. Even after extensive learning, in excess of 10^9 learning episodes, conventional RL did not attain a superior solution. In contrast to our experiments where ARL was used, cf. abstract policy C, Table 4, a superior policy was learned much faster, further demonstrating the advantages of our approach. Figure 4 shows the learning progress for this experiment.

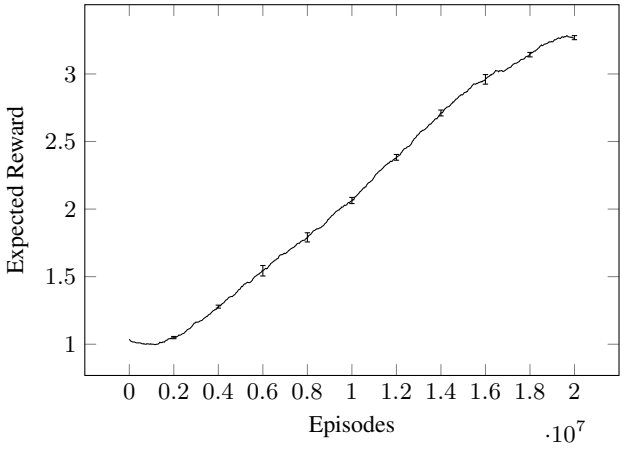


Figure 4. Learning for guarded flag-collection with no ARL applied.

Next, we ran three further sets of RL experiments, one for each of the abstract policies from Table 3. It was not necessary to have so many learning episodes as for the baseline experiment, since the abstract policy had the effect of guiding the agent with regard to the locations to enter next, and therefore only 10^5 episodes were necessary for the learning to converge. Figure 5 show the RL learning progress for each of the abstract policies used for ARL.

As can be seen from the results summarized in Table 4, the experiments where an abstract policy was applied resulted in an RL policy that: (a) satisfied the problem constraints from Table 2; and (b) matched the probability of reaching the 'goal' area and the expected reward of the abstract policy (cf. Table 3). The baseline experiment gave results that do not satisfy our constraints, which was expected given that only 14 of the 10,000 abstract policies synthesised by ARL satisfied these constraints.

6 Related Work

The ARL technique introduced in our paper belongs to a class of RL techniques called safe reinforcement learning [16]. The previous research on safe RL has proposed techniques that can enforce bounds on the either the reward obtained by the RL agent or on simple measures that are related to this reward. The technique proposed

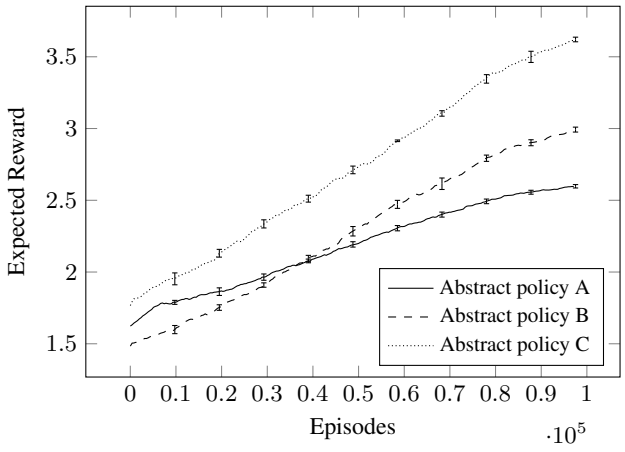


Figure 5. Learning for guarded flag-collection with ARL applied using the selected abstract policies A, B and C.

by Geibel [17] supports an inequality constraint on the reward cumulated by the RL agent or a maximum permitted probability for such a constraint to be violated. Safe RL techniques that support similar constraints by generalizing chance-constrained planning to infinite-horizon MDPs are presented by Mannor and Delage [11] and Ponda *et al.* [34]. The constraints supported by [11, 17, 34] are a subset of the types of constraints supported by ARL, which can handle the wide range of constraints that can be specified in PCTL.

Abe *et al.* [1] describe a safe RL technique in which high-level business and legal "constraint rules" are enforced during each value iteration of the RL process, and apply it to a tax collection optimization problem. Building on insights from financial decision making and robust process control, Casto *et al.* [8] introduce a safe RL technique that enforces constraints on the cumulative reward obtained by the RL agent, on the variance of this reward, or on some combination of the two. Moldovan and Abbeel [33] introduce a safe RL technique that enforces the RL agent to avoid irreversible actions by visiting only states from which it can return to the initial state. Our ARL technique operates with different types of constraints than [1, 8, 33], and is therefore complementary to these results.

ARL also differs from the existing safe RL approaches through its unique integration of quantitative verification and reinforcement learning, and use of abstract policies to enforce safe learning and a safe learnt policy. In contrast, existing techniques operate by modifying the reward function to "penalize" agent actions associated with high variance in the probability of attaining the reward [31] or to avoid irreversible actions [33], or by using domain knowledge to avoid unsafe states altogether [12].

Another distinguishing characteristic of ARL is its synthesis of an approximate Pareto-optimal set of permissible (abstract) policies. This offers a broad choice of trade-offs between relevant attributes of the optimisation problem that is not supported by existing safe RL techniques. A different area of RL research known as multi-objective RL (MORL) [28, 37] has also considered the problem of learning a policy that satisfies multiple objectives that may conflict with each other. However, neither single-policy MORL algorithms (which learn an optimal policy for each objective and then combine them to form a single policy [15, 29]) nor multi-policy algorithms (which learn an approximate Pareto-front for each objective [5] or a joint Pareto-front [32]) support the rich expressiveness provided by ARL through its use of reward-augmented PCTL constraints.

Table 4. Results for baseline and ARL experiments for guarded flag-collection.

Abstract Policy	Probability of Reaching 'goal'	Standard Error	Expected Reward	Standard Error
None	0.72	0.0073	4.01	0.031
A	0.9	0.0012	2.85	0.0029
B	0.81	0.0019	3.62	0.0037
C	0.78	0.0012	4.5	0.0041

7 Conclusion

We proposed the use of an abstract MDP formally analysed using quantitative verification as a means to restrict the action set of an RL agent to the actions that were proven to satisfy a set of required constraints, adding to the growing research on safe RL. Through a case study based on the benchmark RL flag-collection domain, we demonstrated that the hybrid ARL technique can be applied successfully. ARL requires that partial knowledge of the problem is provided a priori, and makes the typical assumption that RL will converge towards the optimal policy.

Unlike standard RL, our technique supports a wide range of safety, performance and reliability constraints that cannot be expressed using a single reward function and are not supported by existing safe RL techniques. Furthermore, the use of an AMDP allows the application of ARL with only limited knowledge about the environment, and ensures that ARL scales to much larger and complex models than would otherwise be feasible. Additionally, the construction of the AMDP and the expressiveness of the PCTL formulae, an expressive and convenient representation formalism for required properties, enables on the fly experimentation of constraints and properties without requiring modification of the underlying model.

Our future work on the ARL technique will include exploiting some of the more sophisticated constraints that can be specified in PCTL. For example, unbounded until PCTL formulae can be used to constrain the order in which the agent visits different rooms in the guarded flag-collection case study, e.g. $\mathrm{P}_{\geq 0.9}[\neg RoomA \ \mathrm{U} RoomB]$ requires that, with a probability of at least 0.9, the agent should not visit RoomA before RoomB. Furthermore, bounded until PCTL formulae can additionally place constraints on the number of time steps taken to achieve a certain outcome.

Additionally, we plan to research a means of updating the AMDP should it not accurately reflect the RL MDP. In the event that the RL agent encounters dynamics in the RL MDP that do not correlate with the AMDP, a means of feeding back this information to update the AMDP can be developed based on [13]. After updating the AMDP the constraints will need to be reverified and, if necessary, a new abstract policy will be generated.

Finally, we also plan to explore the possibility to employ multi-objective RL [28, 37] in the abstract policy synthesis of ARL, and to develop a variant of the technique where the actions restricted through QV and those learnt by the RL agent belong to disjoint sets. This ARL variant will support the common scenario in which a set of system attributes like cost and energy usage need to be optimised once strict constraints are guaranteed to be satisfied for some other system attributes such as availability and response time.

ACKNOWLEDGEMENTS

This paper presents research sponsored by the UK MOD. The information contained in it should not be interpreted as representing the views of the UK MOD, nor should it be assumed it reflects any current or future UK MOD policy.

REFERENCES

[1] N. Abe, P. Melville, C. Pendus, C. K. Reddy, D. L. Jensen, V. P. Thomas, J. J. Bennett, G. F. Anderson, B. R. Cooley, M. Kowalczyk, M. Domick, and T. Gardinier, 'Optimizing debt collections using constrained reinforcement learning', in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '11)*, pp. 75–84, Washington, DC, USA, (July 25-28 2010). ACM.

[2] S. Andova, H. Hermanns, and J.-P. Katoen, 'Discrete-time rewards model-checked', in *Formal Modeling and Analysis of Timed Systems*, eds., K. G. Larsen and P. Niebert, volume 2791 of *Lecture Notes in Computer Science*, 88–104, (2004).

[3] A. Arcuri and L. Briand, 'A practical guide for using statistical tests to assess randomized algorithms in software engineering', in *Proceedings of the 33rd International Conference on Software Engineering (ICSE '11)*, pp. 1–10, Honolulu, Hawaii, USA, (May 21-28 2011). ACM.

[4] J. A. Bagnell and J. G. Schneider, 'Autonomous helicopter control using reinforcement learning policy search methods', in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA '01)*, Seoul, Korea, (May 21-26 2001). IEEE.

[5] L. Barrett and S. Narayanan, 'Learning all optimal policies with multiple criteria', in *Proceedings of the 25th International Conference on Machine learning (ICML '08)*, eds., A. McCallum and S. Roweis, pp. 41–47, Helsinki, Finland, (July 5-9 2008). Omnipress.

[6] J. Baxter, A. Tridgell, and L. Weaver, 'KnightCap: A chess program that learns by combining TD(λ) with minimax search', in *Proceedings of the 15th International Conference on Machine Learning (ICML '98)*, pp. 28–36, Madison, Wisconsin, USA, (1998). Morgan Kaufmann.

[7] R. Calinescu, C. Ghezzi, M. Kwiatkowska, and R. Mirandola, 'Self-adaptive software needs quantitative verification at runtime', *Communications of the ACM*, **55**(9), 69–77, (September 2012).

[8] D. D. Castro, A. Tamar, and S. Mannor, 'Policy gradients with variance related risk criteria', in *Proceedings of the 29th International Conference on Machine Learning (ICML '12)*, eds., J. Langford and J. Pineau, pp. 935–942, Edinburgh, Scotland, UK, (June 26-July 1 2012).

[9] I. Cizelj, X. C. Ding, M. Lahijanian, A. Pinto, and C. Belta, 'Probabilistically safe vehicle control in a hostile environment', in *Proceedings of the 18th World Congress of the The International Federation of Automatic Control (IFAC '11)*, pp. 11803–11808, Milan, Italy, (August 28-September 2 2011).

[10] R. Dearden, N. Friedman, and S. Russell, 'Bayesian Q-learning', in *Proceedings of the Fifteenth National Conference on Artificial Intelligence (AAAI '98)*, pp. 761–768, Madison, Wisconsin, USA, (July 26-30 1998). AAAI Press.

[11] E. Delage and S. Mannor, 'Percentile optimization for Markov decision processes with parameter uncertainty', *Operations Research*, **58**(1), 203–213, (2010).

[12] K. Driessens and S. Džeroski, 'Integrating guidance into relational reinforcement learning', *Machine Learning*, **57**(3), 271–304, (December 2004).

[13] K. Efthymiadis and D. Kudenko, 'Knowledge revision for reinforcement learning with abstract MDPs', in *Proceedings of the 14th International Conference on Autonomous Agents and Multiagent Systems (AAMAS '15)*, eds., R. H. Bordini, E. Elkind, G. Weiss, and P. Yolum, pp. 763–770, Istanbul, Turkey, (May 4-8 2015).

[14] L. Feng, C. Wiltsche, L. Humphrey, and U. Topcu, 'Synthesis of human-in-the-loop control protocols for autonomous systems', *IEEE Transactions on Automation Science and Engineering*, **13**(2), 450–462, (April 2016).

[15] Z. Gábor, Z. Kalmár, and C. Szepesvári, 'Multi-criteria reinforcement learning', in *Proceedings of the 15th International Conference on Machine Learning (ICML '98)*, pp. 197–205, Madison, Wisconsin, USA, (1998). Morgan Kaufmann.

[16] J. García and F. Fernández, 'A comprehensive survey on safe reinforcement learning', *Journal of Machine Learning Research*, **16**(1), 1437–1480, (August 2015).

[17] P. Geibel, 'Reinforcement learning for MDPs with constraints', in *Proceedings of the 17th European Conference on Machine Learning (ECML '06)*, eds., J. Fürnkranz, T. Scheffer, and M. Spiliopoulou, volume 4212 of *Lecture Notes in Artificial Intelligence*, pp. 646–653, Berlin, Germany, (September 18-22 2006). Springer.

[18] S. Gerasimou, G. Tamburrelli, and R. Calinescu, 'Search-based synthesis of probabilistic models for quality-of-service software engineering', in *Proceedings of the 30th IEEE/ACM International Conference on Automated Software Engineering (ASE '15)*, pp. 319–330, Lincoln, Nebraska, USA, (November 9-13 2015). IEEE.

[19] H. Hansson and B. Jonsson, 'A logic for reasoning about time and reliability', *Formal Aspects of Computing*, **6**(5), 512–535, (September 1994).

[20] J.-P. Katoen, I. S. Zapreev, E. M. Hahn, H. Hermanns, and D. N. Jansen, 'The ins and outs of the probabilistic model checker MRMC', *Performance Evaluation*, **68**(2), 90–104, (February 2011).

[21] M. Kwiatkowska, 'Quantitative verification: Models, techniques and tools', in *Proceedings of the 6th joint meeting of the European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE '07)*, pp. 449–458, Dubrovnik, Croatia, (September 3-7 2007). ACM Press.

[22] M. Kwiatkowska, 'Advances in quantitative verification for ubiquitous computing', in *Proceedings of the 10th International Colloquium on Theoretical Aspects of Computing (ICTAC '13)*, eds., Z. Liu, J. Woodcock, and H. Zhu, volume 8049 of *Lecture Notes in Computer Science*, pp. 42–58, Shanghai, China, (September 4-6 2013). Springer.

[23] M. Kwiatkowska, G. Norman, and D. Parker, 'Stochastic model checking', in *Proceedings of the 7th International Conference on Formal Methods for Performance Evaluation (SFM '07)*, eds., M. Bernardo and J. Hillston, volume 4486 of *Lecture Notes in Computer Science*, pp. 220–270, Bertinoro, Italy, (May 28-June 2 2007). Springer.

[24] M. Kwiatkowska, G. Norman, and D. Parker, 'PRISM 4.0: Verification of probabilistic real-time systems', in *Proceedings of the 23rd International Conference on Computer Aided Verification (CAV '11)*, eds., G. Gopalakrishnan and S. Qadeer, volume 6806 of *Lecture Notes in Computer Science*, pp. 585–591, Snowbird, Utah, USA, (July 14-20 2011). Springer.

[25] C. Kwok and D. Fox, 'Reinforcement learning for sensing strategies', in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '04)*, pp. 3158–3163, Sendai, Japan, (September 28-October 4 2004). IEEE.

[26] M. Lahijanian, S. B. Andersson, and C. Belta, 'Temporal logic motion planning and control with probabilistic satisfaction guarantees', *IEEE Transactions on Robotics*, **28**(2), 396–409, (April 2012).

[27] L. Li, T. J. Walsh, and M. L. Littman, 'Towards a unified theory of state abstraction for MDPs', in *Proceedings of the Ninth International Symposium on Artificial Intelligence and Mathematics (ISAIM '06)*, pp. 531–539, Fort Lauderdale, Florida, USA, (January 4-6 2006).

[28] C. Liu, X. Xu, and D. Hu, 'Multiobjective reinforcement learning: A comprehensive overview', *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, **45**(3), 385–398, (March 2015).

[29] S. Mannor and N. Shimkin, 'A geometric approach to multi-criterion reinforcement learning', *Journal of Machine Learning Research*, **5**, 325–360, (April 2004).

[30] B. Marthi, 'Automatic shaping and decomposition of reward functions', in *Proceedings of the 24th International Conference on Machine learning (ICML '07)*, pp. 601–608, Corvallis, Oregon, USA, (June 20-24 2007). ACM.

[31] O. Mihatsch and R. Neuneier, 'Risk-sensitive reinforcement learning', *Machine Learning*, **49**(2), 267–290, (November 2002).

[32] K. V. Moffaert and A. Nowé, 'Multi-objective reinforcement learning using sets of pareto dominating policies', *Journal of Machine Learning Research*, **15**(1), 3663–3692, (November 2014).

[33] T. M. Moldovan and P. Abbeel, 'Safe exploration in Markov decision processes', in *Proceedings of the 29th International Conference on Machine Learning (ICML '12)*, eds., J. Langford and J. Pineau, pp. 1711–1718, Edinburgh, Scotland, UK, (June 26-July 1 2012). ACM.

[34] S. S. Ponda, L. B. Johnson, and J. P. How, 'Risk allocation strategies for distributed chance-constrained task allocation', in *American Control Conference (ACC '13)*, pp. 3230–3236, Washington, DC, USA, (June 17-19 2013). IEEE.

[35] M. Riedmiller, T. Gabel, and R. Hafner, 'Reinforcement learning for robot soccer', *Autonomous Robots*, **27**(1), 55–73, (July 2009).

[36] R. S. Sutton, D. Precup, and S. Singh, 'Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning', *Artificial Intelligence*, **112**(1-2), 181–211, (August 1999).

[37] P. Vamplew, R. Dazeley, A. Berry, R. Issabekov, and E. Dekker, 'Empirical evaluation methods for multiobjective reinforcement learning algorithms', *Machine Learning*, **84(1)**, 51–80, (July 2011).

[38] C. J. C. H. Watkins and P. Dayan, 'Q-learning', *Machine Learning*, **8**(3), 279–292, (May 1992).

[39] M. Wiering and M. Otterlo, 'Reinforcement learning and markov decision processes', in *Reinforcement Learning: State-of-the-art*, eds., M. Wiering and M. Otterlo, volume 12, 3–42, Springer, (2012).

[40] L. Xia and Q.-S. Jia, 'Policy iteration for parameterized markov decision processes and its application', in *Proceedings of the 9th Asian Control Conference (ASCC '13)*, pp. 1–6, Istanbul, Turkey, (June 23-26 2013). IEEE.

