Peter Kello
University of Žilina, Department of Control and Information Systems,
Univerzitná 8215/1,010 26 Žilina, Slovak Republic, *peter.kello@fel.uniza.sk*

Michal Gregor
University of Žilina, Department of Control and Information Systems,
Univerzitná 8215/1,010 26 Žilina, Slovak Republic, *michal.gregor@fel.uniza.sk*

Igor Miklóšik
University of Žilina, Department of Control and Information Systems,
Univerzitná 8215/1,010 26 Žilina, Slovak Republic

s. 49-62

Juraj Spalek
University of Žilina, Department of Control and Information Systems,
Univerzitná 8215/1,010 26 Žilina, Slovak Republic

LEARNING A FUZZY MODEL FOR EVACUATION SPEED ESTIMATION USING FUZZY DECISION TREES AND EVOLUTIONARY METHODS

ABSTRACT
The paper deals with evacuation time estimation using a fuzzy meta-model inferred from data using a combination of fuzzy decision trees (to construct the rule base) and evolutionary tuning (to optimize the membership functions). It uses real data – collected from existing literature. The paper first covers some basic facts concerning evacuation in road tunnels and its simulation. It then proceeds to discuss evolutionary tuning of fuzzy systems and fuzzy decision trees and describes the proposed approach. It shows that this approach improves upon previous results achieved using a purely evolutionary approach.

KEYWORDS
evo-fuzzy systems, fuzzy logic, fuzzy decision trees, evacuation, tunnel simulation

INTRODUCTION

The area of machine learning currently offers a multitude of continually improving tools for automatic construction and tuning of models, some of which are in widespread use across many application areas. These may include the (currently very popular) artificial neural networks, support vector machines, decisions trees, ensembles of models – such as random forests, gradient tree boosting – and many other approaches. There is one common issue with most of these generic methods as regards their interpretability. All of the models mentioned, perhaps with the exception of very small decision trees, are difficult to interpret. While this is not a major problem in many areas, it becomes a significant issue if they are to be applied in others.

INTERPRETABILITY IS REQUIRED

The application we study in the present paper is that of evacuation speed estimation for road tunnels. Now in this particular area interpretability is strongly called for. Indeed, if safety-related decisions are to be made using the model, it is imperative that its structure be clear, its inner workings verifiable and its predictions well understood. Also, one may want to use the model to extract human-understandable knowledge about the ways in which various factors influence the evacuation times. These may in turn be used to derive more robust, effective and cost-efficient guidelines for selecting tunnel equipment and for other related tasks. For all of these reasons, interpretability is the key property that we seek in the model.
This is the main reason we have selected the fuzzy inference system as the model of choice. Fuzzy inference systems have long ago distinguished themselves by their ability to express interpretable models of complex systems. Their key characteristics include the ability to utilize vague expert knowledge, to formalize linguistic statements and to act in the role of controllers as well as models.

THE ABILITY TO LEARN FROM DATA IS REQUIRED

An admitted weakness of most fuzzy systems is their lack of an in-built learning mechanism. Fuzzy inference systems have conventionally been designed by experts, who would have set-up the linguistic rules and the corresponding membership functions by hand. Such fuzzy inference systems (FIS) are by definition interpretable, but they are also necessarily suboptimal. While experts may be reasonably good at determining linguistic rules (if they are fairly well-acquainted with fuzzy logic), they almost invariably perform very poorly at determining numeric parameters (in this case the precise shapes of the membership functions).
For these reasons, a number of research teams have explored ways of constructing fuzzy inference systems automatically. Perhaps the most notable among these are hybrid systems such as the adaptive neuro-fuzzy inference system (ANFIS) [1] and evolutionary approaches as exemplified by the various types of genetic-fuzzy systems [2–4].
In a previous paper [5] we have shown that it is possible to significantly improve upon a baseline hand-designed fuzzy inference system for evacuation speed estimation using evolutionary tuning. We have experimented with several evolutionary methods to this end and we have designed a particular decoder, which ensures that any combination of evolutionarily-selected parameters, results in a valid FIS. One important limitation of those results was that our method was not able to automatically co-evolve the rule

base of the FIS as well. While we have experimented with several ways of encoding the rules (more details in [5]), the results were underwhelming – possibly due to an excessive enlargement of the searched space.

EVOLUTIONARY OPTIMIZATION AND FUZZY DECISION TREES

Motivated by these considerations, we now seek to supplement the evolutionary approach, which has shown itself useful in tuning the membership functions, with a separate mechanism for inducing the rules. One family of methods, which has in the past shown itself particularly efficient in constructing fuzzy rules from data, is that concerned with fuzzy decision trees. Much like standard, crisp decision trees, which can be transformed into crisp rule bases, a fuzzy decision tree can easily be converted into a fuzzy rule base.
Thus, the approach proposed in the present paper is to tune the numeric parameters of the fuzzy model using particle swarm optimization (PSO) and to construct the rule base in the form of a fuzzy decision tree using a variant of the fuzzy ID3 algorithm. Both of these ideas as well as the way in which we propose to combine them into a single system will be described in detail hereinafter, along with experimental results.

