Aiding First Incident Responders using a Decision Support System based on Live Drone Feeds.

Jerico Moeyersons[1 [0000−0001−9494−3170]]✉,
Pieter-Jan Maenhaut[1 [0000−0002−9385−8000]], Filip De Turck[1 [0000−0003−4824−1199]]
and Bruno Volckaert[1 [0000−0003−0575−5894]]

Department of Information technology, Ghent University - imec, IDLab
Technologiepark-Zwijnaarde 15 B-9052 Zwijnaarde, Belgium.
Email: Jerico.Moeyersons@UGent.be

Abstract. In case of a dangerous incident, such as a fire, a collision or an earthquake, a lot of contextual data is available for the first incident responders when handling this incident. Based on this data, a commander on scene or dispatchers need to make split-second decisions to get a good overview on the situation and to avoid further injuries or risks. Therefore, we propose a decision support system that can aid incident responders on scene in prioritizing the rescue efforts that need to be addressed. The system collects relevant data from a custom designed drone by detecting objects such as firefighters, fires, victims, fuel tanks, etc. The drone autonomously observes the incident area, and based on the detected information it proposes a prioritized based action list on e.g. urgency or danger to incident responders.
This paper presents the architecture of the framework and a prototype implementation and evaluation of a decision support system, responsible for digesting and prioritizing the large amount of contextual data captured at an incident site. The evaluation of the decision support system shows that the proposed solution works accurately in supporting incident responders in providing a sorted overview of the actions needed in real-time, with an average response time of 334ms on a less powerful device and 263ms on a powerful device equipped with a GPU.

Keywords: Disaster Management · Recommendation Systems · Decision Analysis and Decision Support Systems

1 Introduction

In 2013, a train carrying chemical products derailed in Wetteren, Belgium [23]. Because of this derailment, chemical products contained in the wagons exploded, causing an inferno and the release of toxic substances in the air. The fire department initially used water to extinguish the fire, resulting in a chemical reaction with the released contents and the spread of toxic water through the sewers of a nearby city. This disaster caused one death and several injuries, all of them

intoxicated by the toxic gases rising from the sewer system. If the fire department had known that the wagons contained chemical products, they would have used foam instead of water to douse the flames, resulting in no spreading of toxic water through the sewers and most likely fewer victims.

To prevent such scenarios in the future, we propose a solution in which a drone supports first incident responders. The drone can fly autonomously to the incident area and circles around the incident area multiple times to scan and analyze it by detecting different relevant objects. This results in a more rapid and better overview of the incident as a whole and, when connected to an automated decision support system, also aids in making split second decisions regarding evacuation, on-scene toxic materials (through object recognition), etc. The main goal of the proposed system is to save lives. The automated decision support is responsible for capturing the huge amount of context and historical data available during such an event and analyzing it, in order to provide incident responders on scene with a better overview of the situation, and to offer dangerous situation warnings and an intervention action list ordered according to urgency. The actual decisions however are made by incident responders and the presented system therefore serves as a decision support system.

In this paper, we primarily focus on the design of a novel decision support process, which takes as input different types of object detections provided by airborne drones and other contextual information (e.g. weather information, database information about gas pipelines).Afterwards we evaluate this decision support process to check whether it can provide real-time decisions in case of an incident. The remainder of this paper is organized as follows. In Section II related work is discussed, followed by an overview of the overall architecture in Section III. In Section IV the decision support system is explained in-depth and subsequently evaluated in Section V. Finally, conclusions and avenues for future work are provided in Section VI.

2 Related Work

Several publications focus on how to improve the Emergency Response Management (ERM) with IT solutions. Banjeree et al. [5] for example describe web and mobile applications that are developed for disaster management. They conclude that many applications and frameworks guarantee better critical communication, situational awareness, resource overview and management, and improved decision support. Because every solution focuses on one of previous improvements, the authors propose a unified framework to combine the different solutions. Their disaster management focuses on alerting civilians in case of a greater event. Our solution on the other hand focuses on alerting incident responders based on detailed real-time context-information (stemming a.o. from a custom autonomous drone), and considering the role of the incident responder, e.g. a firefighter may need to be in closer vicinity of a live fire than a paramedic.