Regularizing Deep Learning Ensembles by Distillation

Alan Mosca[1] and **George D Magoulas**[1]

Abstract. Ensemble methods are often utilised to improve the generalisation of a model, by exploiting the diversity of multiple underlying learners trained on data of the same problem. When applied to Deep Learning, this requires the use of very large models that have to be trained multiple times, and the requirements of using Deep Learning Ensembles at test-time can be considerably high. It has been shown that a Deep Neural Network can be approximated by a smaller network, if it has been trained to reproduce the output of the larger network, in a process called *distillation*. It has also been shown previously that an Ensemble can be approximated by an Artificial Neural Network, with the introduction of additional training *pseudo-data*. We observe that it is possible to apply the same principles to approximate an Ensemble of Deep Neural Networks with a single Deep Network of the same kind and with the same structure as the members of the Ensemble, without significant loss of generalisation, without additional synthetic training data, and without having to use special provisions to train the new network, such as using the *soft target probabilities*. This process leads to a form of model regularisation, because the learning capacity is reduced whilst maintaining similar generalisation results. This behaviour is corroborated by experimental results on popular benchmark datasets in computer vision.

1 Introduction

Many Ensemble methods have been proposed [5], but they all require utilising the full group of trained classifiers at test-time. This can be computationally very expensive, especially when those classifiers come from Deep Learning.

Part of research work in Deep Leanring has been directed towards reducing the amount of computation required to train Deep networks [15, 9, 13]. However, at test-time, the computation required can still be expensive, making the use of Deep Learning Ensembles impractical in industrial and commercial applications.

It has been shown that it is possible, through a process called *distillation*, to train a smaller network to approximate the results of a larger one without loss of generality [7].

It has also been shown that an Artificial Neural Network is able to learn the distilled knowledge of an Ensemble, although this has been done in the context of shallow, fully-connected feed-forward ANNs [4].

In this paper, a new simplified recipe for the distillation of Deep Learning Ensembles is presented, which allows for the creation of a Deep Network that approximates and at the same time regularizes an existing trained Ensemble of Deep Neworks. Such a Distilled model is more portable than the original Ensemble, because it has a smaller footprint, both in computational and memory requirements. It is also shown how this method of distillation can be interpreted as a regularisation technique, and that in most cases the distilled model is able to improve on the generalisation of the Ensemble.

An experiment is described, in which several Deep Learning Ensembles are distilled into single networks, on some well-known benchmark datasets.

The paper is structured as follows. Section 2 reviews the existing literature and research upon which this technique is based. Section 3 explains the process of distillation of Ensembles and how it can be justified theoretically. Section 4 describes a practical experiment on existing benchmark datasets in computer vision. Finally, Section 5 makes final remarks and explores possible future work.

[1] Department of Computer Science and Information Systems, Birkbeck, University of London, email: {a.mosca, gmagoulas}@dcs.bbk.ac.uk

2 Prior Work

The idea of using a small model to represent a larger one has been previously explored in the field of simulation, where *metamodels* are often utilised as a representation of a simulation which is cheaper and faster to run. Many surveys exist on the subject of these metamodels [16], and many of the metamodels that have been adopted are Artificial Neural Networks [2, 6]. It has been shown that the function approximated by an Ensemble can be compressed into a smaller, faster model, by creating *pseudo data* in order to capture the function of the Ensemble, so that the small model can be trained on it [17, 4]. This however requires additional unlabelled data or, more commonly, the creation of synthetic data, which requires knowledge about the distribution of the original data. Whilst originally a random sample from the feature space was proposed [17], the authors of the *model compression* mechanism [4] have devised different ways of estimating the original distribution (RANDOM, NBE, MUNGE) for more accurate samples of the synthetic data. These are all approximate methods for estimating the original distribution of the data, which is unknown.

It has also been shown that a small network can approximate the function of a much deeper network by learning the "soft target probabilities" of the more cumbersome network [7]. The authors also propose a way to accelerate the training of a model on a large dataset by using an *Ensemble of specialists*. However, they do not then try to distil the Ensemble into a smaller model.

It is therefore noticed that there is no experimental work showing how Deep Learning Ensembles can be effectively approximated without having to resort to either additional synthetic training data, or "tricks" to improve the efficiency of the transfer of learning between models.

3 Distillation of Ensembles

A new method for distilling the knowledge from an Ensemble of Deep Networks is hereby proposed which does not include the typical limitations of *compressed models* or *distilled networks*.

The proposed approach does not make use of soft target probabilities and does not create additional synthetic training data. This recipe relies solely on the assumption that the original Ensemble is likely to have overfit the training data. The soft target probabilities are not used to provide a simplification to the procedure. We found empirically that the difference was small enough to justify dropping it.

By obtaining the labels produced by the Ensemble on the training set X_{train} and using them to train a new classifier of the same type and shape as those used as members of the Ensemble (Algorithm 1), it is possible to construct a regularised version of the approximated function $\bar{f}(X)$, which is learned by a model $h(X)$.

The validation and testing of the model is performed on the original validation and test sets. This ensures that the distilled model's learned function $\bar{f}'(X)$ is evaluated on how well it approximates $f(X)$– the (unknown) function that correctly labels the data, which is the target of the learning – rather than $\bar{f}(X)$, which was the original goal of the learning process.

Algorithm 1 Regularized Distillation

$h_E(X, Y) \leftarrow$ ensemble trained on X_{train} and labels Y_{train}
$y_h(X_{train}) \leftarrow$ hard target outputs of $h_E(X_{train}, Y_{train})$
$h_D(X, y_h(X)) \leftarrow$ classifier trained on X_{train} and labels $y_h(X)$
evaluate $h_D(X, y_h(X))$ on the original validation and testing sets (X_{valid}, Y_{valid}) and (X_{test}, Y_{test})

Although in theory it is possible to utilise any architecture for the distilled network, it is practical to re-use the same architecture applied as a base classifier. This avoids the need to fit additional hyperparameters and the architecture is already known to be suitable to the problem. It also serves to demonstrate the point that improved learning can be achieved with no additional capacity.

4 Experimental Analysis

This section demonstrates how the methodology has been applied successfully to distil the knowledge learned by some Deep Learning Ensembles. For all datasets, we tried Bagging [3] and AdaBoost [14], with 3 members for each Ensemble, and then trained a distilled version of the Ensemble according to the procedure described in Section 3. Each experiment was repeated 5 times, and the mean misclassification errors have been reported.

To accelerate the learning process, all the models were trained using Adam [9], with a schedule of 20 epochs with exponentially decaying learning rates. Dropout [8] was also used, as it is a common regularization technique for Deep Networks. Training was limited to 40 epochs per network to accelerate the iterative process. All the Enembles have been trained in less than 8 hours each.

To ensure that the comparison was fair, the distilled network received the same initialisation weights as all the Ensemble members. Although this in general reduces the diversity of the Ensemble, in this specific case it removed the doubt about whether the improved performance could have been due to a more favourable random initialisation.

4.1 Datasets

A few benchmark datasets that are common to Deep Learning have been used to evaluate the proposed methodology. No data augmentation has been performed on the datasets, apart from simple normalization techniques. The networks have been kept intentionally relatively small, compared to some exisiting state-of-the-art results, in order to accelerate the iterative experimental process.

4.1.1 MNIST

MNIST [11] is a common computer vision dataset that associates pre-processed images of hand-written numerical digits with a class label representing that digit. The input features are the raw pixel values for the 28×28 images, in grayscale, and the outputs are the numerical value between 0 and 9.

The Convolutional Neural Network–CNN used for MNIST has the following structure:

- An input layer of 784 nodes
- 64 5×5 convolutions
- 2×2 max-pooling
- 128 5×5 convolutions
- 2×2 max-pooling
- A fully connected layer of 1024 nodes
- Dropout with $P(D) = 0.5$
- a Softmax layer with 10 outputs (one for each class)

4.1.2 CIFAR-10

CIFAR-10 is a dataset that contains 60000 small images of 10 categories of objects. It was first introduced in [10]. The images are 32×32 pixels, in RGB format. The output categories are *airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck*. The classes are completely mutually exclusive so that it is translatable to a *1-vs-all* multiclass classification. Of the 60000 samples, there is a training set of 40000 instances, a validation set of 10000 and a test set of another 10000. All sets have perfect class balance.

The CNN used for CIFAR-10 has the following structure:

- An input layer of 3096 nodes
- 128 3×3 convolutions, with 1×1 padding
- Dropout with $P(D) = 0.25$
- 128 3×3 convolutions, with 1×1 padding
- Dropout with $P(D) = 0.25$
- 2×2 max-pooling
- 256 3×3 convolutions, with 1×1 padding
- Dropout with $P(D) = 0.25$
- 256 3×3 convolutions, with 1×1 padding
- Dropout with $P(D) = 0.25$
- 2×2 max-pooling
- A fully connected layer of 1024 nodes
- Dropout with $P(D) = 0.5$
- a Softmax layer with 10 outputs (one for each class)

Batch normalization is applied at all layers. This is a CNN that can be trained quick enough to allow the creation of an Ensemble and experiment iteratively, whilst still obtaining a good error rate.

4.1.3 CIFAR-100

CIFAR-100 is a dataset that contains 60000 small images of 100 categories of objects, grouped in 20 super-classes. It was first introduced in [10]. The image format is the same as CIFAR-10. Class labels are provided for the 100 classes as well as the 20 super-classes. A superclass is a category that includes 5 of the fine-grained class labels (e.g. "insects" contains *bee, beetle, butterfly, caterpillar, cockroach*). Of the 60000 samples, there is a training set of 40000 instances, a

validation set of 10000 and a test set of another 10000. All sets have perfect class balance.

The CNN used for CIFAR-100 has the following structure:

- An input layer of 3096 nodes
- 128 3×3 convolutions, with 1×1 padding
- Dropout with $P(D) = 0.25$
- 128 3×3 convolutions, with 1×1 padding
- Dropout with $P(D) = 0.25$
- 2×2 max-pooling
- 256 3×3 convolutions, with 1×1 padding
- Dropout with $P(D) = 0.25$
- 256 3×3 convolutions, with 1×1 padding
- Dropout with $P(D) = 0.25$
- 2×2 max-pooling
- A fully connected layer of 1024 nodes
- Dropout with $P(D) = 0.5$
- a Softmax layer with 10 outputs (one for each class)

Batch normalization is applied at all layers. This is a CNN that can be trained quick enough to allow the creation of an Ensemble and experiment iteratively, whilst still obtaining a good error rate.

4.1.4 *UCI Datasets*

The original paper on model compression [4] uses a selection of datasets from the UCI Machine Learning repository [1]. These datasets are generally small and because the data includes features that have already been transformed, they are best suited for networks with a small number of fully-connected layers. We excluded some of the dataset because they were far too small to evaluate using Deep Learning methods. Table 1 shows the details for each of the datasets used.

dataset	train	validation	test	features	classes
adult	29303	3259	16283	14	2
covtype	464808	58101	58102	54	8
letters	16000	2000	2000	16	26

Table 1. UCI Dataset Characteristics

4.2 Results

Considering the misclassification on the test set of each of the experimental datasets, it can be noted that the Distilled Ensemble is able to generalise as well as the original Ensemble, and in some cases even better. These small generalisation improvements have been attributed to the fact that forcing a single network to learn the function approximated by an Ensemble can serve as a form of regularisation.

4.2.1 *MNIST*

Table 2 shows the results on the MNIST dataset. AdaBoost seems to perform worse than bagging, presumably because the added emphasis on the "hard-to-classify" examples heavily imbalances the training sets, given that so few examples are misclassified in the first place. However, even with such a simple network it is possible to improve on the Ensemble's results with the regularization provided by

	AdaBoost	Bagging
Ensemble	0.63%	0.59%
Distilled	0.52%	0.55%

Table 2. Test misclassification error on MNIST

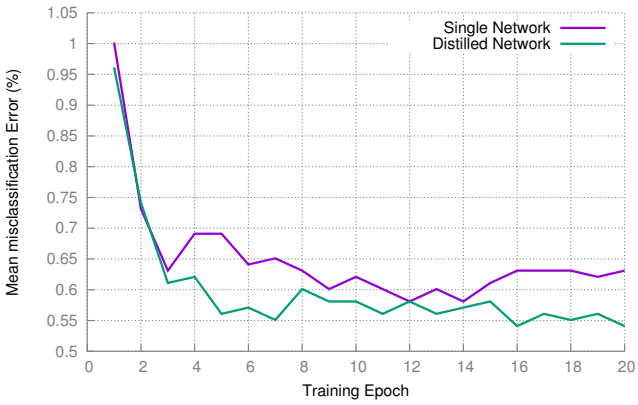


Figure 1. Test misclassification rate - MNIST

the distillation process, reaching a very small error level that is comparable to some much more complex models (e.g. Network in Network [12]), when no dataset augmentation is performed. For comparison, the test misclassification rate for the single network, trained on a random resample of the original training set with the same training schedule, is 0.66%. A random resample is used, because both Bagging and AdaBoost perform random resamples of the training set for each round, which inevitably creates an imbalance in the class label distribution, making the learning process harder.

Figure 1 shows how the distilled network learns the original function $f(X)$ for the MNIST dataset faster than the single original network, despite being trained on the approximated function $\bar{f}(X)$, reinforcing our argument that our method provides regularisation to the model.

4.2.2 *CIFAR-10*

	AdaBoost	Bagging
Ensemble	24.61%	22.30%
Distilled	24.05%	23.65%

Table 3. Test misclassification error on CIFAR-10

Table 3 shows the results on the CIFAR-10 dataset. For comparison, the test misclassification rate for the single network, trained on a random resample of the the original training set with the same training schedule, is 26.77%. As with MNIST, the Distilled network is able to generalise slightly better than the single network, receiving well the information learned by the Ensemble, with no additional synthetic data or learning capacity.

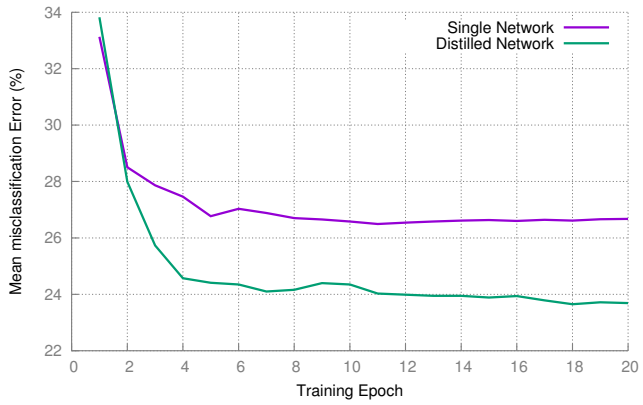


Figure 2. Test misclassification rate - CIFAR-10

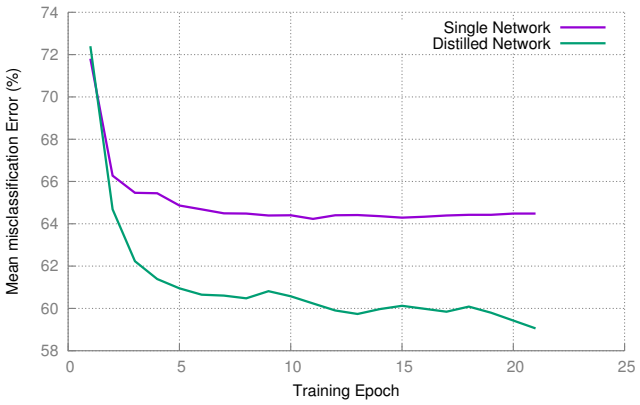


Figure 3. Test misclassification rate - CIFAR-100

However, in the case of Bagging, the distilled network performed slightly worse than the original Ensemble. This is attributed to the fact that, having used a bag of 3 CNNs, it is plausible that the Ensemble hasn't overfit the training data yet, and that therefore the distillation process only reduced the learning capacity without regularizing. However, there are still advantages in using the distilled network: it provides a better estimator than the original single network, it runs faster than the Ensemble, and is a much smaller model than the Ensemble.

Figure 2 also shows how the distilled network learns the original function $f(X)$ of the CIFAR-10 dataset faster than the single original network, despite being trained on an approximated function $\bar{f}(X)$, supporting our argument that the proposed method provides regularisation to the model.

4.2.3 CIFAR-100

	AdaBoost	Bagging
Ensemble	58.41%	57.69%
Distilled	57.85%	58.66%

Table 4. Test misclassification error on CIFAR-100

Table 4 shows the results on the CIFAR-100 dataset. The test misclassification rate for the single network, trained on a random resample of the the original training set with the same schedule, is 63.29%.

The same observations that were made for CIFAR-10 with regards to the distilled Ensemble not beating the original Ensemble because the model is not overfitting also apply to this experiment.

Figure 3 also shows how the distilled network learns the original function $f(X)$ faster than the single original network, despite being trained on $\bar{f}(X)$, reinforcing our argument that our method provides regularisation to the model.

4.2.4 UCI Datasets

Table 5 shows the results on the UCI datasets. For these experiments we only used Bagging as the Ensemble method. We can see that even

	ensemble	distilled	single
adult	24.06%	24.06%	24.09%
covtype	16.47%	18.66%	19.51%
letters	4.17%	5.90%	4.70%

Table 5. UCI Dataset Results - Bagging only

on these small datasets the distilled network mostly performs better than the single network, but given the much smaller size of the base learners, the reduction in learning capacity provided by the distilled network is too strong, and the regularization effect disappears because the distilled network is not able to completely learn $\bar{f}(X)$. This was verified by running variants that had much larger distilled networks than the base classifier of the Ensemble, and it was found that the distilled network again was able to improve the learning of the Ensemble in most cases.

5 Concluding Remarks

In this paper we have shown that it is possible to use a single network to approximate a Deep Learning Ensemble of networks of the same shape, and that the process of distillation can be used as a regulariser to reduce the overfitting of a model, and improve its generalisation. This method does not require additional synthetic training data and does not utilise the *soft target probabilities* from the Ensemble for training of the distilled network.

An experiment on three well-known datasets in the computer vision domain has been described, with results that corroborate the claims of the method. This experiment could be extended by utilising state-of-the-art Deep Learning models with the best known performance on the benchmark datasets. Due to very long times required to train these state-of-the-art models, it is however impractical to perform iterative experiment on such an Ensemble, unless one has vast resources at their disposal. Moreover, the state-of-the-art results on CIFAR-10 and CIFAR-100 have been achieved by using a training set of 50000 examples, obtained by utilising the hold-out validation set as training data. The hold-out validation set is also utilised to perform selection in the Ensemble, so as to avoid selecting an overfitted base classifier. The error obtained by the base classifier would

therefore be larger, due to the lack of the 10000 additional training examples.

In the future, this work can be extended by studying how to generate a Distilled Ensemble whilst training the original Ensemble, therefore creating an *on-the-fly* distillation process, which would eliminate the requirement for saving and running the entire Ensemble solely wiht the intent to create the distillation training set.