1. EVACUATION IN ROAD TUNNELS

Road tunnels are nowadays an important part of traffic infrastructure since they shorten the paths in mountainous regions. Shorter travel times lead to higher economical effectiveness. On the other hand, evacuation of people from the tunnel during unexpected event such as fire is a complicated procedure. Tunnel evacuation paths are usually straightforward, but fire has to be detected by the control system of the tunnel first, then people in the tunnel have to be informed and persuaded to start the evacuation. Finally safe conditions during the evacuation process have to be provided for the longest available time. Heterogeneous technological equipment is installed in the tunnel to provide all mentioned tasks. Impact of each technological subsystem on the safety can be estimated more likely by simulations because the tunnel cannot be closed at any time during its continuous operation.

2. THE ROLE OF SIMULATION

One of the problems in the area of evacuation speed estimation is the considerable difficulty connected with acquiring real data. To carry out any very extensive experiments is hardly practicable due to the excessively large costs this would incur. Any real experiments also have to abstract from many details, because a perfectly realistic reconstruction would endanger human persons in a degree similar to a real emergency. Thus, experiments that have actually been carried out to date are of very limited scale. Furthermore, behaviour of people under critical conditions is largely unpredictable and cannot be precisely estimated using simple evacuation training. This is one of the reasons why a crucial role in this area is played by simulation. It provides a cost-effective way to experiment with a number of different parameter configurations in a relatively short amount of time and given a faithful model, it can help to estimate the evacuation time more accurately.

3. THE TUNNEL SIMULATOR TUSIM

Tunnel simulator (TuSim) is a PLC-based system which can simulate all devices of the tunnel technological equipment by the software inside the PLC. Equipment of several tunnels, control of the traffic sequences and tunnel reflexes are also implemented. Tunnel reflex is a reaction of the control system to an unexpected event in the tunnel like: complete or partial power failure, fire, traffic alarm or pre-alarm, lighting malfunction, SOS button, physical measurements alarm or pre-alarm [6]. There are many graphical screens to visualize the state of each subsystem of the technological equipment – at least one for each subsystem. The simulation of evacuation process can be seen on the part of the traffic control screen on Fig. 1. Numbers on the figure indicates the count of the persons in each zone. The evacuation model is directly interconnected with the traffic model and simulation of technological equipment. Process of the real-time simulation experiment is complex from placement of the vehicles, detection of unexpected event, tunnel reflex, closing of the tunnel and evacuation process. That is the significant advantage over other available evacuation simulation tools described in the next chapter.

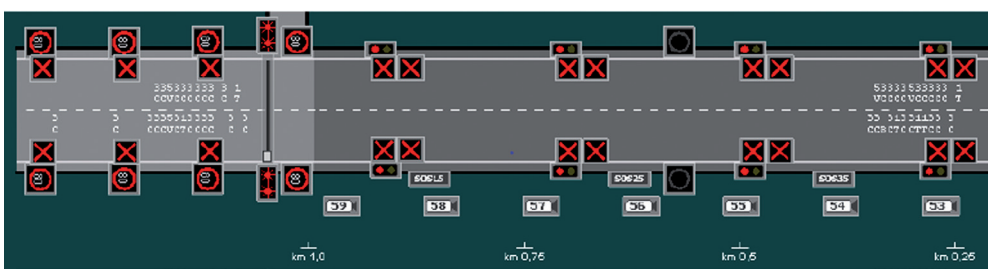


Fig. 1. Simulation of evacuation in road tunnel

4. EVACUATION SIMULATION

There are several software tools available to simulate the evacuation process under fire conditions and smoke toxicity such as FDS+EVACS [7], GridFlow [8], BuildingExodus [9]. Smoke influences the movement of people in several ways such as lower visibility, way-finding, walking speed and physiological aspects.
Three different data sets are available concerning the walking speed and behaviour of people in smoke. The first one by Jin is rather old [10], the second one is based on more recent work by Franzitsch and Nilsson [11], the third one by Fridolf et.al [12] is the most recent. We have decided to analyse only the first two because we wanted to compare the fuzzy decision trees with the model described in our previous article [5]. Jin analysed the movement in two types of smoke: irritant and non-irritant. As can be seen in Fig. 3, speed decreased even if visibility in smoke (extinction coefficient) was below one. Frantzich and Nilsson also analysed the same problem with different smoke types, irritation simulation, population structure, different corridor length and complex structure. Therefore, these data sets cannot be easily exchanged and interpretation of the results has to be considered carefully. Ronchi et al. [13] applied both data sets and their interpretations to simulate the impact of smoke with several simulation software tools. General relationship between walking speed and density of persons in clear conditions can be seen on Fig. 2.