Concerning chemical disasters, Dotson et al. [10] propose a framework for detection and decision making. The decision making is based on existing rule

databases in combination with learning from previous incidents. In our paper, we use a deep neural network for detection and decision making instead of existing rule databases. The approach utilized by Dotson et al. for prioritizing detections and decisions could be useful in eventual further work.

Chen et al. [8] discuss different approaches on how ERM can be improved. They propose a non-IT framework focused on better communication and better situational overview. The target of their framework is that other researchers use it to improve ERM, by understanding the overarching requirements for coordination design and implementation. The conclusions from this work summarize different aspects that are needed when building a whole ERM system and has been marked for future work.

The department of Homeland Security built a middleware platform, called Unified Incident Command and Decision Support (UICDS), described by Morentz et al. [18]. UICDS collects data from both governmental and commercial incident management technologies, combines these and creates a role-based, situational awareness information sharing platform. UICDS primarily focuses on alerting civilians in case of specific events to achieve better risk prevention, protection, response and recovery. The proposed deployment methods of UICDS can be used to deploy our decision support system and the UICDS middleware platform gives a good overview of new components that can be added.

Other approaches for supporting incident responders are based on the use of crowd sourcing information, described in Laskey [16] and Abu-Elkheir et al. [4]. The use of pictures, videos and status updates on social media can offer insights to first incident responders during an emergency event. Both solutions propose a way to implement a crowd sourcing platform, but the main bottleneck with this approach is to verify which data is applicable and which data must be discarded, such as outdated information or fake news. Crowd sourcing information can provide an additional, highly interesting source of information for our decision support system, and has been marked for further research.

In Kotsiantis [15] several classification methods based on machine learning are compared and explained. Table 1 gives a simplified overview of the different features of a selection of learning techniques. It is clear that a Neural Network (NN) and a Support Vector Machine (SVM) have better accuracy, but slower training times compared with a Rule Learner (RL) and a Decision Tree (DT). An important difference between SVM methods and the others is that SVM methods are binary, and thus in the case of multi-class problem one must reduce the problem to a set of multiple binary classification problems. Important features for the described decision support system in this paper are accuracy in general, the speed of a decision, the attempts of incremental learning (e.g. the system can learn from different datasets with the intent to get better each time) and the possibility to handle a multi-class problem (a situation needs to be classified into a clear, warning or danger situation). The decision support system proposed in this paper extends these approaches in order to design, prototype and evaluate a decision-making framework supporting incident responders in making split-second decisions in case of an emergency.

It is clear research into decision support systems for incident response is ongoing. On the first hand however some solutions are very incident specific, e.g. for chemical disasters while on the other hand other solutions are focusing on alerting nearby civilians in case of a large incident. Our decision support system focuses on alerting and aiding incident responders on site during a generic event with a potential to plug in incident-specific and civilian alerting solutions.

Table 1: Overview of different learning algorithms. (**** stars represent the best and * star represent the worst performance)

	Decision Trees [17, 19, 24]	Neural Networks [17]	Support Vector Machines [7, 9]	Rule Learners [11–13]
Accuracy in general	**	***	****	**
Speed of learning	***	*	*	**
Speed of decision	****	****	****	****
Tolerance to missing values	***	*	**	**
Tolerance to highly interdependent attributes	**	***	***	**
Danger of overfitting	**	*	**	**
Attempts of incremental learning	**	***	**	*
Solving multi-class problems	****	****	**	****

3 Architecture Overview

Figure 1 provides a general overview of the system. A custom-built drone, visualized in Figure 2, is used, equipped with both a FLIR BOSON thermal camera [1] and two IMX274 4K cameras [2] used for stereoscopic imaging. The drone autonomously detects all relevant objects in the area, such as the emergency workers, the location of the fire, dangerous objects (e.g. barrels containing explosive fluids) etc. The image processing happens on the drone itself, using a Nvidia Jetson TX2 GPU board, and object detection and localization is implemented using the methods presented in our previous work [21, 22].