REFERENCES

[1] K. Bache and M. Lichman. UCI Machine Learning Repository, 2013.

[2] Adedeji B Badiru and David B Sieger, 'Neural network as a simulation metamodel in economic analysis of risky projects', *European Journal of Operational Research*, **105**(1), 130–142, (1998).

[3] L. Breiman, 'Bagging predictors', *Machine Learning*, **24**(2), 123–140, (1996).

[4] Cristian Bucilu, Rich Caruana, and Alexandru Niculescu-Mizil, 'Model compression', in *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 535–541. ACM, (2006).

[5] Thomas Dietterich, 'Ensemble methods in machine learning', in *Multiple Classifier Systems*, volume 1857 of *Lecture Notes in Computer Science*, 1–15, Springer Berlin / Heidelberg, (2000).

[6] DJ Fonseca, DO Navaresse, and GP Moynihan, 'Simulation metamodeling through artificial neural networks', *Engineering Applications of Artificial Intelligence*, **16**(3), 177–183, (2003).

[7] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean, 'Distilling the knowledge in a neural network', *arXiv preprint arXiv:1503.02531*, (2015).

[8] Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov, 'Improving neural networks by preventing co-adaptation of feature detectors', *arXiv preprint*, (2012).

[9] Diederik Kingma and Jimmy Ba, 'Adam: A method for stochastic optimization', *arXiv preprint arXiv:1412.6980*, (2014).

[10] Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images, 2009.

[11] Yann Lecun and Corinna Cortes, 'The MNIST database of handwritten digits'.

[12] Min Lin, Qiang Chen, and Shuicheng Yan, 'Network in network', *arXiv preprint arXiv:1312.4400*, (2013).

[13] Alan Mosca and George D Magoulas, 'Adapting resilient propagation for deep learning', *UK Workshop on Computational Intelligence*, (2015).

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

[15] T. Tieleman and G. Hinton. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning, 2012.

[16] G Gary Wang and Songqing Shan, 'Review of metamodeling techniques in support of engineering design optimization', *Journal of Mechanical design*, **129**(4), 370–380, (2007).

[17] Xinchuan Zeng and Tony R. Martinez, 'Using a neural network to approximate an ensemble of classifiers', *Neural Processing Letters*, **12**(3), 225–237, (2000).

Hybrid Approaches to Determine Existence of Emotions in Facial Gestures

Isidoros Perikos[1] and **Epaminondas Ziakopoulos**[1] and **Ioannis Hatzilygeroudis**[1]

Abstract. Facial gestures constitute meaningful aspects of human behaviour and can express a wide range of emotional states and feelings. In this work, we present a neuro-fuzzy and a neuro-symbolic approach and examine their performance in specifying the existence of emotions in facial expressions. Due to the domain's high complexity and dimensionality, hybrid neural approaches seem to be quite suitable to be utilized and this work is a contribution towards examining this direction. Initially, a new given photo is analyzed and faces in the photo are detected using the Viola-Jones algorithm. Then, each face is analyzed separately and specific regions of the facial expression such as eyes, eyebrows and mouth, are analyzed and proper geometrical characteristics form each region are extracted. Extracted features represent the deformation of the facial expression and based on them two approaches, a neuro fuzzy inference approach and a neuro symbolic approach that utilizes neurules, are trained and used to specify the existence of emotions. An evaluation study was conducted on JAFFE and Cohn Kanade databases and revealed very promising results.

1 INTRODUCTION

The aim of facial expression recognition is to enable machines to automatically estimate the emotional content of a human face. Facial expressions form a universal language of emotions, which can instantly express a wide range of human emotional states and feelings. Since facial expressions assist in various cognitive tasks, reading and interpreting the emotional content of human expressions is essential to deeper understand human condition. In an early work on human facial emotions [14] it has been indicated that during a face-to-face human communication only 7% of the information of a message is communicated by the linguistic part of the message, such as spoken words, 38% is communicated by paralanguage (vocal part) and 55% is communicated by the facial expressions. So, the facial expressions constitute the most important communication medium in face-to-face interaction.

Giving to computer applications the ability to recognize the emotional state of humans from their facial expressions is a very important and challenging task with wide ranging applications. In general, affective computing systems need to perceive emotional reactions by the user and successfully incorporate this information into the interaction process [10]. The interaction between human and computer systems (HCI) would become much more natural and vivid if the computer applications could recognize and adapt to the emotional state of the human. Indeed, automated systems that can determine emotions of a human based on his/her facial expressions can improve the human computer interaction and give computer systems the opportunity to customize and adapt its response [4]. Embodied conversational agents can greatly benefit from spotting and understanding the emotional states of the participants, achieving more realistic interactions at an emotional level [11]. In intelligent tutoring systems, emotions and learning are inextricably bound together; so, recognizing the learner's emotional states could significantly improve the efficiency of the learning procedures delivered to him/her [1] [19]. Moreover, surveillance applications such as driver monitoring and elderly monitoring systems could benefit from a facial emotion recognition system, gaining the ability to deeper understand and adapt to the person's cognitive and emotional condition. Also, facial emotion recognition could be applied to medical treatment to monitor patients and detect their status. However, the analysis of the human face characteristics and the recognition of its emotional state are considered to be very challenging and difficult tasks. The main difficulty comes from the non-uniform nature of the human face and various limitations such as lightening, shadows, facial pose and orientation conditions [9].

In this work, we present two approaches to determine the emotional state of human facial expressions. Initially, facial expressions are analyzed and proper features are measured and extracted following an analytical, local-based approach. Given a new image, Viola-Jones algorithm is utilized to detect human faces in images [22]. Then, it locates and measures facial deformations of specific regions such as eyes, eyebrows and mouth and extracts geometrical characteristics such as locations, length, width and shape. The extracted features represent the deformations of the facial expression and based on them two classification approaches a neuro fuzzy and a neurule one are trained and are used to recognize whether a facial gesture conveys emotional content or it is neutral. The evaluation study was conducted on JAFFE and Cohn Kanade databases and revealed very encouraging results regarding the performance of the two approaches in determining whether a facial expression is neutral or conveys emotional content.

The rest of the paper is structured as follows: Section 2 presents related literature. Section 3 presents our work on facial emotion recognition and illustrates the functionality of the neuro fuzzy and the neuro-symbolic approaches. Section 4 presents the evaluation study conducted and the performance results of the approaches. Finally, Section 5 concludes the paper and provides directions that future work will examine

[1] Department of Computer Engineering and Informatics, University of Patras, Hellas, email: {perikos,ziakopoulo,ihatz}@ceid.upatras.gr

2 RELATED WORKS

In the literature there are a lot of research efforts on facial image analysis and emotion recognition [3] [21] [24]. Although a human can detect and interpret faces and facial expressions naturally with little or no effort, accurate facial expression recognition by machines is still a challenge. Authors, in the work presented in [2], present a method for emotional classification of facial expressions, which is based on histogram sequence of feature vector. The system is able to recognize five human expression categories: happy, anger, sad, surprise and neutral, based on the geometrical characteristics of the human mouth with an average recognition accuracy of 81.6%. The work presented in [20] recognizes facial emotions based on a novel approach using Canny, principal component analysis (PCA) technique for local facial feature extraction and an artificial neural network for the classification process. The average facial expression classification accuracy of the method is reported to be 85.7%. The authors in the work presented in [18], recognize four basic emotions: happiness, anger, surprise and sadness, focusing in preprocessing techniques for feature extraction such as, Gabor filters, linear discrimination analysis and PCA. They achieve in their experiments 93.8% average accuracy for images of the Jaffe face database with little noise and with particularly exaggerated expressions and an average accuracy of 79% in recognition of just smiling/non smiling expressions in the ORL database. In [15], authors use a SVM classifier to recognize the 6 basis emotions defined by [5] in Cohn-Kanade database. In the study, authors extract 22 features from facial still images and report an average accuracy of 87.9%. In [12], authors recognize the six basic Ekman's emotions in elders' facial expressions using a SVM classifier. Features are extracted for expressions based on the active shape model, which fits in the shape parameters, using techniques such as gradient descent. The SVM is trained on a dataset such as, JAFFE in Cohn-Kanade and MII, and authors report promising results. In [23], authors developed a system for the emotion classification through lower facial expressions using the adaptive SVM. The system extracts eight feature points from the mouth, chin and nose and feed them into the A-SVMs classifier to categorize expression. The system was tested on Jaffe database and reports an average accuracy of 74.5%.

3 EMOTION RECOGNITION APPROACHES

In general, the task of the recognition of emotions from facial expressions can be divided in three main stages. These stages are the detection and recognition of a face so that a face in an image is known for further processing, the analysis and the extraction of facial features which concerns the method used to represent the facial expressions and finally the classification stage which classifies the features extracted from a facial expression in the proper emotional category [9].

In the context of our work, initially, a given new image is analyzed and human faces that may appear in it are detected using the Viola-Jones method [22]. Then, each face is isolated analyzed, appropriate features are extracted and the based on them the expression is classifies it to the proper emotional or neutral category. When a face is detected in the image, the area of the face is located and is further analyzed. Initially, the facial area is normalized and the contrast is enhanced and after that, specific regions which contribute in recognizing the emotional content of the facial expression are specified. The specific regions, from which the proper features are extracted, are named Areas of Interest (AOIs) and are the region of the eyes, the area of the mouth and the eyebrows. The feature extraction process analyzes the AOIs and tries to extract proper facial features, which describe and model the characteristics of the facial region. Our methodology follows an analytical local-based approach and so the feature extraction is implemented in the specific AOIs of the face. The stages of the methodology's workflow are illustrated in Figure 1 on an example photo.

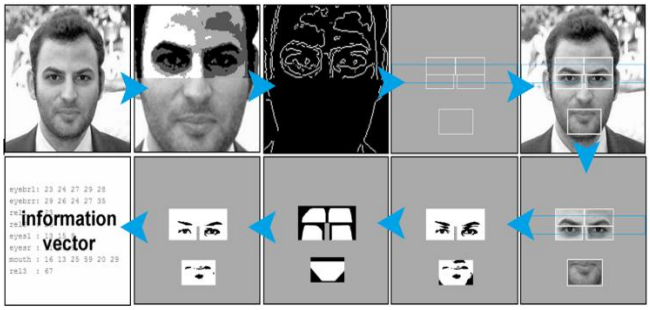


Figure 1. The stages of the analysis of a facial image.

The results of the analysis concern the extraction of the proper features and the formulation of the information vector that represents the characteristics of the facial gesture. In Figure 2, the expression representation and the features extracted from each AOI are illustrated.

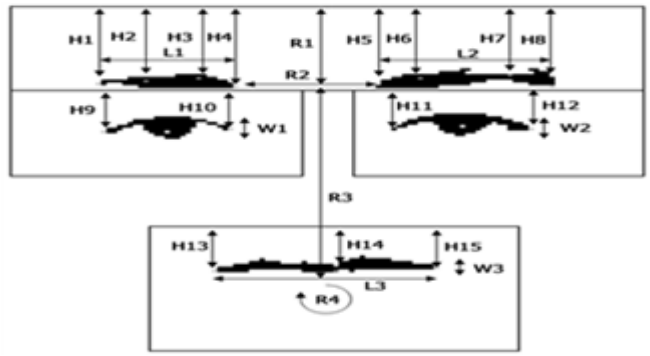


Figure 2. Expression Representation and Features Extracted from each AOI.

Left eyebrow

- H1: The height of the far left part.
- H2: The height of the part found on the 1/3 of the distance between the far left part and far right part.
- H3: The height of the part found on the 2/3 of the distance between the far left and far right part.
- H4: The height of the far right.
- L1: The length of the eyebrow.

Right eyebrow

- H5: The height of the far left part.
- H6: The height of the part found on the 1/3 of the distance between the far left and far right part.

- H7: The height of the part found on the 2/3 of the distance between the far left and far right part.
- H8: The height of the far right part.
- L2: The length of the eyebrow.

Left eye

- H9: The height of the far left part.
- H10: The height of the far right part.
- W1: The width of the eye.

Right eye

- H11: The height of the far left part.
- H12: The height of the far right part.
- W2: The width of the eye.

Mouth

- H13: The height of the far left part of the mouth.
- H14: The height of the part found on the 1/2 of the distance between the far left and far right part of the mouth.
- H15: The height of the far left part of the mouth.
- W3: The width of the mouth.
- L3: The length of the mouth.

Relative Values

- R1: The average height of the eyebrows.
- R2: The horizontal distance between the eyebrows.
- R3: The vertical distance between the eyebrows and the bottom of the mouth.
- R4: The ratio of the mouth length to the mouth width.

The features extracted from each facial expression are used to represent the characteristics of the expressions. Based on them we train and examine the performance of a neuro fuzzy and a neuro symbolic classification approach which relies on neurules are used to specify whether the facial gestures convey emotions or are neutral.

3.1 Neuro-Fuzzy and Neurule Classification Approaches

The complexity of the facial emotion recognition requires a different approach to solve other than standard hard computing. Using hard computing such as neural networks or other probabilistic classifiers, a high performance is achievable. However, soft computing approaches are more fitting, more intuitive and overall have the potential to yield better results for certain classification problems. In addition, they can face over fitting problems without any loss of generality. In the context of our study, we explore the performance of a neuro fuzzy and a neurule classification approach.

3.1.1 Neuro Fuzzy Inference System

The Fuzzy Inference System structure used to classify the human emotion is a typical example of soft computing classification. A basic idea underlying fuzzy logic is that notions or concepts are generally preferred over numbers, since numbers are very hard to be intuitively understood by human intelligence. Also fuzzy logic utilizes fuzzy if/then rules. Decision making should be able to tolerate uncertainty and inconsistency in order to bring forth the real world significance of the situation.

The Takagi-Sugeno-Kang method of fuzzy inference is used in this classification problem and consists of five steps. The first step is to fuzzify the inputs. The idea is to turn a fixed input number into a relevant value based on a linguistic set. In other words the output is a fuzzy degree of membership in the qualifying linguistic set. After the inputs are fuzzified a fuzzy operator is applied to obtain a single truth value if the fuzzified input variables have more than one membership values. Then an implication method is applied. The implication method results in a fuzzy set represented by a membership function. The consequent is then reshaped using a function associated with the antecedent. Finally the outputs are aggregated and defuzzified.

For the utilization of the neuro fuzzy system Matlab toolbox was utilized. For the training phase, the dataset of the facial expressions is divided into two subsets. The first subset contains 65% of the facial dataset and the second the remaining 35%. The first is used to create the fuzzy inference system and the second to evaluate it. The system is generated using subtractive clustering. Subtractive clustering is a fast, efficient algorithm for estimating the number of clusters and the clusters in a dataset. A custom vector that specifies the clusters center's range of influence in each of the data dimensions is required to achieve a very high performance for the system. The cluster estimates obtained from the subclust function can be used to model identification methods. In this case the subclust function is used to generate the Takagi-Sugeno-Kang fuzzy inference system trained to classify the input values.

3.1.2 Neurules

Neurules (**Neu**ral **rules**) are a kind of hybrid rules. Each neurule (Fig. 3a) is considered as an adaline unit (Fig.3b). The inputs C_i (i =1, ... , n) of the unit are the conditions of the rule. Each condition C_i is assigned a number sf_i, called a significance factor, corresponding to the weight of the corresponding input of the adaline unit. Moreover, each rule itself is assigned a number sf_0, called the bias factor, corresponding to the bias of the unit. Each input takes a value from the following set of discrete values: [1 (true), -1 (false), 0 (unknown)]. The output D, which represents the conclusion (decision) of the rule, is calculated via the formulas:

$$D = f(\boldsymbol{a}), \boldsymbol{a} = sf_0 + \sum_{i=1}^{n} sf_i\, C_i$$

where **a** is the *activation value* and *f(x)* the *activation function*, a threshold function:

$$f(a) = \begin{cases} 1\ if\ a \geq 0 \\ -1\ otherwise \end{cases}$$

Hence, the output can take one of two values, '-1' and '1', representing failure and success of the rule respectively.

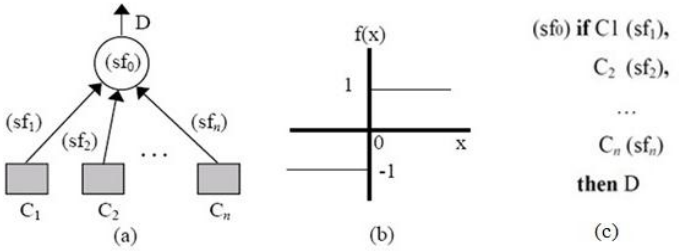


Figure 3. (a) Neurule as an adaline unit (b) Activation function (c) Neurule textual form.

The textual form of a neurule is presented in Fig.3c. C_is are the *conditions* and *D* represents the *conclusion* (decision) of the rule. The significance factor sf_i of a condition represents the significance (weight) of the condition in drawing the conclusion.

Neurules can be produced from empirical data through a well-defined process [6]. The produced neurules constitute an integrated rule base, which can be used for making inferences [6] and even produce explanations [7], through a well-defined inference process [6].

4 EXPERIMENTAL STUDY

The methodology and the system developed were extensively evaluated on facial images from two popular and widely used databases, the Japanese Female Facial Expression Database (JAFFE) [13] and the Cohn-Kanade database [8]. The Jaffe database consists of 213 facial pictures of 10 different posers. Approximately the two thirds (140 images) of the images of the database were selected to be part of the training dataset and the remaining 71 images were part of the test dataset. The Cohn-Kanade is a popular database which includes 486 sequences from 97 posers.

4.1 Subsection

After the facial images are analyzed and the features of each image are extracted, a dataset was formulated where each row includes values of the facial features of the expression and the corresponding emotion. For the needs of the study, the two thirds of the dataset were used for training, i.e. for producing the models of the two approaches, and the remaining one third of the dataset was used for the evaluation of the models. Given that we have a binary classification, we use the following four metrics: accuracy, precision, specificity and sensitivity, which are calculated as follows:

$$acc = \frac{TP + TN}{TP + FP + TN + FN}, \quad prec = \frac{TP}{TP + FP},$$

$$sen = \frac{TP}{TP + FN}, \quad spec = \frac{TN}{FP + TN}$$

where *TP* is the number of valid cases correctly classified, *FP* is the number of invalid cases that are misclassified, *TN* is the number of invalid cases correctly classified and *FN* is the number of valid cases that are misclassified.

The performance results of the neuro-fuzzy approach and the neurules approach are presented in Table 1 and Table 2 respectively.

Table 1. Neuro Fuzzy Performance Results

Metric	Jaffe	Cohn Kanade	Total
Accuracy	95.9	98.6	97.2
Precision	81.2	90.9	86.3
Sensitivity	96.8	100	98.4
Specificity	90.0	98.3	94.2

The evaluation results indicate that the neuro fuzzy approach is performing very well in both databases. The results indicate that the mechanism has a very good performance in determining whether a generated facial expression conveys emotions or not. The high sensitivity indicates that the mechanism can accurately identify emotional gestures that indeed are emotional. In Table 2, the performance of the neurules is illustrated.