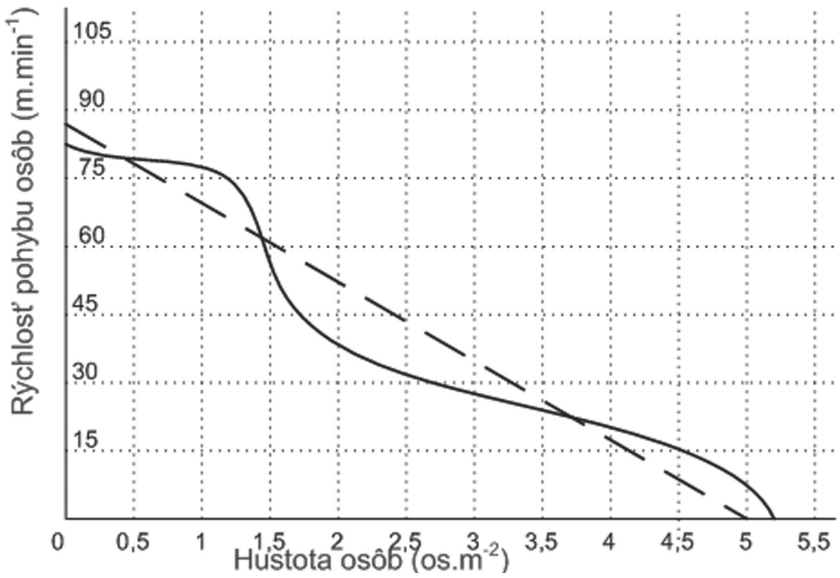


Fig. 2. Walking speed vs. density [14].

Fig. 3 shows results of both mentioned experiments. It's easier to follow and approximate the trend of decreasing the speed in Jin's experiment, results of the newer research from Frantzich and Nilsson are preferred for our simulations. A wider range of the extinction coefficient was analysed than in Jin's experiment and also the results of the movement with and without illumination were included. As can be seen illumination has markedly slowed down the decrease of the walking speed. This information should be included in our TuSim simulator evacuation model, since also the influence of technological equipment during an unexpected event in the tunnel should be analysed.

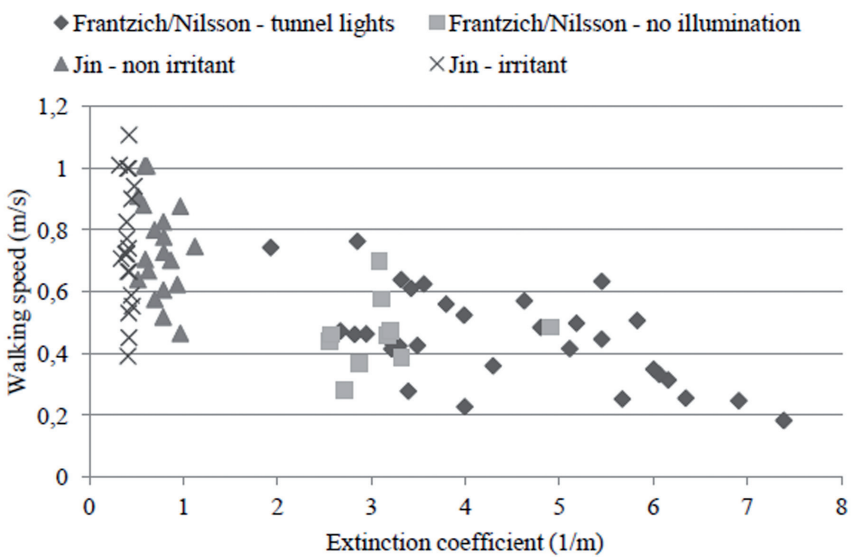


Fig. 3. Walking speed vs. extinction coefficient [13].

5. TUSIM EVACUATION MODEL

With respect to the measured data from the experiments we should include the following factors as inputs to our model of walking speed during evacuation in the tunnel: density of persons on the sidewalks (pers/m2), extinction coefficient (1/m) and operational tunnel lighting level (%).
Operational lighting should, in the normal case, be turned to 100% when the control system detects an unexpected event in the tunnel. TuSim simulator can adjust the lighting level according to meteorological conditions on the tunnel portals, so the fuzzy input should not be limited to just turning the lights on and off – e.g. lighting failure. Implementation of the FIS into the TuSim was realized by creating the interface from the simulator to the existing fuzzy logic library. The solution was faster to implement than realization of the complete fuzzy framework and was limited only by the importing the existing DLL functions into the scripting language [15] of the TuSim visualization.