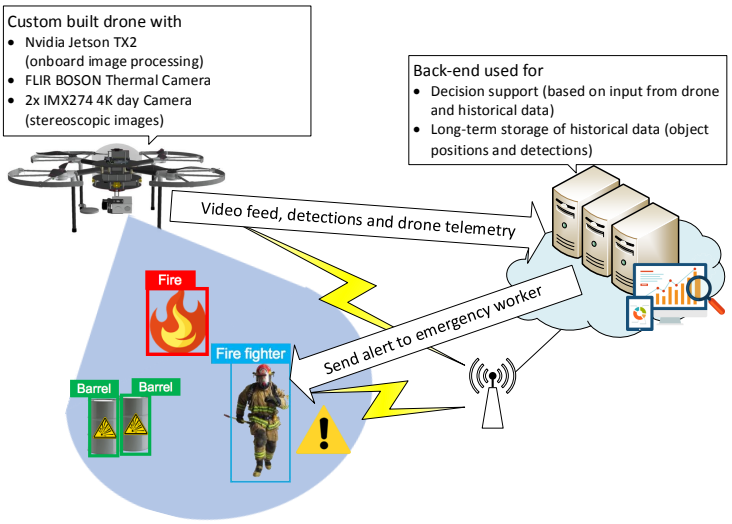

Fig. 1: General overview of the autonomous drone-based decision support system for incident response.

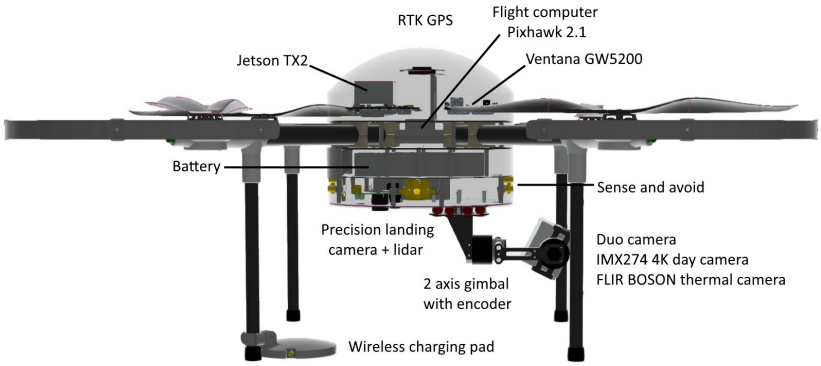

Fig. 2: General overview of the custom build drone.

The contextual information captured by the drone is forwarded to the back-end for further processing, together with the original video feed and additional drone telemetry (location, heading, altitude, battery level, etc.). The back-end analyses this data in real time, using a decision support system. Alongside current feeds, historical data from previous interventions is considered. Furthermore, these data feeds are stored in the back-end for future use (e.g. to learn from mistakes made while dealing with a crisis situation and potentially introduce new rules avoiding such situations in the future). Decision making processing on the drone itself is not recommended as object recognition and communication already demand a lot of processing power / energy on the drone causing an extra stress on the battery which will results in shorter flight time. When the decision support system detects a dangerous situation, e.g. when a fire is located near explosive materials, an alert is generated and forwarded to all emergency workers located in the danger zone.

The back-end also provides a user-friendly dashboard in which all identified objects are visualized on a map, together with other relevant information such as weather information / forecast, gas pipelines and potential alerts or warnings. The location of the drone together with video feeds and overall status such as battery level, altitude, heading, etc. can be optionally visualized on the dashboard. The web-technology based dashboard also shows the flight path of the drone which can be changed by the user during flight. A screenshot of the dashboard with detections, a gas pipeline, drone location and potential decisions is shown in Figure 3. The complete dashboard can be used within a control room setting, or it can be displayed on a tablet used by the supervisor located at the emergency site.

In the remainder of this paper, we will focus on the design, prototype implementation and evaluation of the decision support system.

4 Decision Support - TensorFlow

The decision support system is implemented using TensorFlow [3], an open-source, state-of-the-art machine learning framework created by Google. TensorFlow contains a wide range of functionality, and can therefore be used in a wide area of applications, but is mainly designed for creating a Deep Neural Network (DNN). The reason to opt for a DNN instead of the other proposed learning methods in Section 2 is related to the specific use case in this paper. The use case under investigation requires an accurate and fast approach of a multi-class problem that can be distributed and re-trained. Eventual training speed is less important because training can happen between different interventions on a computer or server with stronger hardware. If we compare these requirements with the features of the different learning methods in Table 1, a DNN was picked as a suitable candidate. The used DNN for the decision support system consists of three layers, consisting of 100, 200 and 100 neurons respectively, whereby the whole network is dense. The used activation function is the Rectified Linear Unit (ReLu) activation function [20], the most successful in terms of results and widely