Table 2. Neurules performance Results

Metric	Jaffe	Cohn Kanade	Total
Accuracy	85.2	87.1	86.2
Precision	79.3	80.0	79.7
Sensitivity	86.8	89.1	88.0
Specificity	84.0	83.3	83.7

The results of the study show that both models report very good performance. The best performance is achieved by the neuro-fuzzy model, for which accuracy and precision are better than those of the neurule approach in the experimental study. Regarding the neuro fuzzy approach, it performs better on both databases than the neurule approach. From a dataset scope, both methods report better performance in separating emotional from neutral facial expression in the Cohn Kanade database. We believe that this is mainly due to the fact that Cohn Kanade models present neutral expressions in a very consistent and inactive way. A noticeable point of both approaches on the two databases is that their high sensitivity indicates that the mechanism can accurately identify emotional facial expressions that indeed are emotional.

5 CONCLUSIONS

In this paper, we present two approaches, a neuro fuzzy and a neuro-symbolic one, that recognize whether a facial gesture expresses emotional content or not, and a performance on various facial expressions from databases. Initially, faces in images are detected via Viola Jones algorithm and after that, each facial expression is analyzed and proper features are measured and extracted following an analytical, local-based approach. Facial deformations of specific regions such as eyes, eyebrows and mouth

are deep analyzed and geometrical characteristics are extracted. The extracted features represent the deformations of the facial expression and based on them two classification approaches a neuro fuzzy and a neurule one are trained and are used to recognize expressions emotional content. The experimental study conducted on JAFFE and Cohn-Kanade face databases indicated quite promising performance and confirm the quality of performance of the identifications.

There are some points that the future work can focus on. Initially, a bigger scale evaluation on additional facial expression databases such as the ORL will give a deeper insight of the approaches performance. Moreover, currently the face analysis and the determination of the AOIs work on still frontal images of human face and so, a challenging extension will be to detect and analyse facial expressions from different poses. Furthermore, the future work will also examine ensemble classification schemas that combine base classifiers under different ensemble approaches. Exploring this direction is a key aspect of our future work.

REFERENCES

[1] Akputu, K. O., Seng, K. P., & Lee, Y. L.: Facial Emotion Recognition for Intelligent Tutoring Environment. In 2nd International Conference on Machine Learning and Computer Science (IMLCS'2013) , pp. 9-13, 2013

[2] Aung, D. M., & Aye, N. A.: Facial Expression Classification using Histogram Based Method. In. International conference on Signal Processing Systems, 2012

[3] Bettadapura, V.: Face expression recognition and analysis: the state of the art. arXiv preprint arXiv:1203.6722, 2012

[4] Clavel, C. (2015). Surprise and human-agent interactions. *Review of Cognitive Linguistics*, *13*(2), 461-477.

[5] Ekman, P.: Basic emotions. Handbook of cognition and emotion, pp. 45 - 60, 1999

[6] Hatzilygeroudis, I., & Prentzas, J. (2010). Integrated rule-based learning and inference. *IEEE Transactions on Knowledge and Data Engineering*, 22, 1549-1562.

[7] Hatzilygeroudis, I., & Prentzas, J. (2015). Symbolic-neural rule based reasoning and explanation. *Expert Systems with Applications*, 42, 4595-4609.

[8] Kanade, T., Cohn, J. F., & Tian, Y: Comprehensive database for facial expression analysis. In Automatic Face and Gesture Recognition, Proceedings, 2000

[9] Koutlas, A., & Fotiadis, D. I.: An automatic region based methodology for facial expression recognition. In Systems, Man and Cybernetics, 2008. SMC 2008. IEEE International Conference on, pp. 662-666, 2008

[10] Liebold, B., Richter, R., Teichmann, M., Hamker, F. H., & Ohler, P. (2015). Human Capacities for Emotion Recognition and their Implications for Computer Vision. *i-com*, *14*(2), 126-137.

[11] Liebold, B., & Ohler, P. (2013, September). Multimodal Emotion Expressions of Virtual Agents, Mimic and Vocal Emotion Expressions and Their Effects on Emotion Recognition. In *Affective Computing and Intelligent Interaction (ACII), 2013 Humaine Association Conference on* (pp. 405-410). IEEE.

[12] Lozano-Monasor, E., López, M. T., Fernández-Caballero, A., & Vigo-Bustos, F.: Facial Expression Recognition from Webcam Based on Active Shape Models and Support Vector Machines. In Ambient Assisted Living and Daily Activities, pp. 147-154, Springer, 2014

[13] Lyons, M. J., Akamatsu, S., Kamachi, M., & Gyoba, J.: Coding facial expressions with Gabor wavelets. In Proceedings of the Third IEEE International Conference on Automatic Face and Gesture Recognition, pp. 200-205, Los Alamitos, CA: IEEE Computer Society, 1998

[14] Mehrabian, A.: Communication without words. Psychology Today, vol. 2, no. 4, pp. 53–56, 1968

[15] Michel, P., El Kaliouby, R.: Facial expression recognition using support vector machines. In The 10th International Conference on Human-Computer Interaction, Greece, 2005

[16] Perikos, I., Ziakopoulos, E., & Hatzilygeroudis, I.: Recognizing Emotions from Facial Expressions Using Neural Network. In Artificial Intelligence Applications and Innovations pp. 236-245, Springer Berlin Heidelberg, 2014

[17] Perikos, I., Ziakopoulos, E., & Hatzilygeroudis, I. (2015). Recognize Emotions from Facial Expressions Using a SVM and Neural Network Schema. In *Engineering Applications of Neural Networks* (pp. 265-274). Springer International Publishing.

[18] Přinosil, J., Smékal, Z., & Esposito, A.: Combining features for recognizing emotional facial expressions in static images. In: Esposito, A., Bourbakis, N.G., Avouris, N., Hatzilygeroudis, I. (eds.) Verbal and Nonverbal Features of Human-Human and Human-Machine Interaction, pp. 56-69, Springer Berlin Heidelberg, 2008

[19] Shen, L., Wang M., Shen, R.: Affective e - learning: Using "emotional" data to improve learning in pervasive learning environment. In: Educational Technology & Society 12.2: pp. 176 – 189, 2009

[20] Thai, L. H., Nguyen, N. D. T., & Hai, T. S.: A facial expression classification system integrating canny, principal component analysis and artificial neural network. arXiv preprint arXiv:1111.4052, 2011

[21] Verma, A., & Sharma, L. K.: A Comprehensive Survey on Human Facial Expression Detection. International Journal of Image Processing (IJIP), 7(2), 171, 2013

[22] Viola, P., & Jones, M. J.: Robust real-time face detection. International journal of computer vision, 57(2), pp. 137-154, 2004

[23] Visutsak, P.: Emotion Classification through Lower Facial Expressions using Adaptive Support Vector Machines, JMMT: Journal of Man, Machine and Technology, Vol. 2, No. 1, pp. 12-20, 2013

[24] Zeng, Z., Pantic, M., Roisman, G. I., & Huang, T. S. (2009). A survey of affect recognition methods: Audio, visual, and spontaneous expressions. *IEEE transactions on pattern analysis and machine intelligence*, *31*(1), 39-58.

Extraction of Drug-Drug Interactions by Recursive Matrix-Vector Spaces

Víctor Suárez-Paniagua and **Isabel Segura-Bedmar** [1]

Abstract. The purpose of this paper is to explore in detail how a Recursive Neural Network (RNN) can be applied to classify drug-drug interactions from biomedical texts. The system is based on MV-RNN, a Matrix-Vector Recursive Neural Network built from the Stanford constituency trees of sentences. Drug-drug interactions are usually described by long sentences with complex structures (such as subordinate clauses, oppositions, and coordinate structures, among others). Our experiments show a low performance that may be probably due to the parser not being able to capture the structural complexity of sentences in the biomedical literature. This paper describes an extension of our previous work that has been accepted as a short paper at ECAI 2016.

1 INTRODUCTION

Nowadays there is a growing concern about drug adverse events (ADEs) since they are a serious risk for patient safety [4] as well as a cause of rising health care costs [33]. Drug–drug interactions (DDIs), a subset of ADE, are harms caused by the alteration of the effects of a drug due to recent or simultaneous use of one or more other drugs. Unfortunately, most DDIs are not detected during clinical trials, mainly because these trials are designed to assess the effectiveness of drugs rather than their safety [30].

The management of DDIs is a critical issue due to the overwhelming amount of information available on them [13]. The introduction of new technologies in primary care and hospitals has led to the development of electronic medical record systems, which has opened the possibility of incorporating decision support systems to prevent drug-drug interactions and inform on possible actions to take. However, the deployment of these systems is not widespread yet [22] and most systems in primary care do not support the management of DDIs. Therefore, clinicians and pharmacists must be able to manage by themselves the richness of information available on DDIs. There is a great amount of DDI databases [22]. Some of them are Stockley [29], Drug Interactions Facts [32], Medinteract, SEFH guide[2], Medscape[3], Hansten, Micromedex[4]. The diversity of DDI databases currently available poses a significant problem to health care professionals when collecting and evaluating information about a particular interaction from these databases.

In addition, several studies [18, 9] have shown that the quality of the DDI databases is very uneven and the consistency of their content is scarce, so it is very difficult to assign a real clinical significance to each drug interaction. Ideally, prescribing information about a drug should list its potential interactions, together with the following information about each interaction: its mechanism, its relation to the doses of both drugs, its time course, the factors that alter an individual's susceptibility to it, its seriousness and severity, and the probability of its occurrence [10, 2]. In practice, however, this information is rarely available [3]. A set of criteria to evaluate and compare 24 databases was proposed in [22]. The minimum quality criteria includes levels of severity and evidence, bibliographic reference, and the description of clinical management for each drug interaction. This study concluded that only 9 databases satisfied the minimum criteria. In particular, an increasingly important issue is the update periods of these databases. The update period is described only in 12 of the 24 databases, from immediate updates to a period of 3 years. Updates over 1 year should be inadmissible, and even more frequent updates should be required.

On the other hand, despite the availability of these databases, a great amount of the most current and valuable information is unstructured, written in natural language and hidden in published articles, scientific journals, books and technical reports. Drug interactions are bread and butter to journals of clinical pharmacology due to the vast number of interactions that can happen [3]. The number of articles published in the biomedical domain is increasing between 10,000 and 20,000 articles per week [5]. Each year 300,000 articles are published just within the pharmacology domain [7]. Therefore, the medical literature is probably by far the most effective system for detection of DDIs [3].

Physicians have to spend a long time reviewing DDI databases as well as the pharmacovigilance literature in order to assess the real clinical significance of a particular DDI and to prevent harmful drug interactions. The great amount of DDI databases and the deluge of published research have overwhelmed most health care professionals because it is not possible to be kept up-to-date of everything published about drug-drug interactions. Information extraction (IE), from both structured and unstructured data sources can be of great benefit in the pharmaceutical industry allowing identification and extraction of relevant information and providing an interesting way of reducing the time spent by health care professionals on reviewing the literature. Therefore, the development of tools for automatically extracting DDIs from biomedical texts is essential for improving and updating the drug knowledge databases.

In recent years, several NLP challenges have been organized to promote the development of NLP techniques applied to the biomedical domain, in particular, to the pharmacovigilance subject. In particular, the DDIExtraction shared tasks [24, 25] have been conceived with a dual objective: advancing the state-of-the-art of text-mining

[1] Computer Science Department, University Carlos III of Madrid, email: vspaniag,isegura@inf.uc3m.es

[2] www.sefh.es

[3] http://www.medscape.com/druginfo/druginterchecker

[4] http://www.micromedex.com/products/drugreax/

[5] http://www.nlm.nih.gov/pubs/factsheets/medline.html

techniques applied to the pharmacological domain, and providing a common framework for evaluation of the participating systems and other researchers interested in the task. While the first edition in 2011 only addressed the detection of drug DDIs from biomedical texts, the second edition also included their classification. Each DDI is classified with one of the following types of DDIs: mechanism (when the DDI is described by their pharmacokinetic mechanism), effect (for DDIs describing an effect or a pharmacodynamic mechanism), advice (when sentence is providing a recommendation or advice about a DDI) and int (the DDI appears in the text without providing any additional information). Most of the participating systems as well as the systems developed later have been based on Support Vector Machines (SVM) and on both linear and non-linear kernels, being the state-of-the-art performance of 77.5% F-score for detection and 67% F-score for classification [16]. All of them are characterized by the use of large and rich sets of linguistic features proposed by text miners and domain experts.

The prominent use of Deep Learning in Natural Language Processing and its good performance on this field makes it a promising technique in Relation Extraction, such as, Convolutional Neural Network (CNN) [35] or Recurrent Neural Network [34]. Recursive Neural Network (RNN) is a Deep Learning architecture, which is created from the parsing tree, that captures the semantic meaning for phrases and sentences. Specially, RNN used in Matrix-Vector spaces (MV-RNN) was the first Deep Learning architecture that obtained improvements in Relation Extraction [28]. This model introduces a RNN that captures the compositional[6] vector representation of long phrases or sentences. To this end, the model assigns a vector and a matrix to every word and it learns a compositional function for computing these representations.

Deep learning methods can be an interesting alternative to the classical methods since they are able to learn the best features to represent a problem. To the best of our knowledge, only two works so far have used Deep Learning methods for the classification of DDIs from biomedical texts. The first work exploited a RNN [8], achieving 68.64% F-measure for DDI classification, however the performance for each DDI type was not studied. The second one was based on the use of a CNN presented in [27]. The goal of our work is to continue the work started by [8], performing a detailed study about if RNNs are able to correctly classify the different DDI types. The main advantage of this approach over other Deep Learning architectures is that captures the semantic information in the whole sentence and in each word through the matrix and the vector, respectively. In this paper, we explore the use of MV-RNN applied to the DDI classification task. This paper describes an extension of our previous work that has been accepted as a short paper at ECAI 2016 [31].

The paper is organized as follows: Section 2 shows the state-of-the-art systems in DDI focusing on Deep Learning architectures. Section 3 describes the dataset and the methodology used for the extraction of DDIs. The experiments of this work are showed and discussed in Section 4. Finally, Section 5 presents the principal conclusions extracted from results obtained as well as proposals for futures works.

2 RELATED WORK

The DDIExtraction 2013 was the following edition of a first event organized in 2011, DDIExtraction Shared Task 2011 [24] whose main goal was the detection of drug-drug interactions from biomedical texts. The DDIExtraction 2013 task relies on the DDI corpus[7], which is a semantically annotated corpus of documents describing drug-drug interactions from the DrugBank database and MedLine abstracts on the subject of drug-drug interactions.

The highest performance presented in DDIExtraction Shared Task 2013 was Fondazione Bruno Kessler team (FBK-irst) [5]. The system consisted of two-phase DDI extraction framework: the DDIs were detected first and then, the extracted DDIs were classified according to the proposed types (mechanism, effect, advice and int) in the guidelines task. In the DDI detection phase, filtering techniques based on the scope of negation cues and the semantic roles of the entities involved were proposed to rule out possible negative instances from the test dataset. In particular, a binary SVM classifier was trained using contextual and shallow linguistic features to find these sentences which were not considered in the relation extraction phase. Once these negative instances were discarded from the test dataset, a hybrid kernel (combining a feature-based kernel, the shallow linguistic kernel (SL) [12] and the Path-enclosed Tree (PET) kernel [20]) was used to train a RE classifier. For the classification of the extracted DDIs, four separate SVM models were trained for each DDI type (using ONE-vs-ALL). The trained models were applied only on the extracted DDIs (by the DDI detection module) from the test dataset. Experiments on the training dataset showed that the filtering techniques improve both precision and recall with respect to applying only the hybrid kernel. They obtained 65.1% in F-measure. Table 1 shows the ranking information in the classification subtask.

	DDI-DrugBank	DDI-MedLine	Total
EFFECT	0.664	0.407	0.628
MECHANISM	0.705	0.383	0.679
ADVICE	0.705	0.286	0.692
INT	0.545	0.571	0.547
TOTAL	0.676	0.398	0.651

Table 1. FBK-irst [5] F1-scores for DDIExtraction Shared Task 2013.

Afterwards, the work of [16] overcomes this top ranking system using a rich set of lexical and syntactic features, such as word features, word pair features, dependency parse features, parse tree features and noun phrase-constrained coordination features to indicate if the target drugs are coordinated in a noun phrase. To ensure generalization of these features, target drug mentions and the rest of drugs in the sentence are anonymized. Moreover, numbers and tokens contained in the sentences are replaced by a generic tag and normalized into lemmas, respectively. They also removed interactions with the same drug name and drug mentions on the left of the colon that are detailed description of their interactions with other drugs. The system consisted of two separate steps: first a DDI detector is built to extract interacting drug pairs from all candidate interactions and second, a DDI type classifier is then trained to classify the interacting pairs into predefined relation categories. They reach 67% F-score for the classification task (see Table 2).

Beyond these machine learning approaches, MV-RNN obtained improvements in classification of semantic relationship with SemEval-2010 Task 8 [14] dataset. This approach generates a Neural Network from a binarized constituency parse tree of each sentence, learns the meaning of each constituent (a word of the sentence), and

[6] The compositionality is the important quality of natural language that determines the meaning of its words and the rules used to combine them [11]

[7] http://labda.inf.uc3m.es/DDIcorpus

	DDI-DrugBank	DDI-MedLine	Total
EFFECT	0.706	0.352	0.662
MECHANISM	0.714	0.455	0.693
ADVICE	0.736	0.429	0.725
INT	0.497	0.250	0.483
TOTAL	0.698	0.382	0.670

Table 2. Kim et al [16] F1-scores for DDIExtraction Shared Task 2013.

captures how this constituent changes the meaning of their neighbors through matrix-vector spaces.This system was the starting point in Relation Extraction via Deep Learning.

Following the MV-RNN approach, the work of [8] demonstrates that using dependency parse instead of constituency parse in a recursive neural network (RNN) model improves the performance and the training time for the task of relation classification. They use a modification of a RNN (C-RNN) to incorporate dependency graph nodes where each dependent between entities has a unique common ancestor. In addition, they add some internal features: the depth of the tree, distance between entities, context words and the type of dependencies. They evaluate in the SemEval-2013 Task 9.b where C-RNN obtains a 68.64% in F-measure. To the best of our knowledge, only this work use RNN for the DDI task, but they only report results in the overall dataset.