6. GENETIC-FUZZY SYSTEMS

In the past a considerable research effort has been invested into the development of hybrid genetic-fuzzy systems. These use genetic algorithms to automatically synthesize fuzzy models or controllers using data. The data is usually in the form of pairs, where is an input and is its corresponding desired output. We can then compute the error that the model makes on a particular dataset using some measure of the difference*:* , where is the actual output of the fuzzy model with parameters .
When applying genetic-fuzzy systems, we distinguish between two tasks [4]:
- *Genetic tuning* – which, given an existing rule base, only optimizes the numerical parameters of membership functions (MFs);
- *Genetic learning* – which also learns the rule base (RB), i.e. it is able to create a fuzzy inference system from scratch.

6.1. TUNING THE MEMBERSHIP FUNCTIONS

In the case of genetic tuning, the representation of solutions is not over-complicated. The solution consists of genes that express:
- The type of the membership function (most often a triangle, a trapezoid of a Gaussian function [3]);
- Several numeric parameters that determine the actual shape of the membership function (e.g. the position of vertices – usually 2 to 4 parameters).

7. FUZZY DECISION TREES

Fuzzy decision trees (FDT) represent a generalization of traditional, crisp decision trees. The difference is that in fuzzy decision trees, branching points do not split the original dataset into crisp sets, but rather into fuzzy sets. In other words, the attributes, which characterize the data samples, do not correspond to standard categorical variables, but can instead be considered as linguistic variables. Thus, for an FDT we might have a dataset with numeric values – from a certain universe of discourse, over which the MFs of the linguistic values will be constructed. One of the consequences of such setup is that several leaves may correspond to a sample with a non-zero degree of membership.

7.1. THE FUZZY ID3 ALGORITHM

When creating a decision tree – whether crisp or fuzzy – the main problem in determining its structure is which attribute to pick for any given branching. In the crisp version of the ID3 algorithm, one selects attributes, which offer the greatest information gain, i.e. which have the smallest classification entropy.

The notion of classification entropy can be generalized for the fuzzy setting – for the most part, we only have to replace the classical notion of set cardinality with the cardinality of a fuzzy set, when computing relative frequencies. This yields a version of the fuzzy ID3 algorithm.

Cardinality of a Fuzzy Set The cardinality of a fuzzy set is a sum of the degrees of membership for all its members (although there are also other competing definitions) [16]: where *U* is the universe of discourse and is the membership function of a fuzzy set *A.*

Notation For the most part, we will follow the notation of [17]. Let there be *n* attributes to select from at a particular branching point – at a non-leaf node *S.* Let each attribute have different linguistic terms. Also, let denote the classification attribute (the output) with values .

Relative Frequency of a Term w.r.t. the Class The relative frequency of a linguistic term with respect to the *j*-th fuzzy class (output term) at a non-leaf node *S* is defined as [17]: , i.e. how many times and co-occur in a fuzzy sense in *S* over the number of times itself occurs in *S.*

Fuzzy Classification Entropy of a Term Given these concepts, fuzzy classification entropy of a linguistic term can be defined as [17]: .

Fuzzy Classification Entropy of an Attribute The classification entropy of attribute is then merely a weighted sum of the entropies of its individual attributes, i.e. [17]: , where the weight is determined as: . That is to say, is the relative frequency of linguistic term among all the terms of attribute .

Finally then, when constructing a fuzzy decision tree using the fuzzy ID3 algorithm, the procedure is to greedily select the attribute with the minimum fuzzy classification entropy. It is also necessary to be able to decide when to branch vs. when to add a leaf node. We will discuss the way in which we address this in the next section – specifically with regard to the proposed approach.

When forming a leaf node *L* we label the leaf node with the output term, which has the greatest cardinality in it.

8. THE PROPOSED APPROACH

As mentioned in the introduction, the proposed approach combines the evolutionary tuning of membership functions (MFs) with a fuzzy decision tree technique for inducing the rule base. We will start this section by discussing the evolutionary approach and then proceed to the specifics of the fuzzy decision tree technique. Finally we will show how both of these fit together to construct the entire fuzzy inference system.

What we need to note before we proceed to any details is that in each case the optimization process starts from a simple hand-designed template FIS. This FIS determines how many input/output variables there are, how many MFs they should comprise and what their analytic forms should be. The template FIS need not have rules (they would get replaced in the course of optimization in any case) and the actual numerical parameters of the MFs may simply be zero-initialized.

8.1. TUNING THE MFS USING PARTICLE SWARM OPTIMIZATION

In our previous paper [5] we have experimented with several evolutionary methods – namely with genetic algorithms, differential evolution and particle swarm optimization (PSO). In the present paper we will limit our attention to the PSO approach, which seemed to achieve the most robust results (although the other two approaches were not so far behind and we did not carry out any exhaustive analysis of their various possible parameter configurations). We use a Python implementation of PSO as provided in the inspyred library. Unless explicitly stated otherwise, the parameter settings for PSO will be as follows: the inertia, and both the acceleration coefficients will be set to 0.5.
The optimization criterion is to minimize the mean squared error (between desired and actual outputs of the FIS, given the training data). There is also a penalization term, which we will discuss later.
As mentioned hereinbefore, we have experimented with co-evolving the rules along with the MFs (i.e. the MFs and the rules were both part of the evolved individual), but the results were not encouraging – that is the main reason for instead applying fuzzy decision trees in the present work. We will nevertheless include the version of the experiment with co-evolution for the purpose of comparison.
We will not consider the representation in any detail, since it has already been described in [5]. Suffice it to say that we start with a template FIS and we concatenate its numeric parameters to form the genotype. A decoder is used when transforming the genotype into a phenotype in order to make sure that every possible genotype results in a valid FIS. If a FIS still yields a NaN (not-a-number) output, we penalize it with a flat penalty of 5.