Recently, the work [27] demonstrates that a Deep Learning architecture, as Convolution Neural Network, can outperform the rest of machine learning techniques only using the word embeddings and position embeddings. Currently, this approach is the state-of-the-art system in DDI Extraction which obtained 69.75% in the subtask of classification in F-measure. As we can see in the Table 3, they also obtained a good performance in each class 70.24% for *mechanism*, 69.33% for *effect*, 77.75% for *advice* and 46.38% for *int* over the whole dataset.

From the review of the related work, we note that no work has performed a full and detailed study of the DDI classification by using MV-RNN. Therefore, our aim is to explore the Recursive Matrix-Vector Spaces in the biomedical domain and to work in a multi-class classification setting for reporting a complete analysis on the DDI corpus, studying in depth its performance for each type of DDI.

	DDI-DrugBank	DDI-MedLine	Total
EFFECT	-	-	0.693
MECHANISM	-	-	0.702
ADVICE	-	-	0.777
INT	-	-	0.463
TOTAL	0.715	0.521	0.697

Table 3. Liu et al [27] F1-scores for DDIExtraction Shared Task 2013.

3 EXPERIMENTAL SETTINGS

The Section 3.1 reviews the dataset used in the DDIExtraction Shared Task 2013, the DDI corpus [15]. In addition, the Section 3.2 briefly describe the details of the MV-RNN system proposed by Socher et al. [28] that is used in this work for the drug-drug interaction extraction.

3.1 Dataset

The major contribution of DDIExtraction has been to provide a benchmark corpus, the DDI corpus. The DDI corpus is a valuable annotated corpus that provides a gold standard data for training and evaluating supervised machine learning algorithms to extract DDIs form texts. It contains 233 selected abstracts about DDIs from Medline (DDI-MedLine) and other 792 texts from the DrugBank database (DDI-DrugBank). The corpus was manually annotated with a total of 18,502 pharmacological substances and 5028 DDIs. The quality and consistency of the annotation process was ensured through the creation of annotation guidelines and was evaluated by the measurement of the inter-annotator agreement (IAA) between two annotators. It should be noted that IAA can be considered as an upper bound on the performance of the automatic systems for the detection of DDIs. The agreement was very high for the DDI-DrugBank dataset (Kappa=0.83), and moderate for the DDIs in DDI-MedLine (0.55-0.72). This is due to the fact that MedLine abstracts have far more complexity than texts from the DrugBank database, which are usually expressed in simple sentences. A detailed description of the method used to collect and process documents can be found in [23]. The corpus is distributed in XML documents following the unified format for PPI corpora proposed by Pyysalo et al., [21] (see Figure 1). A detailed description and analysis of the DDI corpus and its methodology is described in [15].

As it was already mentioned, four different types of DDI relationships are included in the DDI corpus. Bellow, they are described in more detail by examples:

- mechanism: This type is used to annotate DDIs that are described by their pharmacokinetic (PK) mechanism (e.g. Grepafloxacin may inhibit the metabolism of theobromine).
- effect: This type is used to annotate DDIs describing an effect (e.g. In uninfected volunteers, 46% developed rash while receiving SUSTIVA and clarithromycin) or a pharmacodynamic (PD) mechanism (e.g. Chlorthalidone may potentiate the action of other antihypertensive drugs).
- advice: This type is used when a recommendation or advice regarding a drug interaction is given (e.g. UROXATRAL should not be used in combination with other alpha-blockers).
- int: This type is used when a DDI appears in the text without providing any additional information (e.g. The interaction of omeprazole and ketoconazole has been established).

Figure 2 shows some examples of annotated texts in the DDI corpus. The first example (A), taken from the MedLineDDI dataset, describes a DDI of mechanism type between a drug (named using a synonym different from its most common generic name, fomepizole) that inhibits the metabolism of a substance notapproved to be used in humans (1,3-difluoro-2-propranol). The second example (B) is also a sentence taken from MedLine and describes the consequence of a DDI (effect type) between estradiol (a generic drug) and endotoxin (a substance notapproved to be used in humans) in an experiment performed in animals. The last example (C) is a paragraph from the DDI-DrugBank dataset. Its first sentence describes the consequence of the interaction (effect type) of a drug, denominated by its brand name (Inapsine), when is co-administered with five different groups of drugs. The third sentence in C shows a recommendation to avoid these DDIs (advice type). Table 4 shows the distribution of the DDI types in the DDI corpus.

```
−<document id="DDI-DrugBank.d372">
  −<sentence id="DDI-DrugBank.d372.s0" text="Cytadren accelerates the metabolism of dexamethasone;">
      <entity id="DDI-DrugBank.d372.s0.e0" charOffset="0-7" type="brand" text="Cytadren"/>
      <entity id="DDI-DrugBank.d372.s0.e1" charOffset="39-51" type="drug" text="dexamethasone"/>
      <pair id="DDI-DrugBank.d372.s0.p0" e1="DDI-DrugBank.d372.s0.e0" e2="DDI-DrugBank.d372.s0.e1" ddi="true" type="mechanism"/>
    </sentence>
  −<sentence id="DDI-DrugBank.d372.s1" text="therefore, if glucocorticoid replacement is needed, hydrocortisone should be prescribed.">
      <entity id="DDI-DrugBank.d372.s1.e0" charOffset="14-27" type="group" text="glucocorticoid"/>
      <entity id="DDI-DrugBank.d372.s1.e1" charOffset="52-65" type="drug" text="hydrocortisone"/>
      <pair id="DDI-DrugBank.d372.s1.p0" e1="DDI-DrugBank.d372.s1.e0" e2="DDI-DrugBank.d372.s1.e1" ddi="false"/>
    </sentence>
  −<sentence id="DDI-DrugBank.d372.s2" text="Aminoglutethimide diminishes the effect of coumarin and warfarin.">
      <entity id="DDI-DrugBank.d372.s2.e0" charOffset="0-16" type="drug" text="Aminoglutethimide"/>
      <entity id="DDI-DrugBank.d372.s2.e1" charOffset="43-50" type="group" text="coumarin"/>
      <entity id="DDI-DrugBank.d372.s2.e2" charOffset="56-63" type="drug" text="warfarin"/>
      <pair id="DDI-DrugBank.d372.s2.p0" e1="DDI-DrugBank.d372.s2.e0" e2="DDI-DrugBank.d372.s2.e1" ddi="true" type="effect"/>
      <pair id="DDI-DrugBank.d372.s2.p1" e1="DDI-DrugBank.d372.s2.e0" e2="DDI-DrugBank.d372.s2.e2" ddi="true" type="effect"/>
      <pair id="DDI-DrugBank.d372.s2.p2" e1="DDI-DrugBank.d372.s2.e1" e2="DDI-DrugBank.d372.s2.e2" ddi="false"/>
    </sentence>
  </document>
```

Figure 1. Example of an annotated document of the DDI corpus.[26].

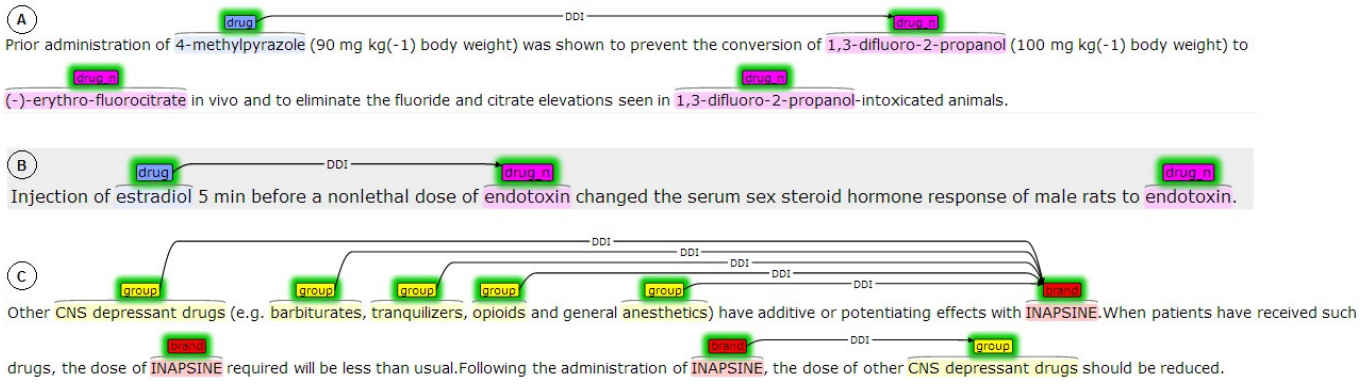


Figure 2. Some examples of sentences in the DDI corpus [26].

	DDI-DrugBank	DDI-MedLine	Total
EFFECT	1855 (39.4%)	214 (65.4%)	2069 (41.1%)
MECHANISM	1539 (32.7%)	86 (26.3%)	1625 (32.3%)
ADVICE	1035 (22%)	15 (4.6%)	1050 (20.9%)
INT	272 (5.8%)	12 (3.7%)	284 (5.6%)
TOTAL	4701	327	5028

Table 4. DDI types in the DDI corpus.

3.2 MV-RNN

Our approach is based on Recursive Neural Network. In particular, Matrix-Vector spaces (MV-RNN) was the first Deep Learning architecture that obtained improvements in classification of semantic relationships [28]. This model can determine the meaning of each word and the rules used to combine them in long sentences. To this end, the model assigns a vector and a matrix to every word and it learns a compositional function for computing these representations (Figure 3).

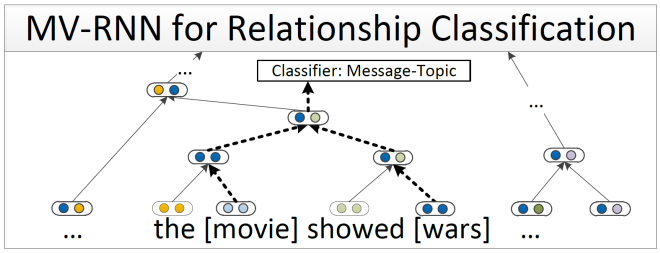


Figure 3. An example of how MV-RNN architecture learns the vectors in the nodes of the path between the two entities (dotted line) to classify their relationship [28].

Firstly, MV-RNN uses as input a binarized parse tree of phrases and sentences of arbitrary syntactic type and length from the Stanford Parser [17] as the RNN structure. Then, MV-RNN learns, in every node of the tree, a vector that represents the meaning of a constituent (a word or a sentence), and a matrix that captures how this constituent changes the meaning of their neighbours. Initially, we use the pre-trained 50-dimensional word vectors from [6] and the word matrices as an identity matrix with a small Gaussian noise. Afterwards, the MV-RNN architecture computes the parent vector p of each node as a single layer neural network:

$$p = g(W \begin{bmatrix} C_1 c_2 \\ C_2 c_1 \end{bmatrix} + b)$$

where c_1 and c_2 are the word vectors of their children in the binarized tree with dimensionality n, C_1 and $C_2 \in \mathbb{R}^{n \times n}$ are matrices for single words, $W \in \mathbb{R}^{n \times 2n}$ is a matrix that maps both words back into the same n-dimensional space, g is a non-linearity function and b is the bias term. In addition, another function is used for computing non-terminal phrase matrices:

$$P = W_M \begin{bmatrix} C_1 \\ C_2 \end{bmatrix}$$

where $W_M \in \mathbb{R}^{n \times 2n}$ are the weight matrices. Finally, the MV-RNN uses the computed vector of the highest nodes in the path between the pairs of words as features for predicting a DDI type label using a simple *softmax* classifier.

MV-RNN was adapted by Socher et al., [28] for the SemEval-10 task 8, whose goal was the classification of relationships between nominals. Thus, we had to transform the DDI corpus to the format of the SemEval-2010 task 8. Since the implementation of MV-RNN does not deal with discontinuous entities, we removed DDI candidates involving this kind of entities. In addition, if a sentence contains more than one interaction we separate them into independent sentences, i. e. we create one instance per interaction. Following this, we got a total of 33351 sentences. It should be noted that sentences from the SemEval-2010 task 8 dataset are much simpler than our sentences in the DDI corpus. Drug-drug interactions are usually described by long sentences with complex structures (such as subordinate clauses, oppositions, and coordinate structures, among others). Moreover, many drugs have very long and complex names, specially, chemical compounds (for example, *1,3-difluoro-2-propanol*). These drug names poses a significant challenge for the tokenization task of the biomedical texts. In fact, we observed that the Standford parser not being able to provide an accurate tokenization of the sentences in the DDI corpus. A wrong tokenization can cause a wrong syntactic tree parser, and thereby, a bad input for the MV-RNN. For this reason, chemical compound names were replaced by easier names of common drugs. For example, *1,3-difluoro-2-propanol* was substituted by *Rifampin*. Similarly, numerical expressions were also simplified.

4 RESULTS AND DISCUSSION

This section summarizes the evaluation results with the method MV-RNN with the DDI corpus and provides detailed analysis and discussion. The results have been measured by Precision (P), Recall (R) and F-score (F) for all the categories in the classification.

Table 5 shows the performance of MV-RNN over the DDI corpus test dataset. The model achieves an F-score of 46% using a syntactic information for building the RNN architecture. In general, precision is greater than recall due to the large number of False Negatives in each class caused by the misclassification. The class *advice* achieves the best performance (54% in F-measure) with respect to the other classes because these recommendations follows specific patterns and are easy to learn.

	True Positives	False Positives	False Negatives	Precision	Recall	F-measure
EFFECT	163	206	197	0.44	0.45	0.45
MECHANISM	105	102	194	0.51	0.35	0.42
ADVICE	108	71	113	0.60	0.49	0.54
INT	36	18	60	0.67	0.38	0.48
TOTAL	412	397	564	0.51	0.42	0.46

Table 5. Results on DDI Corpus using MV-RNN without external features.

Table 6 shows the results adding external features such as Part-of-Speech tags, the WordNet hypernyms and the name entity tags of the two words to the computed vector of the highest node in the relation for the classification in the *softmax* layer. These three features increase the performance F-measure (+4%) and the Recall for all the classes. Although the features raise the number of instances classified correctly, the False Positives are 38 instances bigger than without using external features. It may be due to an over-fitting in the *softmax* layer because in all the cases the number of False Negatives decreases whereas that the False Positives increases with respect to the Table 5 causing a trade-off problem.

	True Positives	False Positives	False Negatives	Precision	Recall	F-measure
EFFECT	193	232	167	0.45	0.54	0.49
MECHANISM	121	110	178	0.52	0.40	0.46
ADVICE	119	76	102	0.61	0.54	0.57
INT	37	17	59	0.69	0.39	0.49
TOTAL	470	435	506	0.52	0.48	0.50

Table 6. Results on DDI Corpus using MV-RNN with external features.

The main cause of our low performance may be due to the fact that Stanford parser is not able to correctly build the syntactic trees of sentences from the DDI corpus, which usually have complex structures (such as subordinate clauses, oppositions and coordinate structures, complex named entities among others). Wrong syntactic trees involve wrong RNN structures that are not able to capture the compositionality of sentences.

5 CONCLUSIONS

In this work we explore the extraction of interactions between drugs in the DDI corpus with a Recursive Neural Network used in Matrix-Vector spaces. From the review of the related work, Deep Learning architectures, such as Recursive or Convolutional Neural Networks, outperform the most common machine learning algorithms applied to relation extraction so far. MV-RNN can learn the meaning of a word and how that word modifies the context of the sentence through the combination of vectors and matrices. This recursive network contains the parsing information of each sentence regardless of length and grammatical structure.

However, MV-RNN does not seem to provide satisfactory results for the classification of DDIs. In fact, our results are much lower than those reported using a CNN [27]. We think that this is mainly due to the Standford parser not being able to capture the structural complexity of sentences in the biomedical literature. It should be noted that DDIs are usually described by long sentences with complex structures (such as subordinate clauses, oppositions, and coordinate structures, among others). Moreover, drugs can have long and complex, specially, chemical compounds (for example, 1,3-difluoro-2-propanol). This kind of names poses a significant challenge for the correct tokenization of the texts. For this reason, we decided to simplify this kind of named entities by blinding with common drug names. Unfortunately, our experiments show that the solution is not enough to solve this problem. Thus, we should have used a biomedical parser capable to provide accurate tokenization and syntactic trees of the sentences. Furthermore, MV-RNN uses the WordNet dictionary in order to achieve the hypernyms for the interacting drugs. However, most drugs are not included in WordNet since it is not a biomedical domain specialized resource.

As future work we propose to replace the initial word vectors from [6] by those from a new word vector model generated using a state-of-the-art word embedding system, such as Wor2Vec [19] and trained on a large collection of biomedical texts (for example, the latest version of MedLine). Thus, our new model will also include biomedical technical terms and jargon, which are not generally represented in [6]. Moreover, we would like to explore if the position embeddings and negative instance filtering improve our results. Alternatively, instead of the Stanford parser, we also plan to use a biomedical parser capable to capture the complex structures of the biomedical sentences in order to build the RNN structures. Moreover, other biomedical terminological resources such as the UMLS Methatesaurus [1] or the ATC drug classification system [8] will be included in order to achieve the hypernyms, instead of the use of WordNet. In addition, other deep learning architectures (such as CNNs) will be also studied.

ACKNOWLEDGEMENTS

This work was supported by eGovernAbility-Access project (TIN2014-52665-C2-2-R).

REFERENCES

[1] Alan R Aronson, 'Effective mapping of biomedical text to the umls metathesaurus: the metamap program.', in *Proceedings of the AMIA Symposium*, p. 17. American Medical Informatics Association, (2001).

[2] JK Aronson, 'Drug interactions-information, education, and the British National Formulary.', *British Journal of Clinical Pharmacology*, **57**(4), 473–86, (2004).

[3] JK Aronson, 'Communicating information about drug interactions', *British Journal of Clinical Pharmacology*, **63**, 637–639, (2007).

[4] CA Bond and Cynthia L Raehl, 'Adverse drug reactions in united states hospitals', *Pharmacotherapy: The Journal of Human Pharmacology and Drug Therapy*, **26**(5), 601–608, (2006).

[5] Md Faisal Mahbub Chowdhury and Alberto Lavelli, 'Fbk-irst: A multi-phase kernel based approach for drug-drug interaction detection and classification that exploits linguistic information', *Atlanta, Georgia, USA*, **351**, 53, (2013).

[6] Ronan Collobert and Jason Weston, 'A Unified Architecture for Natural Language Processing: Deep Neural Networks with Multitask Learning', in *Proceedings of the 25th International Conference on Machine Learning*, ICML '08, pp. 160–167, New York, NY, USA, (2008). ACM.

[7] S. Duda, C. Aliferis, R. Miller, A. Statnikov, and K. Johnson, 'Extracting drug-drug interaction articles from MEDLINE to improve the content of drug databases', in *AMIA Annual Symposium Proceedings*, volume 2005, p. 216, (2005).

[8] Javid Ebrahimi and Dejing Dou, 'Chain Based RNN for Relation Classification', in *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pp. 1244–1249, Denver, Colorado, (May–June 2015). Association for Computational Linguistics.