8.2. CO-EVOLVING THE RULES

In addition to the version of the method, which uses fuzzy decision trees to generate the rule base, we also include a version, which co-evolves the rules together with the MFs using PSO. This approach does not perform very well, but we include it nevertheless for the sake of comparison.
We presume that there is some maximum number of rules – in our case predefined to 100. The rules have fixed structure – for each input/output variable we encode which linguistic value it takes. There is a special value, at which we drop the variable from the rule altogether (i.e. we encode that its value does not matter). For every rule, there is also a single binary parameter, which determines whether the rule is active. Only active rules are included in the final rule base. For regularization purposes we penalize the solution for every active rule by the penalty of 10. This encourages PSO to find solutions with compact rule bases.

8.3. LEARNING THE RULES USING A FUZZY DECISION TREE

The main approach to constructing the rule base employed in this paper is based on fuzzy decision trees. Thus, given the input/output variables and their membership functions, we employ a version of the fuzzy ID3 algorithm described above to generate a fuzzy decision tree. Once the fuzzy decision tree has been generated, it is easy to transform it into a standard fuzzy rule base by traversing all paths from the root to the leaves.

We use a simple set of rules to determine whether a node should be a non-terminal node, or a leaf node – a leaf node is created if any of the following criteria are met:
If the maximum depth has been reached.
If repetition of attributes along a path is not allowed and we have already used all the attributes.
If the total cardinality of samples at the present node is lower than a predefined threshold (i.e. there are not enough samples, and we might just be observing noise).
In our present experiments, we do not allow for repetition of attributes along a path. While repeatedly applying the associated norms makes sense from a purely numerical standpoint and may indeed improve accuracy, repeating attributes also considerably impairs interpretability. Unless mentioned otherwise, the maximum depth of the FDT will be set to 20 in the following experiments, and the minimum cardinality to branch will be set to 5.
When computing relative frequencies (as well as when doing inference), some paths will have very low degrees of membership. In consequence, we can ignore their contributions without any significant loss of accuracy, thus incurring non-negligible performance improvements.

8.4. USING PSO AND FDTS TOGETHER

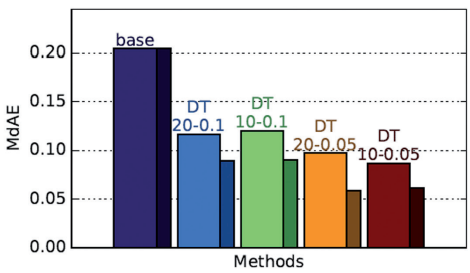
To combine the evolutionary tuning of membership functions with the generation of fuzzy rule base using a fuzzy decision tree is a relatively straight-forward task. We know that in order to construct the FDT using a variant of the fuzzy ID3 algorithm, we already need to have the MFs. Conversely, when we create candidate MFs using PSO, we need rules in order to evaluate them. Thus, there is a very obvious way to combine PSO and FDTs: We first use PSO to create candidate membership functions. We then use these MFs to construct a rule base using a fuzzy decision tree. Finally, we use the resulting FIS to evaluate the MF candidates and proceed with PSO.
The downside of wrapping the FDT within the PSO process in this manner is that an FDT has to be constructed every time a candidate solution is to be evaluated. While this is very simple conceptually, it may be quite computationally intensive for large datasets.

9. THE EXPERIMENTS

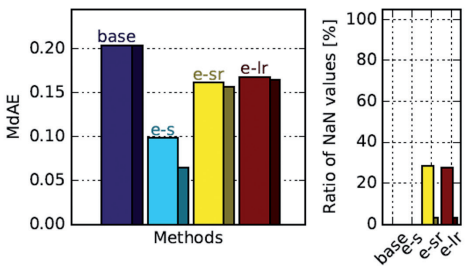
In this section we will present some empirical results. These are all computed using 10-fold cross-validation. Also, in each case 10 trials were made and their results averaged. The results are reported in terms of Median Absolute Error (MdAE). In each case we report both the results on training data (in-sample results) and testing data (out-sample results). Where relevant, we also report the portion of runs in which the FIS yielded NaN outputs.
Some of the default parameters have been described in the preceding sections, thus we can now focus our attention on the ones that will be changing. Let us start with the baseline approaches. The first baseline against which we compare our approach is a hand-designed fuzzy inference system for the problem (shown as “base” in the figures). As a second baseline, we chose crisp decision trees for regression – to this end we have employed the implementation from the scikit-learn Python package.
The results follow in Fig. 4. In the following sections, we will discuss them in some detail. The remaining portion of the paper will then consider their implications and present the conclusions we can draw from them.