[9] J.A.D.C.M. Edward, P.A.A.J.G. Philip, and D.H.R.C. Van Bergen, 'Concordance of Severity Ratings Provided in Four Drug Interaction Compendia', *Journal of the American Pharmaceutical Association*, **44**(2), (2004).

[10] RE. Ferner and JK. Aronson, 'Communicating drug safety.', *JBM*, **333**, 1435, (2006).

[11] Gottlob Frege, 'Über Sinn und Bedeutung', *Zeitschrift für Philosophie und philosophische Kritik*, **100**, (1892).

[12] Claudio Giuliano, Alberto Lavelli, and Lorenza Romano, 'Exploiting Shallow Linguistic Information for Relation Extraction from Biomedical Literature.', in *EACL*, volume 2006, pp. 98–113, (2006).

[13] PD Hansten, 'Drug interaction management', *Pharmacy World & Science*, **25**(3), 94–97, (2003).

[14] Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz, 'Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals', in *Proceedings of the Workshop on Semantic Evaluations: Recent Achievements and Future Directions*, DEW '09, pp. 94–99, Stroudsburg, PA, USA, (2009). Association for Computational Linguistics.

[15] María Herrero-Zazo, Isabel Segura-Bedmar, Paloma Martínez, and Thierry Declerck, 'The DDIcorpus: An annotated corpus with pharmacological substances and drug–drug interactions', *Journal of Biomedical Informatics*, **46**(5), 914 – 920, (2013).

[16] Sun Kim, Haibin Liu, Lana Yeganova, and W. John Wilbur, 'Extracting drug–drug interactions from literature using a rich feature-based linear kernel approach', *Journal of Biomedical Informatics*, **55**, 23 – 30, (2015).

[8] http://www.whocc.no/atc_ddd_index/

[17] Dan Klein and Christopher D. Manning, 'Accurate Unlexicalized Parsing', in *In Proceedings of the 41St Annual Meeting of the Association for Computational Linguistics*, pp. 423–430, (2003).

[18] M.V. LAM, G.M. MCCART, and C. TSOUROUNIS, 'An assessment of Free, online drug-drug interaction screening programs(DSPs)', *Hospital pharmacy(Philadelphia, PA)*, **38**(7), 662–668, (2003).

[19] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean, 'Distributed representations of words and phrases and their compositionality', in *Advances in neural information processing systems*, pp. 3111–3119, (2013).

[20] Alessandro Moschitti, 'A study on convolution kernels for shallow semantic parsing', in *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, p. 335. Association for Computational Linguistics, (2004).

[21] S. Pyysalo, A. Airola, J. Heimonen, J. Bjorne, F. Ginter, and T. Salakoski, 'Comparative analysis of five protein-protein interaction corpora', *BMC bioinformatics*, **9**(Suppl 3), S6, (2008).

[22] A. Rodríguez-Terol, C. Camacho, et al., 'Calidad estructural de las bases de datos de interacciones', *Farmacia Hospitalaria*, **33**(03), 134, (2009).

[23] Isabel Segura-Bedmar, Paloma Martinez, and Cesar de Pablo-Sánchez, 'Using a shallow linguistic kernel for drug–drug interaction extraction', *Journal of biomedical informatics*, **44**(5), 789–804, (2011).

[24] Isabel Segura-Bedmar, Paloma Martinez, and Daniel Sanchez-Cisneros, 'The 1st ddiextraction-2011 challenge task: Extraction of drug-drug interactions from biomedical texts', in *Proceedings of the 1st Challenge Task on Drug-Drug Interaction Extraction 2011*, pp. 1–9, (2011).

[25] Isabel Segura-Bedmar, Paloma Martínez, and María Herrero-Zazo, 'Semeval-2013 Task 9: Extraction of Drug-Drug Interactions from Biomedical Texts', in *In Proceedings of the 7th International Workshop on Semantic Evaluation (SemEval)*, (2013).

[26] Isabel Segura-Bedmar, Paloma Martínez, and María Herrero-Zazo, 'Lessons learnt from the DDIExtraction-2013 Shared Task', *Journal of Biomedical Informatics*, **51**, 152 – 164, (2014).

[27] Qingcai Chen Shengyu Liu, Buzhou Tang and Xiaolong Wang, 'Drug-Drug Interaction Extraction via Convolutional Neural Networks', *Computational and Mathematical Methods in Medicine*, **Vol. 2016**, 8 pages, (2016).

[28] Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng, 'Semantic Compositionality Through Recursive Matrix-Vector Spaces', in *Proceedings of the 2012 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, (2012).

[29] IH Stockley, *Stockleys Drug Interaction*, Pharmaceutical Press, 2007.

[30] B.H.C. Stricker and B.M. Psaty, 'Detection, verification, and quantification of adverse drug reactions', *British Medical Journal*, **329**(7456), 44–47, (2004).

[31] Suárez-Paniagua, Víctor and Segura-Bedmar, Isabel, 'Using Recursive Neural Networks to detect and classify drug-drug interactions from biomedical texts', in *ECAI 2016: 22nd European Conference on Artificial Intelligence 29 August - 2 September 2016, The Hague, Holland*, (2016).

[32] D.S. Tatro, *Drug interaction facts*, Facts and Comparisons, St. Louis, 2003.

[33] Cornelis S van Der Hooft, Miriam CJM Sturkenboom, Kees van Grootheest, Herre J Kingma, and Bruno H Ch Stricker, 'Adverse drug reaction-related hospitalisations', *Drug Safety*, **29**(2), 161–168, (2006).

[34] Yan Xu, Lili Mou, Ge Li, Yunchuan Chen, Hao Peng, and Zhi Jin, 'Classifying relations via long short term memory networks along shortest dependency paths', in *In Proceedings of Conference on Empirical Methods in Natural Language Processing*, (2015).

[35] Daojian Zeng, Kang Liu, Siwei Lai, Guangyou Zhou, and Jun Zhao, 'Relation classification via convolutional deep neural network', in *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, pp. 2335–2344, Dublin, Ireland, (August 2014). Dublin City University and Association for Computational Linguistics.

Heuristic Constraint Answer Set Programming

Erich C. Teppan[1] and **Gerhard Friedrich**[1]

Abstract. Constraint answer set programming (CASP) is a family of hybrid approaches integrating answer set programming (ASP) and constraint programming (CP). These hybrid approaches have already proven to be very successful in various domains. In this paper we present the CASP solver ASCASS (A Simple Constraint Answer Set Solver), which provides novel methods for defining and exploiting problem-dependent search heuristics. Beyond the possibility of using already built-in problem-independent heuristics, ASCASS allows on the ASP level the definition of problem-dependent variable selection, value selection and pruning strategies, which guide the search of the CP solver. The concepts are exemplified and evaluated with respect to the real world Partner Units Problem (PUP). Due to a sophisticated heuristic, which cannot be represented by other ASP or CASP solvers, ASCASS shows superior performance.

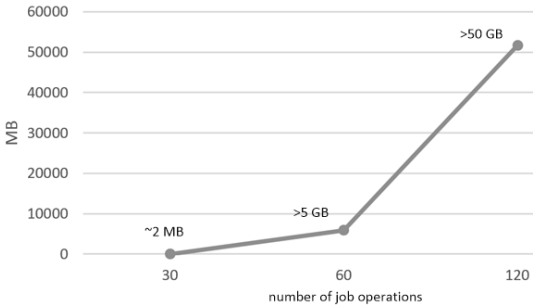


Figure 1. Grounding size for the incremental scheduling problem with respect to number of job operations

1 Introduction

During the last decade, Answer Set Programming (ASP) under the stable model semantics [9] has evolved to an extremely powerful approach for solving combinatorial problems. Especially conflict-driven search mechanisms contribute to the high performance of state-of-the-art solvers [8]. Furthermore, ASP provides superior problem encoding capabilities as ASP is strongly declarative in nature and even provides language features which go beyond first order.

However, the expressive power on the one hand and the potent conflict-driven search approach on the other hand do not come for free. Current ASP solvers employing conflict-driven search transform the higher-order problem representation to propositional logic. This transformation (called grounding) constitutes the space bottleneck of current ASP systems. Once the grounding step is completed, conflict-driven search in combination with state-of-the-art look-back heuristics like VSIDS and restarts [12] typically shows superior performance compared to other search approaches. Yet, grounding is not possible for many industrial-sized problem instances.

Firgure 1 shows how the size of the grounding explodes for the incremental scheduling problem instances from the ASP competition[2]. For the instances incorporating 120 job operations the size of the grounding is more than 50 GByte.

In industrial scheduling domains the sizes of problem instances are typically significantly higher. Scheduling instances of our project partners Infineon Austria incorporate $>$10000 job operations for a weekly workload performed on $>$100 machines in the back-end (i.e. where chips are cut and packaged) and $>$100000 job operations for a weekly workload performed on $>$1000 machines in the front-end (i.e. where the chips are actually produced). Thus, such instances are clearly out of reach for conventional ASP approaches.

One approach that emerged also out of the need of easing the grounding was Constraint Answer Set Programming (CASP) [14]. CASP can be seen as a hybrid approach extending ASP by Constraint Programming (CP) features. Conceptually, it is very close to satisfiability (SAT) modulo theory approaches, which integrate first-order formulas with additional background theories such as real numbers or integers [16].

For combining ASP and CP there are basically two approaches. First, solvers like Clingcon [15] are based on the extension of the ASP input language in order to support the formulation of constraints. A different approach has been introduced by the Ezcsp solver [4] where ASP and CP are not integrated into one language. ASP rather acts as a specification language for Constraint Satisfaction Problems (CSPs). The main idea is that answer sets constitute CSP encodings, which are used as input for a CP solver.

For certain classes of problems like industrial-sized scheduling CASP was already successfully applied [5]. Especially search problems with large variable domains often profit from the CASP representation due to the alleviation of the grounding bottleneck [13].

Of course, the complexity of problem solving does not vanish by easing the grounding bottleneck but it is rather shifted from grounding in ASP to search in CP. In the context of CASP, often a majority of the solution calculation is done by the CP solver. Hence, the applied search strategies on the CP level play a crucial role for the successful application of CASP in real-world problem domains. However, up to now there was no focus on the development of sophisticated features for expressing and exploiting search strategies in CASP solvers. Consequently, the means for expressing and exploiting search strategies on the CP level are rather limited.

Clingcon[3] and Ezcsp[4] provide a set of built-in problem-

[1] Universität Klagenfurt, Austria, email: firstname.lastname@aau.at

[2] Find instances, encodings and grounders/solvers at www.mat.unical.it/aspcomp2014. Our tests were done with gringo4 and the provided 'new' encoding.

[3] www.cs.uni-potsdam.de/clingcon/

[4] mbal.tk/ezcsp/index.html

independent strategies depending on the underlying CP solver. Clearly, for many real-world problem domains problem-independent strategies are not sufficient and problem-dependent heuristics are needed. Any problem-dependent heuristic on the CP level basically consists of three components:

1. a problem-dependent variable selection strategy
2. a problem-dependent value selection strategy
3. a problem-dependent pruning strategy

In EZCSP, problem-dependent variable selection strategies are already supported by a special label-ordering predicate. What is missing is the possibility of expressing custom value ordering and pruning strategies.

In this paper we present ASCASS, a novel CASP solver, which uses Clingo for answer set solving and the Java framework Jacop for CP solving. ASCASS combines and extends the heuristic possibilities of state-of-the-art CASP solvers and makes them completely available on the problem encoding level. Beyond the usage of built-in strategies, ASCASS provides powerful constructs for the formulation and exploitation of problem-dependent heuristics consisting of variable selection, value selection and pruning strategies.

By means of the real-world Partner Units Problem (PUP), which constitutes one of the hardest benchmarks in the ASP competition, we exemplify problem encoding in ASCASS. We furthermore show how to express the currently most powerful heuristic for this problem in ASCASS. It is shown that due to this heuristic, which, to the best of our knowledge, can not be expressed within any other ASP or CASP approach, ASCASS outperforms state-of-the-art ASP and CASP solvers.

2 Background

In this section we introduce the basic concepts of answer set and constraint answer set programming as it is needed for the purposes of this article. In particular, we ignore disjunctive logic rules and classical negation in ASP for readability reasons. For information about ASP and CASP please refer to [9], [8], [14], [15] and [4].

2.1 Syntax of ASP

in ASP, a *term* refers either to a *variable* or a *constant*. Strings starting with upper case letters denote variables. Constants are represented by strings starting with lower case letters, by quoted strings or by integers. An *atom* is either a *classical atom*, a *cardinality atom* or an *aggregate atom*. A classical atom is an expression $p(t_1, \ldots, t_n)$ where p is an n-ary predicate and $t_1, \ldots, t_n$ are terms. A *negation as failure (NAF) literal* is either a classical atom λ or its negation $not\ \lambda$. A cardinality literal is either a cardinality atom ψ or its negation $not\ \psi$. A *cardinality atom* is of the form

$$l \prec_l \{a_1 : l_{1_1}, \ldots, l_{1_m}; \ldots; a_n : l_{n_1}, \ldots, l_{n_o}\} \prec_u u$$

where

- $a_i : l_{i_1}, \ldots, l_{i_j}$ represent *conditional literals* in which a_i (the heads of the cardinality atom) constitute classical atoms and l_{i_j} are NAF literals
- l and u are terms (i.e. variables or constants) representing non-negative integers. If not specified, the defaults are 0 respectively ∞.
- $\prec_l$ and $\prec_u$ are comparison operators. If not specified, the default is $\leq$.

An aggregate literal is either an aggregate atom φ or its negation $not\ \varphi$. An *aggregate atom* is of the form

$$l \prec_l \#op\{t_{1_1}, \ldots, t_{1_m} : l_{1_1}, \ldots, l_{1_n}; \ldots; t_{o_1}, \ldots, t_{o_p} : l_{o_1}, \ldots, l_{o_q}\} \prec_u u$$

Most syntactical parts of aggregate literals are the same as for cardinality atoms, except that

- a head of a conditional literal is a tuple of terms $t_{i_1}, \ldots, t_{i_j}$ and
- $\#op$ is an aggregate function in $\{\#min, \#max, \#count, \#sum\}$.

Generally, a *rule* is of the form

$$h \leftarrow b_1, \ldots, b_m, not\ b_{m+1}, \ldots, not\ b_n.$$

where

- $h, b_1, \ldots, b_m$ are atoms (i.e. positive literals),
- $not\ b_{m+1}, \ldots, not\ b_n$ are negative literals,
- $H(r) = \{h\}$ is called the *head* of the rule,
- $B(r) = \{b_1, \ldots, b_m, \ldots, not\ b_{m+1}, \ldots, not\ b_n\}$ is called the body of the rule,
- $B^+(r) = \{b_1, \ldots, b_m\}$ is called the positive body of the rule and
- $B^-(r) = \{not\ b_{m+1}, \ldots, not\ b_n\}$ is called the negative body of the rule.

A rule r with $H(r)$ including a cardinality atom is called *choice rule*. A rule r where $B(r) = \{\}$, e.g. '$a \leftarrow$' is called *fact*. For facts, typically '$\leftarrow$' is omitted. A rule r where $H(r) = \{\}$, e.g. '$\leftarrow b$', is called *integrity constraint*, or simply *constraint*.

Furthermore, we allow the typically built-in arithmetic functions ($+$, $-$, $*$, $/$) and comparison predicates ($=$,$\neq$,$<$,$>$,$\leq$,$\geq$).

2.2 Semantics of ASP

The semantics of a non-ground ASP program is defined w.r.t. its *grounding*. A program's grounding can be defined in terms of its Herbrand universe and base. The *Herbrand universe* HU_P of a program P is the set of all constants appearing in P.

The grounding for a rule r without cardinality atoms and aggregates is the set of rules obtained by applying all possible substitutions of variables in r with constants in HU_P. The grounding of a rule, which contains cardinality or aggregate literals, is defined by the two-step instantiation described in [17]: first produce a set of partially grounded rules by substitution of variables occurring outside the cardinality/aggregate literal and then, within each partially grounded rule, substitute each conditional literal by a set of ground conditional literals by substituting the remaining variables inside the cardinality or aggregate literal.

The *grounding* P_G of a program P is the union of all rule groundings. The *Herbrand base* HB_P w.r.t P is the set of all positive NAF literals (i.e. classical atoms) that occur in P_G.

An interpretation I satisfies a (ground) positive NAF literal λ (written as $I \vDash \lambda$) iff $\lambda \in I$. A positive cardinality literal is satisfied by I iff the number of satisfied head literals in the cardinality atom satisfies the lower and upper bounds l and u w.r.t. the order

relations $\prec_l$ and $\prec_u$. Both, bounds and comparison symbols are optional. By default, $0 \leq$ is used for the lower and $\leq \infty$ for the upper bound. A positive aggregate literal is satisfied iff the value returned by the aggregate function $\#op$ applied on the set of term tuples fulfilling its conditions does not violate the lower and upper bounds. Here, $\#count$ counts the number of distinct term tuples fulfilling the related conditions, and $\#min, \#max$ and $\#sum$ are calculating the minimum, maximum or sum of the first terms in the distinct term tuples fulfilling the related conditions. A negative literal $not\ \omega$ is satisfied (written as $I \vDash not\ \omega$) iff ω is not satisfied.

A ground rule r is satisfied by I (written as $I \vDash r$) iff the head is satisfied or the body is not. The body of a rule is satisfied by I iff all literals in the body are satisfied. The head of a rule is satisfied iff the literal in it is satisfied. In particular, an empty body is always satisfied and integrity constraints are satisfied iff the body is not satisfied, i.e. the constraint is not violated. A program P is satisfied by an interpretation I iff all rules in its grounding P_G are satisfied.

An answer set for a program can be defined on the basis of the program's *reduct* [10, 17]. The reduct P^I of a ground program P relative to an interpretation $I \subseteq HB_P$ is defined as $P^I := \{H(r) \leftarrow B(r)^+ : r \in P, B(r)^- \cap I = \emptyset\}$.

An interpretation $I \subseteq HB_P$ (which may be empty) is an *answer set* for a program P not containing choice rules iff

- I satisfies all rules r in P^I, i.e. $\forall r \in P^I$: $I \vDash r$ and
- I is subset-minimal, i.e. there is no $I' \subset I$ so that I' satisfies all rules in $P^{I'}$.

Choice rules can produce answer sets that are not subset-minimal, which leads to a slight change of semantics when such rules are present. For example, the program consisting only of the choice rule $\{a\}.$ possesses the two answer sets $\{\}$ and $\{a\}$. In order to be in line with the original semantics and thus restore subset-minimality an equivalent program can be produced by extending the program as follows:

For every head a_i within a cardinality atom of a choice rule, add a new atom a_i' (which is not occurring elsewhere in the program) and a constraint $\leftarrow a_i, a_i'$. Informally, a_i' expresses that a_i is not in the interpretation. This way, the choice rule $\{a\}.$ equivalently produces the two answer sets $\{a'\}$ and $\{a\}$. For convenience, we can imagine the new atoms a_i' and the constraints $\leftarrow a_i, a_i'$ to be invisible. For details, consult [8].

An ASP program is *unsatisfiable* iff it has no answer sets and *satisfiable* otherwise.

2.3 Constraint Answer Set Programming

A constraint satisfaction problems (CSP) can be defined as a three-tuple $\langle V, D = \{dom(v) : v \in V\}, C\rangle$ whereby V is a set of variables, D is the set of domains of the variables in V and C is a set of constraints on variables in V. A solution to a CSP is an assignment $\forall v \in V, v := d \in dom(v)$ such that all constraints $c \in C$ are fulfilled. A CSP comprising only finite domains is called finite. If all domains are defined over discrete values (most commonly integers), the CSP is called discrete.

For integrating CP into ASP there are basically two approaches. First, solvers like Clingcon [15] are based on the extension of the ASP input language in order to support the definitions of constraints. Take as a simple example the following encoding in Clingcon (':-' represents left-implication and '!=' represents $\neq$):

```
num(N):-N=1..3.

$domain(1..6).

var(X) $+ var(Y) $+ var(Z) $== 6 :-
  num(X), num(Y), num(Z), X!=Y, Y!=Z, X!=Z.

var(1) $> 1.

$distinct{var(N):num(N)}.
```

The above encoding expresses that the sum of the three CSP variables $var(1)$, $var(2)$ and $var(3)$ must be equal to six. The domain of the variables is 1..6. Furthermore, $var(1)$ must be greater than one and all CSP variables must be distinct to each other. The ASP and CSP are fully integrated into one language. CSP specific constructs are indicated by $, like $+ or $==. $*distinct* constitutes a well known global constraint, i.e. a constraint over a set of variables. In Clingcon CSP variables are not defined explicitly but indirectly by the constraints. CP solving is integrated in the answer set production process and carried out by the Gecode solver. For more information on Clingcon please refer to [15].

The Ezcsp solver [4] is based on a different approach where ASP and CP are not integrated into one language. ASP rather acts as a specification language for Constraint Satisfaction Problems (CSPs). The main idea is that answer sets constitute CSP encodings, which are used as input for a CP solver. The above example can be expressed in Ezcsp as:

```
num(N):-N=1..3.

cspdomain(fd).

cspvar(var(N),1,6):-num(N).

required(var(X) + var(Y) + var(Z) == 6):-
  num(X), num(Y), num(Z), X!=Y, Y!=Z, X!=Z.

required(var(1) > 1).

required(all_distinct([var/1]).
```

After some pre-processing, an answer set is calculated which includes $cspdomain$-, $cspvar$- and $required$ facts. $cspdomain(fd)$ denotes that the CSP is finite and discrete. Ezcsp is also able to handle real domains. CSP variables are explicitly defined by $cspvar$ facts also defining lower and upper bounds of the variable domains. Constraints are represented as $required$ facts. For expressing global constraints, and thus refer to sets of CSP variables, Ezcsp allows the usage of functional symbols. E.g. $[var/1]$ refers to all variables formed by the unary function var. Once an answer set has been produced, the CSP encoded within the $cspdomain$-, $cspvar$- and $required$ facts is passed to the CP solver. As answer set production and CSP solution search are two separated processes, different CP solvers can be used in Ezcsp. Currently, Sicstus- and B-Prolog are supported.

The semantics of a program builds on the notion of *extended answer sets* [4]: A pair $\langle A, S\rangle$ is an extended answer set of program Π iff A is an answer set of Π and S is a solution to the CSP defined by A. We further define that the empty CSP (i.e. without any CSP variables) possesses the empty solution.

For CSP solution search, Ezcsp provides different search strategies impacting the underlying CP solver. In case of Sicstus Prolog as a CP solver, the built-in value selection strategies *step* (min domain value,

when ascending order is used, max domain value when descending order is used) and *bisect* (bisection of the domain in the middle) are available. Similarly in case of B-Prolog, the bisection strategies *split* and *reverse_split* are supported. The supported variable selection strategies are *leftmost* (leftmost variable), *min* (leftmost variable with minimal lower bound), *max* (leftmost variable with maximal upper bound), and *ff* (first-fail). By the special *label_order/2* predicate it is also possible to define problem-dependent CSP variable orderings for the CP solver. However, there are no constructs for expressing problem dependent value or pruning strategies.

3 A Simple Constraint Answer Set Solver

In the following we introduce our novel CASP solver ASCASS. First, we give a brief introduction to the overall architecture of ASCASS and the encoding of CSPs in ASCASS. After that, we present the means for formulating problem-dependent heuristics. This constitutes the significant novel feature of our solver and the main contribution of this article.

3.1 Architecture

ASCASS[5] is a finite discrete CASP solver following the approach of Ezcsp, i.e. the input language is pure ASP and the answer sets encode CSPs. Figure 2 shows the overall architecture of ASCASS. Answer set production (grounding and solving) is done by Clingo[6], which is currently one of the most powerful ASP systems. The input language is the ASP standard ASP-Core-2[7].

After answer set solving, a produced answer set is handed over to a parsing module that extracts the facts which encode the CSP and search directives. This information is used to instantiate a corresponding CSP in the CP solver and perform search conforming to the given search directives. Currently, Jacop[8] is used within ASCASS as a CP solver. In case that the CSP could not be solved by the CP solver or a timeout occurred (defined by the special predicate $csptimeout(\Delta)$), the process continues with the next answer set, until a solution is found, or there are no more answer sets. The empty CSP (i.e. when there is not a single CSP variable) is always satisfiable and possesses the empty CSP solution.

3.2 Encoding of CSPs

ASCASS focuses on finite discrete Constraint satisfaction problems (CSPs). In order to encode a CSP within ASCASS there can be used a number of specific predicates. Of course, in the input these predicates can contain variables. The following explanations refer to their grounded form.

The predicates $cspvar(\alpha, \lambda, \upsilon)$ and $cspvar(\alpha, \lambda, \upsilon, \eta)$ are responsible for encoding CSP variables. Hereby, α represents the variable name and λ and υ represent respectively the numerical lower and upper bound of the variable's domain. For example $cspvar(x, 1, 10)$ stands for a CSP variable v with the domain $[1..10]$. The numerical priority η is used to define a custom variable selection ordering. When using the variable selection strategy $priority$ (see below), the CP solver selects the variable with the highest priority first.

The predicate $cspconstr(\alpha, \rho, \tau)$ encodes a relational constraint (i.e. $=, <>, <, <=, >, >=$) over a variable α. ρ denotes the type of relation and must be a constant out of $\{eq, neq, lt, lteq, gt, gteq\}$. τ denotes another CSP variable or a numerical constant. For example, $cspconstr(x, lt, 5)$ expresses that variable x must be lower than 5.

The predicate $csparith(\alpha, \pi, \beta, \rho, \gamma)$ encodes arithmetic constraints. α, β and γ are CSP variable names. Like for $cspconstr$, the constant ρ denotes the type of relation. π is a constant representing an arithmetic operation. Currently, ASCASS supports addition ($plus$), subtraction ($minus$), multiplication ($mult$), division (div) and exponent (exp). For example, $csparith(xa, plus, xb, eq, xc)$ states that the sum of the values of xa and xb must be equal the value of xc.

For expressing logical constraints predicates of the form $cspif(\Xi_1, and, \Xi_2, and, \ldots, and, \Xi_m, then, \Xi_{m+1}, or, \Xi_n)$ can be used. Each Ξ consists of a variable α, a relational symbol ρ and another variable or numerical constant τ. For example, $cspif(x, lt, 5, and, y, gt, 10, then, z, gteq, 0)$ is to be read as 'if x is lower than 5 and y is greater than 10 then z must be non-negative'.

Global constraints are constraints over arrays of variables. In ASCASS global constraints are defined by predicates of the form $cspglobal(\sigma_1, \ldots, \sigma_m, \kappa)$ and $cspglobal(\sigma_1, \ldots, \sigma_m, \kappa, \tau_1, \ldots, \tau_n)$. κ is a constant denoting the type of global constraint. $\sigma_1, \ldots, \sigma_m$ represent arrays of variables. $\tau_1, \ldots, \tau_n$ represent single CSP variables or integers. The selection of global constraints currently supported by ASCASS has been determined by the needs of our application areas and will be further expanded. ASCASS currently supports the following global constraints[9]:

- min: $cspglobal(\sigma, min, \tau)$, the minimum value of the variables σ is equal to τ
- max: $cspglobal(\sigma, max, \tau)$, the maximum value of the variables σ is equal to τ
- sum: $cspglobal(\sigma, sum, \tau)$, the sum of values of the variables σ is equal to τ
- count: $cspglobal(\sigma, count, \tau_1, \tau_2)$, τ_1 is equal to the counted number of variables in σ with value τ_2
- global cardinality: $cspglobal(\sigma_1, \sigma_2, gcc)$, a more general counting constraint where the occurring values in σ_1 are counted in the corresponding counter variables in σ_2
- all different: $cspglobal(\sigma, alldiff)$, all variables in σ are mutually unequal
- element: $cspglobal(\sigma, element, \tau_1, \tau_2)$, the value of the τ_1-th variable in σ is equal to τ_2
- cumulative: $cspglobal(\sigma_1, \sigma_2, \sigma_3, cumulative, \tau)$, σ_1 represents the starting times of $|\sigma_1|$ many jobs, σ_2 represents the durations of the jobs, σ_3 represents the amounts of needed resources of the jobs and τ represents the allowed accumulated amount of resources at any time point
- bin packing: $cspglobal(\sigma_1, \sigma_2, \sigma_3, binpacking)$, σ_1 represents bin assigments for $|\sigma_1|$ many items, σ_2 represents the bin sizes of the $|\sigma_2|$ many bins and σ_3 represents the item sizes

In order to address arrays of CSP variables, ASCASS not only allows simple constants but also n-ary functional terms for variable names of the form $\phi(\iota_1, \ldots, \iota_n)$ with $\iota_1, \ldots, \iota_n$ representing string or integer arguments (see Figure 3). The special

[5] http://isbi.aau.at/hint/ascass
[6] sourceforge.net/projects/potassco/files/clingo
[7] www.mat.unical.it/aspcomp2013/files/ASP-CORE-2.03b.pdf
[8] jacop.osolpro.com

[9] More information about global constraints can be found at http://jacop.osolpro.com/guideJaCoP.pdf and http://sofdem.github.io/gccat/

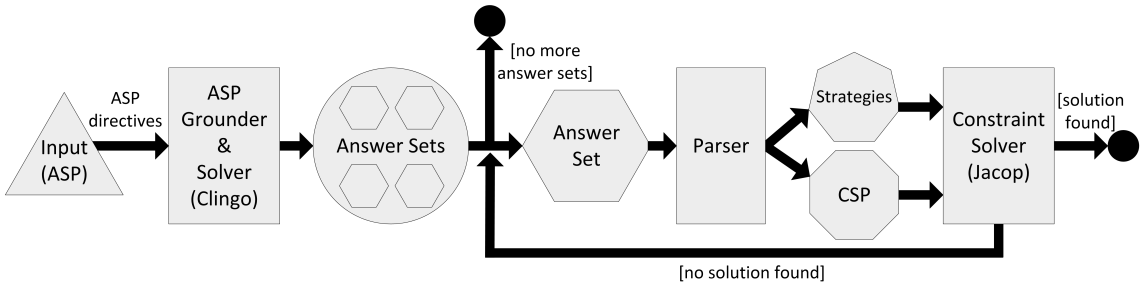


Figure 2. Architecture of ASCASS

functional argument *all* acts as a placeholder and can be used for addressing arrays of variables. For example, take the four variable definitions $cspvar(v(1,1),1,10)$, $cspvar(v(1,2),1,10)$, $cspvar(v(2,1),1,10)$ and $cspvar(v(2,2),1,10)$. A natural interpretation of the arguments is *row* and *column* of a two-dimensional variable array. Consequently, $cspglobal(v(all,2),alldiff)$ expresses that the values of all second column's variables, in our case $v(1,2)$ and $v(2,2)$, must be different to each other. *v(all,all)* stands for all variables in the two-dimensional array, i.e. all variables formed by the functional symbol v with arity 2.

v(1,1)	v(1,2)	...	v(1,c)	v(1,all)
v(2,1)	v(2,2)	...	v(2,c)	v(2,all)
...	...	...	...	...
v(r,1)	v(r,2)	...	v(r,c)	v(r,all)
v(all,1)	v(all,2)	...	v(all,c)	v(all,all)

Figure 3. Concept of variable arrays in ASCASS

3.3 Encoding of variable selection strategies

Apart from the predicates for defining a CSP, ASCASS provides predicates for steering the search of the CP solver. The predicates $cspvarsel(\epsilon)$ and $cspvarsel(\epsilon,\theta)$ define the variable selection strategy to be used. Herby, ϵ is the primary selection strategy and, if defined, θ acts as a secondary, tiebreaking strategy. For variable selection, ASCASS currently supports the problem-independent built-in strategies *smallestDomain*, *mostConstrainedStatic*, *mostConstrainedDynamic*, *smallestMin*, *largestDomain*, *largestMin*, *smallestMax*, *maxRegret*, *weightedDegree* and the problem-dependent strategy *priority*.

When using the *priority*-strategy, ASCASS builds an ordering of the CSP variables based on the provided priorities η in $cspvar(\alpha,\lambda,\upsilon,\eta)$. Variables with high priorities are selected first. Variables for which there is no η defined are selected as the last ones. Hence, the $priority$ strategy in combination with the variable priorities is similar to the *label_order* predicate in Balduccini's EZCSP.

3.4 Encoding of value selection strategies

For value selection ASCASS provides the predicates $cspvalsel(\phi)$ and $cspvalsel(\phi,\varphi)$ where ϕ and φ are constants denoting the strategy. As it is often important to have different value selection strategies for different sets of variables, ASCASS provides also the predicates $cspvalsel(\sigma,\phi)$ and $cspvalsel(\sigma,\phi,\varphi)$ where σ represents an array of variables like in global constraints. ASCASS supports the already built-in strategies *indomainMin*, *indomainMiddle*, *indomainMax* and *indomainRandom*. For expressing problem-dependent value selection strategies, the novel strategy *indomainPreferred* can be used.

When using *indomainPreferred*, the CP solver first tries to use specified values before changing to the built-in strategy φ (*minDomain* if not stated otherwise). For specifying preferred values, ASCASS provides the special predicate $csppprefer(\alpha,\rho,\tau)$ and $csppprefer(\alpha,\rho,\tau,\eta)$. Like for relational constraints, α represents a CSP variable, ρ represents a relational symbol and τ stands for a further variable or a numerical constant. For example, $csppprefer(v,eq,5)$ states that for the CSP variable v a preferred value is 5. In order to specify an ordering of the specified values, it is possible to make use of a numerical priority η. Higher priority statements are taken into account first by ASCASS. For example, if there is given $csppprefer(v,eq,5,1)$ and $csppprefer(v,eq,20,2)$, ASCASS tries to first label v with 20 and only after that with 5. Of course, only preferred values, which are still in the variable's domain, are taken into account. In case that τ denotes another variable, the minimum value in the current domain of τ is used as a preferred value, i.e. τ does not need to be singleton for specifying a preferred value of α. This in combination with global constraints is a highly dynamic and powerful mechanism.

As with the relational constant eq in combination with the priorities η every ordering of preferred values can be expressed, the usage of lt, $lteq$, gt and $gteq$ can be clearly seen as syntactic sugar. By using lt, $lteq$, gt and $gteq$ sets of preferred values can be expressed:

- $lteq\ \tau : \{\tau, \tau-1, \ldots, -\infty\}$
- $lt\ \tau : \{\tau-1, \ldots, -\infty\}$
- $gteq\ \tau : \{\tau, \tau+1, \ldots, \infty\}$
- $gt\ \tau : \{\tau+1, \ldots, \infty\}$

Note that all preferred values of such a set P have the same priority (possibly given explicitly by η). For defining an order relation over P, i.e. fix the order in which ASCASS considers the preferred values in P, the following holds: For lt and $lteq$ decreasing order is used, i.e. $\tau, \tau-1, \ldots, -\infty$ and for gt and $gteq$ increasing order is used, i.e. $\tau, \tau+1, \ldots, \infty$. For example having the variable definition $cspvar(v,1,10)$ and the value selection strategy $cspvalsel(indomainPreferred, indomainMin)$, $csppprefer(v,lt,5)$ would effect that ASCASS considers the domain values in the following order: $4,3,2,1,5,6,7,8,9,10$. The reason

why for lt and $lteq$ descending order and for gt or $gteq$ ascending order is used is simply the following: Would it be the other way round, the behavior with lt and $lteq$ would conform to $indomainMin$ and with gt and $gteq$ to $indomainMax$.

3.5 Encoding of pruning strategies

The third component of many problem-dependent heuristics is the pruning strategy. For specifying how a search tree is pruned, ASCASS provides the special predicate $cspsearch(\omega, \mu)$. Hereby, ω specifies the pruning type and μ specifies a numerical limit that, when reached, triggers backtracking. Again it could be beneficial having different limits for different groups of variables or even having no limit on certain variables whilst search on others is limited. To this, ASCASS provides the predicate $cspsearch(\sigma, \omega, \mu)$ with σ denoting an array of variables like for global constraints.

Currently, ASCASS provides two pruning types. $cspsearch(limited, \mu)$ limits the number of wrong decisions for variables. If the number μ of wrong choices for a variable is reached, backtracking is triggered and the counter for the variable is reset. For example, $cspsearch(limited, 3)$ specifies that for every variable v there must not be more than three labeling trials for v within a search branch. The second pruning type is based on limited discrepancy search [11] and operates on the level of search paths. When specifying $cspsearch(lds, \mu)$ only a certain number of wrong decisions (called discrepancies) along the whole search path is allowed. If this number reaches μ, backtracking is triggered.

Furthermore, it is possible to limit search time of the CP solver by $csptimeout(\Delta)$ where Δ is the number of seconds when the timeout is triggered. The timeout concerns only the search of Jacop so that search might start over based on the next answer set if such exists.

3.6 Directives for answer set production

In order to specify which heuristic is to be used by Clingo, the special ASCASS predicate $aspheuristic(\nu)$ can be used. Hereby, ν is a constant denoting the heuristic, which is passed to Clingo as a command line option $--heur = \nu$. As this happens before the actual answer set solving, $aspheuristic(\nu)$ must only be used as a single fact within program source code. Common heuristics to be used are $VSIDS$ or $Berkmin$ [12]. When using $aspheuristic(domain)$, Clingo uses a user-defined heuristic defined via the special predicate $\_heuristic$ [7]. For limiting the number of produced answer sets, the special predicate $aspnumas(\Delta)$ can be used. Δ is a non-negative integer and is passed to Clingo as a command line option. The default is '1' and '0' effects the production of all answer sets. Like $aspheuristic$, also $aspnumas$ must only be used as a single fact within the problem source code. Similarly, $asptimeout(\Delta)$ specifies a timeout for answer set solving.