9.1. CRISP DECISION TREES

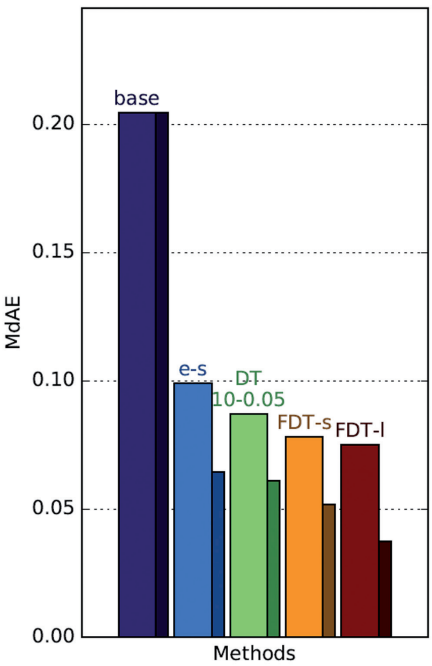
Fig.4a shows a comparison of the results achieved using crisp decision trees under various configurations. The configurations are denoted as DT-n-p, where DT stands for crisp decision trees, n stands for the maximum number of leaf nodes and p stands for the minimum ratio of samples for a leaf (i.e. the number of samples required to form a leaf node over the total number of samples available). As we can see, the decision trees (unsurprisingly) do much better than the hand-designed baseline. Their results vary a little depending on the regularization parameters.



(a) Decision trees in several configurations. DT-n-p means crisp decision trees with the max. of n leaves and p is the min. ratio of samples for a leaf.



(b) PSO-based results: e-s is base with MF tuning, e-sr also co-evolves the rules, e-lr co-evolves too, but for a larger template FIS.



(c) A comparison showing the results of the baseline vs. the PSO vs. crisp decision trees vs. the proposed approach. The FDTs have max. depth of 20 and min. cardinality to branch of 5.

Fig.4: Comparison of several methods in terms of MdAE (smaller is better). The wider bars correspond to out-sample results and the narrow bars to in-sample results. Where appropriate, percentage of NaN outputs is also shown

In the present case, a tree with 10 leaves seems to be quite sufficient – it generalizes better than a tree with 20 leaves. Also, 0.05 seems to be a better fit for p than 0.1. This is because with 0.1 creation of a leaf requires that at least 10% of the data fall under it. This imposes a relatively severe limit on the depth of the resulting decision tree. In any case, all the results are either slightly above 0.10 MdAE or slightly below it.

9.2. PSO WITH AND WITHOUT RULE CO-EVOLUTION

Further results, shown in Fig.4b, represent a comparison between several PSO-based configurations. The e-s configuration corresponds to taking the baseline FIS (with 3 trapezoid MFs for each I/O variable) as a template and tuning its MFs using PSO. The e-sr configuration also uses the baseline FIS as a template, but in addition to tuning the MFs, it also replaces the original rule base with one generated using co-evolution. In a similar way, e-lr applies co-evolution to a larger template FIS (with 7 trapezoid MFs for each I/O variable), for which there are no hand-designed rules.
As shown in the figure, although in the e-sr configuration, PSO is given strictly more control over the resulting solution, their performance deteriorates noticeably. This may be due to the fact that including the rules in the representation excessively enlarges the searched space. Simple attempts to fix this, such as increasing the size of the population or the maximum number of generations do not help significantly.
The performance deteriorates further when using the larger FIS in the e-lr case. Here too, the deterioration can probably be ascribed to a further enlargement of the searched space. The difference in performance is much less noticeable than that between e-s and e-sr. This may again be because the search space is enlarged to a much lesser degree by adding 16 MFs than by adding a large number of fixed-structure rules.

9.3. PSO AND FUZZY DECISION TREES

Now for the proposed approach in its full form: the results are presented in Fig. 4c, and compared against the best performing configurations from the previous sections. The configurations are denoted as FDT-x-m-r. Symbol x denotes which template FIS was used: s means that the baseline FIS (without the rules) was used as the template and l denotes the larger template FIS with 7 MFs for each I/O variable. Symbols m and r denote the maximum depth and the minimum cardinality to branch respectively.
We have also tried varying the maximum depth and the minimum cardinality to branch, but found this to have minimum effect. Some of the results are presented in Fig. 5. It is clear that the results are better with the larger template FIS – and that class of results is also more robust to the other parameters.