4 Proof of concept

In this section we exemplify the programming in ASCASS and give first evaluation results for the real-world partner units configuration problem [3, 1]. We want to emphasize that the focus is on the formulation of problem-dependent heuristics.

4.1 The Partner Units Problem

We want to exemplify the expressive power of ASCASS with respect to the partner units problem (PUP) out of three reasons.

1. The PUP is a real world combinatorial problem with many different application domains [2].
2. The PUP is one of the hardest benchmark problems participating in the ASP competitions[10].
3. There exists an effective and non-trivial problem-dependent heuristic to solve the PUP.

The PUP originates in the domain of railway safety systems. One of the problems in this domain is to make sure that certain rail tracks are not occupied by a train/wagon before another train enters this track. The signals for the corresponding occupancy indicators are calculated by special processing units based on the input of several observing sensors. Because of fail-safety and realtime requirements the number of sensors respectively indicators, which can be connected to the same unit, is limited (called unit capacity, UCAP). Also one sensor/indicator device can only be directly connected to one unit. However, a unit can be connected to a limited number (called inter unit capacity, IUCAP) of other units. These units are called the partner units of the unit. Devices (i.e. sensors and indicators) can only communicate with devices connected to the same unit and with devices connected to one of the partner units. Given the IUCAP, UCAP and a bipartite input graph represented by edges specifying which sensor data is needed in order to calculate the correct signal of an occupancy indicator, the problem consists in connecting sensors/indicators with units and units with other units such that all communication requirements are fulfilled and IUCAP and UCAP are not violated.

The state-of-the-art heuristic for solving PUP is the QuickPup heuristic proposed in [18]. QuickPup is based on three major techniques. First, based on the input graph and a distinguished root indicator, QuickPup produces a topological ordering of the devices, which is basically the minimum distances from the root indicator to all other devices. The distance to itself is zero, the distance to the direct neighbors is one, the distance to the neighbors of the neighbors is two and so forth. This reflects the (partial) ordering in which the devices should be processed. Second, for each device, first try to place it on the next empty unit and if this is unsuccessful try the already used units in descending order. Third, try different root indicators, and consequently different topological orderings, and limit search for each trial. The intuition behind that is that not all root indicators are equally good to start search from.

The input comprises of a set of $egde(i, s)$ facts where i takes the numerical identifier of an indicator and s takes the identifier of a sensor. Additionally the input includes a fact $ucap(x)$ with $x > 0$ that defines the unit capacity (UCAP) and a fact $iucap(y)$ with $y > 0$ that defines the inter-unit capacity (IUCAP).

For the code snippets given in the remainder of this section we use the standard notation of logic programming. In particular, left-implication $\leftarrow$ is represented as ':-'.

In order to produce explicit indicator and sensor information the following lines of code are used:

```
sensor(S):-edge(I,S).
indicator(I):-edge(I,S).
numIndicators(N):-N=#count{I:indicator(I)}.
numSensors(N):-N=#count{S:sensor(S)}.
```

The number of indicators ($numIndicators$) respectively sensors ($numSensors$) are calculated by means of the $\#count$ aggregate literal provided by Clingo.

[10] Further information can be found at www.mat.unical.it/aspcomp2014/

We restrict the number of units ($numUnits$) available for a solution to the theoretical lower bound, i.e. $numUnits = \left\lceil \frac{max(numIndicators, numSensors)}{UCAP} \right\rceil$:

```
max(M):-numIndicators(E),numSensors(F),
                                    M=#max(E;F).
numUnits(N):-max(M),ucap(C),N=(M+1)/C.
unit(Z):-numUnits(N),1<=Z,Z<=N.
```

For each indicator i there is a CSP variable $device(i, 1)$ and for each sensor s there is a CSP variable $device(s, 2)$. This way it is also possible to refer to the array of all CSP device variables as $device(all, all)$, to only the indicator variables as $device(all, 1)$ and to the sensor variables as $device(all, 2)$ which will be useful later. The value range for these CSP variables is $[1..numUnits]$. Furthermore, the variables get a priority defining the topological order in which they are labeled by ASCASS:

```
cspvar(device(I,1),1,N,P):-
                    numUnits(N),iPriority(I,P).
cspvar(device(S,2),1,N,P):-
                    numUnits(N),sPriority(S,P).
```

The calculation of the priorities is explained in detail below.

In order to assure UCAP, for each unit u there are two counting variables $ci(u)$ and $cs(u)$. These variables can take values in the range $[0..UCAP]$. Furthermore, for each unit u there are two $count$ global constraints counting the number of indicator respectively sensor variables taking the value u:

```
cspvar(ci(U),0,C):-ucap(C),unit(U).
cspvar(cs(U),0,C):-ucap(C),unit(U).
cspglobal(device(all,1),count,ci(U),U):-
                                        unit(U).
cspglobal(device(all,2),count,cs(U),U):-
                                        unit(U).
```

In order to capture which unit $u1$ is connected to which unit $u2$ there are $numUnits \times numUnits$ many CSP variables (i.e. $conn(U1, U2)$). The variables can take values in the range $[0..1]$ if $u1 <> u2$. Otherwise, the variables' ranges consists of only a single value, i.e. $[1..1]$. This is because in our model each unit u is always connected to itself. Furthermore, there is a constraint assuring symmetry, i.e. if $u1$ is connected to $u2$ also $u2$ is connected to $u1$:

```
cspvar(conn(U1,U2),0,1):-unit(U1),unit(U2),
                                      U1<>U2.
cspvar(conn(U,U),1,1):-unit(U).
cspconstr(conn(U1,U2),eq,conn(U2,U1)):-
                      unit(U1),unit(U2),U1<U2.
```

For summing up how many units are connected to a unit u we make use of the global sum constraint. The used summing variables can hereby take values in the range $[1..IUCAP + 1]$ as every unit is also connected to itself:

```
cspvar(sumconns(U),1,K+1):-iucap(K),unit(U).
cspglobal(conn(U,all),sum,sumconns(U)):-
                                        unit(U).
```

In order to make the summing variables and constraints take effect, it must be assured that any connection variable $conn(u1, u2)$ is set to one whenever there is an $edge(i, s)$ in the input so that $device(i, 1) = u1$ and $device(s, 2) = u2$. Following the approach of [6], this is implemented by means of the global $element$ constraint. Given an array of CSP variables arr, an index i and a value v, an $element$ constraint assures that the i^{th} variable in arr is equal to v. In our case, for each $edge(i, s)$ in the input there is such a global constraint setting the appropriate connection variable within $conn(all, all)$ to one:

```
cspglobal(conn(all,all),element,index(I,S),1)
                                    :-edge(I,S).
```

As the $element$ constraint cannot directly handle multi-dimensional arrays, the respective index is calculated as $index(i, s) = (device(i, 1) - 1) \times numUnits + device(s, 2)$. The formulation with constraints is straightforward.

The priorities for the device variables (i.e. $device(i, 1)$ and $device(s, 2)$ are based on a topological ordering of the devices. Given the layer of a sensor or indicator whereby the root of the topological graph is at layer zero, the priority is higher the lower the layer is:

```
iPriority(I,P):-indicatorLayer(I,L),P=9999-L.
sPriority(S,P):-sensorLayer(S,L),P=9999-L.
```

The effect is that, given a root indicator, ASCASS first tries to label the root indicator, then the neighbors of the root indicator, then the neighbors of the neighbors, and so on. In our implementation a choice rule is used to express that there is exactly one distinguished indicator that acts as root. This indicator is always placed at the first unit:

```
1{root(I):indicator(I)}1.
cspconstr(device(I,1),eq,1):-root(I).
```

The choice rule $1\{root(I) : indicator(I)\}1$ produces one answer set for each root indicator and asserts a $root(i)$ fact.

For calculating the actual layers, we first calculate the minimum distances to the root whereas root indicator has a zero distance to itself[11]:

```
indicatorDist(I0,0):-root(I0).
sensorDist(S,D+1):-indicatorDist(I,D),
                   edge(I,S),numDevices(M),D<M.
indicatorDist(I,D+1):-sensorDist(S,D),
                   edge(I,S),numDevices(M),D<M.
numDevices(N):-numIndicators(E),numSensors(F),
                                      N=E+F.
```

The layers are calculated by using the $\#min$ aggregate literal from Clingo:

```
indicatorLayer(I,Dmin):-indicator(I),
              Dmin=#min{D:indicatorDist(I,D)}.
sensorLayer(S,Dmin):- sensor(S),
               Dmin = #min{D:sensorDist(S,D)}.
```

First to try to place devices on unused units and, only if not successful, on used units in descending order can be expressed in ASCASS by means of preferred values:

```
cspprefer(device(I,1),lteq,nextUnit):-
                                  indicator(I).
cspprefer(device(S,2),lteq,nextUnit):-
                                    sensor(S).
```

[11] In order to make grounding safe, we have to limit the maximum possible distance, which is equal to the total number of devices.

The CSP variable $nextUnit$ points to the next unused unit, which is the current unit plus one[12]:

```
cspvar(curUnit,1,N):-numUnits(N).
cspvar(nextUnit,1,N+1):-numUnits(N).
csparith(curUnit,plus,one,eq,nextUnit).
```

For the calculation of the current unit, i.e. the highest number taken by some $device(i,1)$ or $device(s,2)$ variable, the global max constraint is used:

```
cspglobal(device(all,all),max,curUnit).
```

As ASCASS uses the lower bound of variables for calculating the preferred values, each device variable is first tried to be bound to values lower than or equal to the lower bound of $nextUnit = curUnit + 1$ in descending order.

In order to control how many units are maximally tried per device variable, the search is pruned such that only the next unit and a limited number of already used units can be tried before backtracking is triggered. In our implementation we use the following statement for only trying the next, the current and the last unit:

```
cspsearch(limited,3).
```

We furthermore restrict the maximum CSP search time for each call of the CP solver in order to try different start indicators:

```
csptimeout(300).
```

For making ASCASS respect the problem-dependent selection strategies, $cspvarsel(priority)$ and $cspvalsel(device(all,all),indomainPreferred)$ must be included. Thus, the concepts of QuickPup can be fully expressed in a declarative way by ASCASS. To the best of our knowledge, this is not possible within any other ASP or CASP approach.

4.2 Evaluation

We tested the ASP solver Clingo 4 and the CASP solvers ASCASS, Clingcon and Ezcsp on the PUP benchmark suite used in [2][13]. Clingo was tested using the PUP encoding proposed in [2][14]. The tests were run on a 3.2 Ghz machine with 64 GByte of RAM, assuring that the grounding bottleneck does not play a role for the tested instances[15] and performance can be attributed to the search phase.

In the Clingcon model, problem-dependent CSP variable selection, value selection or pruning strategies cannot be exploited. For Ezcsp, it is possible to express the topological variable orderings similar to ASCASS. However, there are no means for pruning search or problem-dependent value strategies.

Table 1 depicts how many instances of each type in the benchmark suite could be solved by the different approaches within a 1000 seconds time frame. Clingo using VSIDS heuristic peformed very well on the benchmark suite showing once again that the conflict-driven search techniques employed by Clingo are quite powerful. Also Ezcsp was able to solve some instances. Using other built-in heuristics did not result in better performance. Clingcon was not able to solve a single instance. In the contrary, ASCASS was able to solve all but one instances within time limits. We want to point out that only optimal solutions (i.e. minimum number of units) were allowed for easing the grounding bottleneck of conventional ASP. Increasing the number of allowed units in a solution would increase grounding size for ASP significantly. In the cases of ASCASS and Ezcsp, increasing the number of allowed units would not affect the grounding size as the number of allowed units is captured by the upper bounds of the CSP variables.

	#	Clingo	ASCASS	Clingcon	Ezcsp
double(IUCAP=2)	10	2	10	0	2
doublev(IUCAP=2)	6	3	6	0	0
triple(IUCAP=2)	3	2	3	0	2
triple(IUCAP=4)	7	6	6	0	3
grid(IUCAP=4)	10	10	10	0	0
total	36	23	35	0	7

Table 1. Solved instances whithin 1000 seconds

Furthermore, we want to make clear that the superior performance of ASCASS can be attributed to the inclusion of the QuickPup strategies. This was crosschecked by removing the heuristic parts from the ASCASS problem encodings. It is to be noted that QuickPup originally was designed for producing only near-optimal solutions. However, the concepts of QuickPup obviously also work well for finding optimal solutions.

5 Conclusions

Elaborated engineering and sophisticated general problem-independent heuristics have significantly improved the runtime performance of general problem-solvers. However, it is a well known observation that general problem-solvers which are applied to $\mathcal{NP}$-hard problems deliver satisfiable performance up to a certain size of the problem instances. For many problems special heuristics were developed which resulted in exceptional runtime improvements compared to state-of-the-art general problem-solvers.

With ASCASS we provide a constraint answer set programming solver, which allows the declarative formulation of problem-specific heuristics. In particular, we exploit ASP to generate problem-specific heuristics for CSP including variable and value selection as well as pruning strategies. By employing the real-world Partner Units Problem, which constitutes one of the hardest benchmark problems of the ASP competitions, we exemplified an encoding in ASCASS that includes both the declarative description of the problem and an effective heuristic for solving the problem. By this encoding we could demonstrate that the non-trivial problem-dependent QuickPup heuristic can be expressed quite naturally in ASCASS. Due to this heuristic, which cannot be expressed by any other ASP or CASP system, ASCASS clearly outperforms state-of-the-art ASP or CASP systems on the tested instances.

ACKNOWLEDGEMENTS

Work has been funded by the Austrian Research Fund (FFG) in the context of project Heuristic Intelligence (*HINT*, FFG-PNr.: 840242).

[12] Within the constraint, the helping variable $cspvar(one,1,1)$ is used as arithmetic constraints only accept variables in ASCASS.

[13] Encodings and benchmark instances can be found at http://isbi.aau.at/hint/ascass

[14] The 'new' encoding provided by the ASP competition 2014 was found to be inconsistent as it also produces answer sets for unsatisfiable instances.

[15] The biggest grounding in the ASP model was $\sim$ 12 GByte.

REFERENCES

[1] Falkner A., A. Haselboeck, G. Schenner, and H. Schreiner, 'Modeling and solving technical product configuration problems', *AI EDAM*, 115–129, (2011).

[2] M. Aschinger, C. Drescher, G. Friedrich, G. Gottlob, P. Jeavons, A. Ryabokon, and E. Thorstensen, 'Optimization methods for the partner units problem', in *CPAIOR'11*, pp. 4–19, Berlin, Heidelberg, (2011). Springer-Verlag.

[3] M. Aschinger, C. Drescher, G. Gottlob, P. Jeavons, and E. Thorstensen, 'Tackling the partner units configuration problem.', in *IJCAI'11*, pp. 497–503, (2011).

[4] M. Balduccini, 'Representing constraint satisfaction problems in answer set programming', in *ICLP09 Workshop on Answer Set Programming and Other Computing Paradigms (ASPOCP'09)*, (2009).

[5] Marcello Balduccini, 'Industrial-size scheduling with asp+cp', in *Proceedings of the 11th Int. Conf. on Logic Programming and Nonmonotonic Reasoning*, LPNMR'11, pp. 284–296, Berlin, Heidelberg, (2011). Springer-Verlag.

[6] Conrad Drescher, 'The partner units problem: A constraint programming case study', in *ICTAI'12*, (2012).

[7] M. Gebser, B. Kaufmann, J. Romero, R. Otero, T. Schaub, and P. Wanko, 'Domain-specific heuristics in answer set programming', in *27th AAAI Conf. (AAAI'13)*, pp. 350–356, (2013).

[8] Martin Gebser, Roland Kaminski, Benjamin Kaufmann, and Torsten Schaub, *Answer Set Solving in Practice*, Synthesis Lectures on Artificial Intelligence and Machine Learning, Morgan and Claypool Publishers, 2012.

[9] Michael Gelfond and Vladimir Lifschitz, 'The stable model semantics for logic programming', in *Proceedings of the Fifth Int. Conf. and Symp. of Logic Progr. (ICLP'88)*, eds., R. Kowalski and K. Bowen, pp. 1070 – 1080. MIT Press, (1988).

[10] Michael Gelfond and Vladimir Lifschitz, 'The stable model semantics for logic programming', in *Proceedings of the Fifth International Conference and Symposium of Logic Programming (ICLP'88)*, eds., R. Kowalski and K. Bowen, pp. 1070 – 1080. MIT Press, (1988).

[11] William D. Harvey and Matthew L. Ginsberg, 'Limited discrepancy search', in *Proceedings of the 13th International Joint Conf. on Artificial Intelligence*, pp. 607–613. Morgan Kaufmann, (1995).

[12] Matthew D. T. Lewis, Tobias Schubert, and Bernd W. Becker, 'Speedup techniques utilized in modern sat solvers', in *Proceedings of the 8th Int. Conf. on Theory and Applications of Satisfiability Testing*, SAT'05, pp. 437–443, Berlin, Heidelberg, (2005). Springer-Verlag.

[13] Yuliya Lierler, Shaden Smith, Miroslaw Truszczynski, and Alex Westlund, 'Weighted-sequence problem: Asp vs casp and declarative vs problem-oriented solving', in *Proceedings of the 14th Int. Conf. on Practical Aspects of Declarative Languages*, PADL'12, pp. 63–77, Berlin, Heidelberg, (2012). Springer-Verlag.

[14] Veena S. Mellarkod, Michael Gelfond, and Yuanlin Zhang, 'Integrating answer set programming and constraint logic programming', *Annals of Mathematics and Artificial Intelligence*, **53**(1-4), 251–287, (August 2008).

[15] Max Ostrowski and Torsten Schaub, 'Asp modulo csp: The clingcon system', *Theory Pract. Log. Program.*, **12**(4-5), 485–503, (September 2012).

[16] Roberto Sebastiani, 'Lazy satisfiability modulo theories', *J. on Satisfiability, Boolean Modeling and Computation*, **3**, 141–224, (2007).

[17] Tommi Syrjänen, 'Cardinality constraint programs', in *JELIA 2004*, volume 3229 of *Lecture Notes in Computer Science*, pp. 187–199. Springer, (2004).

[18] Erich Christian Teppan, Gerhard Friedrich, and Andreas Falkner, 'Quickpup: A heuristic backtracking algorithm for the partner units configuration problem.', in *International Conference on Innovative Applications of AI (IAAI'12)*, pp. 2329–2334. AAAI, (2012).