10. DISCUSSION OF THE RESULTS

The results clearly indicate that using automatic tuning and learning methods it is possible to improve significantly upon the baseline hand-designed fuzzy inference system. Mere PSO-based tuning of membership function parameters is able to halve the median of the absolute error. These results can further be improved by automatically constructing the rules as well. An added advantage of this is that it also allows us to use template fuzzy inference systems, for which hand-designed rules are not available – i.e. we can build the FIS from scratch.
The method seems to generalize slightly better than the implementation of crisp decision trees we were working with. However, it is dubious whether the improvement is large enough to merit the vast amount of extra computational power required to arrive at such solution. It therefore seems necessary to search for different methods of determining the membership functions in the future, e.g. by generalizing the procedures employed by decision tree regressors.

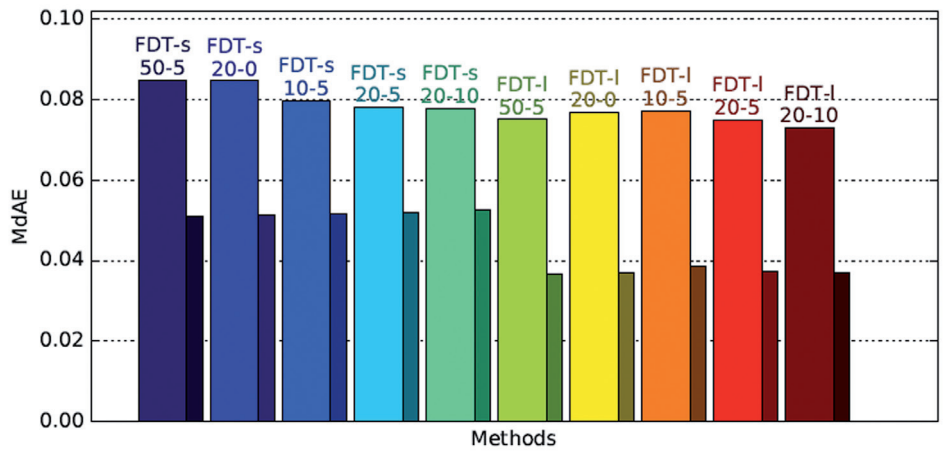


Fig. 5: A comparison of the PSO-FDT approach under several configurations. The notation is FDT-x-m-r: x is either s for the baseline FIS template or l for the larger template; m is max. depth and r is min. cardinality to branch.

10.1. META-MODELLING TUNNELS

One the main takeaways from this paper is that although it is possible to fit a fuzzy model to the empirical data used here, the data itself is extremely sparse and contains very few instances. Indeed, with so few samples, it is not easy to leave a portion of the data aside as a validation set for hyperparameter tuning. The data is also extremely noisy. The provided features do not seem to have sufficient explanatory power. It is probable that some factors that have a large influence on the outcome were not included.
However, it is notoriously difficult to acquire real data for this kind of application and to carry out any extensive experiments is hardly practicable due to the large costs this would incur. There is also the inevitable necessity to abstract from many details of any actual scenario, because a perfectly realistic reconstruction would endanger human persons in a degree similar to a real emergency.
Most experiments that were in fact carried out in the past were done on a relatively small scale and under many simplifying assumptions, such as using cold and non-irritant smoke instead of observing the effects of a real combustion.
Given the existing simulation tools for the problem, the most reasonable approach for getting richer data seems to be to create a powerful and comprehensive model and to acquire a large amount of data through simulation. The real data, despite its being noisy and sparse, could be used to provide a degree of verification for such synthetic data. A simpler, computationally less expensive meta-model can then be made to approximate this synthetic data.
The idea is that fine-grained, highly realistic simulation – taking into account details such as the physics of airflow and other natural phenomena – can be used to provide rich data for subsequent approximation. This data can be computed at leisure, in an offline manner. The estimates themselves can then be provided rapidly, using a meta-model and at a fraction of the computational costs associated with full simulation.

CONCLUSION

We have proposed a hybrid approach combining particle swarm optimization (PSO) and fuzzy decision trees (FDT), to automatic creation of a fuzzy model for evacuation speed estimation. The results show that the approach is viable and that it can significantly improve upon hand-designed fuzzy inference systems as well as upon other methods such as evolutionary tuning of the membership functions without rule learning and co-evolution of rules together with the membership functions.
However, the approach is also computationally expensive and it may be advisable to search for cheaper alternatives – e.g. by replacing PSO with a simpler and greedier way of determining the membership functions.
One the main takeaways from this paper is that the available empirical data for evacuation speed estimation is very small, sparse and noisy. Realistic experiments are very expensive in this domain and they necessarily abstract from many details, because a perfectly realistic reconstruction would endanger human persons in a degree similar to a real emergency.
These conclusions highlight the crucial need for powerful, high-fidelity simulation tools, allowing for the creation of highly-detailed and physically faithful models. Such simulation tools enable us to generate richer data – although the models themselves need to be verified first. Such simulation still requires a lot of time and computational resources. Our next goal, therefore, is to produce a larger amount of data from various configurations of a model and then use this data to create a simpler, faster meta-model for evacuation speed estimation, e.g. using the approach present in this paper, or a similar approach.

This contribution/publication is the result of the project implementation: **Centre of excellence for systems and services of intelligent transport,** ITMS 26220120050 supported by the Research \& Development Operational Programme funded by the ERDF.

We support research activities in Slovakia / Project is co-financed from EU funds.

REFERENCES

[1] Jang, J.S.: Anfis: adaptive-network-based fuzzy inference system. IEEE transactions on systems, man, and cybernetics 23(3) (1993) 665-685.
[2] Tunstel, E., Jamshidi, M.: On genetic programming of fuzzy rule-based systems for intelligent control. Intelligent Automation & Soft Computing 2(3) (1996) 271-284.
[3] Cordón, O., Herrera, F., Gomide, F., Hoffmann, F., Magdalena, L.: Ten years of genetic fuzzy systems: current framework and new trends. In: IFSA World Congress and 20th NAFIPS International Conference, 2001. Joint 9th. Volume 3., IEEE (2001) 1241-1246.
[4] Herrera, F.: Genetic fuzzy systems: taxonomy, current research trends and prospects. Evolutionary Intelligence 1(1) (2008) 27-46.
[5] Gregor, M., Miklóšik, I., Spalek, J.: Automatic tuning of a fuzzy meta-model for evacuation speed estimation. In: 2016 Cybernetics & Informatics (K&I), IEEE (2016) 1-6.
[6] Kopásek, J.: SW for simulation of functionality of road tunnel technology equipment. ELTODO Ltd. (2013) (in Slovak).

[7] Korhonen, T., Hostikka, S.: Fire Dynamics Simulator with Evacuation: FDS+EVACS Technical reference and User's Guide. VTT Technical Research Centre of Finland (2010).
[8] Bensilum, M., Purser, D.: GridFlow: an object-oriented building evacuation model combining pre-movement and movement behaviors for performance-based design. Fire Safety Science – Proceedings of the Seventh International Symposium (2003).
[9] Galea, E.R.: A General Approach to Validating Evacuation Models with an Application to EXODUS. Journal of Fire Sciences. 01/1998 (1998).
[10] Jin, T.: Visibility and Human Behavior in Fire Smoke. SFPE Handbook of Fire Protection Engineering Third Edition (2002).
[11] Frantzich, H., Nilsson, D.: Evacuation experiments in a smoke filled tunnel. Interscience communications Limited, London, UK (2004).
[12] Fridolf, K., Andrée, K., Nilsson, D., Frantzich, H.: The impact of smoke on walking speed. Fire and Materials, Volume 38, Issue 7 (2014).
[13] Ronchi, E., Gwynne, S., Purser, D., Colonna, P.: Representation of the Impact of Smoke on Agent Walking Speeds in Evacuation Models. Fire Technology 49 (2013).
[14] Matis, P., Spalek, J.: Model of people evacuation from a road tunnel. Archives of Transport System Telematics 5 (2012) 20-25.
[15] Miklóšik, I., Spalek, J.: Fuzzy logic based calculation of evacuation time for the tunnel simulator, QUAERE 2014, Hradec Králové, Czech Republic (2014).
[16] Wygralak, M.: Cardinalities of fuzzy sets. Springer (2003).
[17] Wang, X.Z., Yeung, D.S., Tsang, E.C.: A comparative study on heuristic algorithms for generating fuzzy decision trees. IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics) 31(2) (2001) 215-226.

UCZENIE SIĘ ROZMYTEGO MODELU DO OSZACOWANIA SZYBKOŚCI EWAKUACJI PRZY UŻYCIU ROZMYTYCH DRZEW DECYZYJNYCH I METOD EWOLUCYJNYCH

STRESZCZENIE

Artykuł zajmuje się oszacowaniem czasu ewakuacji przy użyciu rozmytego meta-modelu wyprowadzonego z danych używając kombinacji rozmytych drzew decyzyjnych (aby konstruować podstawę reguły) oraz ewolucyjnego dostrajania (aby zoptymalizować funkcje przynależności). Wykorzystuje on dane rzeczywiste - zgromadzone z istniejącej literatury. Artykuł omawia najpierw pewne podstawowe fakty dotyczące ewakuacji tuneli drogowych i jej symulacji. Następnie przechodzi do omówienia ewolucyjnego dostrajania systemów rozmytych i rozmytych drzew decyzyjnych oraz opisuje proponowany sposób podejścia. Pokazuje, że to podejście poprawia wcześniejsze wyniki osiągnięte przy użyciu podejścia czysto ewolucyjnego.

SŁOWA KLUCZOWE

systemy Evo rozmyte, rozmyte drzewa decyzyjne, symulacja ewakuacji