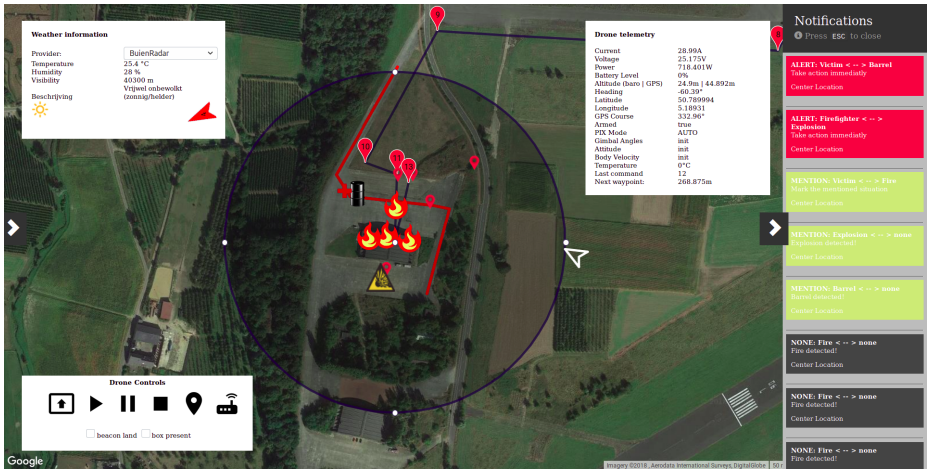

Fig. 3: General overview of the web-enabled dashboard. The top left corner contains the weather overview. The bottom left corner contains the mission control (e.g. start, pause, stop, upload and change current mission. The right top corner contains the drone telemetry (e.g. current, voltage, battery level, altitude, heading). On the right side the prioritized decision list is provided where a red color describes urgent actions (danger situations), a yellow color describes a warning situation and a gray color describes an informative message. The current drone location is marked with a white arrow marker and the blue lines show the flight path of the drone. In the Point-of-Interest circle (part of the drone flight path) the current detections are marked and labeled with a self-explainable marker.

used activation function. It gives an output x when the input (x) is positive and 0 otherwise, resulting in fewer activated neurons in the DNN which means less computational overhead and thus faster results for the incident responders. The used classifier function is the Softmax cross entropy function [6]. It squashes a K-dimensional vector of arbitrary real values to a K-dimensional vector of real values in the range (0, 1) that add up to 1 (the probabilities). This means that the commander and dispatcher can see how certain the decision support system is about its result and eventually can take that into account when making a final decision. A result with a higher probability is more likely to be correct than one with a lower probability.

The input for the DNN are two detected objects and the distance between them, all detected and calculated by the drone discussed in Section 3. Note that other, non-distance related inputs are possible e.g. to generate an alert when a victim is detected by the drone. For easier calculations within the neural network, these input values, and especially the two detections are converted into a label with the Labelencoder from the Python `sklearn` package. Afterwards, these categorical integer labels are transformed with a one-hot, also called an one-of-K scheme. When applied to a scenario with four possible detections, this results in 9 input neurons, namely four for the first detection, four for the second detection and one for the distance between them. Figure 4 provides a general overview of the functioning of the DNN.

The output (prediction) of the DNN is the classification of the input into one of the three possible classes, namely a danger, warning or normal situation. These predictions, together with their probability is shown in an ordered fashion to the user (i.e. the commander at scene or in the dispatch center), to support them in gaining control of the crisis situation. The user can dismiss specific warnings or mark dangerous situations which are not flagged by the system.

After training the DNN, the model can be exported to be used as a *servable* within the TensorFlow Serving API [14]. The TensorFlow serving API allows to serve different machine learning models and manages their lifetimes, so clients can access versioned machine learning models via a high-performance, reference-counted lookup-table. This allows to train multiple different models and request a prediction from a specific model, managed by the TensorFlow server API. In case that there is no reliable communication channel between the back-end and the incident responders on scene, the model can be deployed on a local device in e.g. the intervention truck.

5 Evaluation Results

Now that we have the architecture and the decision support system explained, an evaluation is needed to check whether the system can give decisions accurately and in real-time.

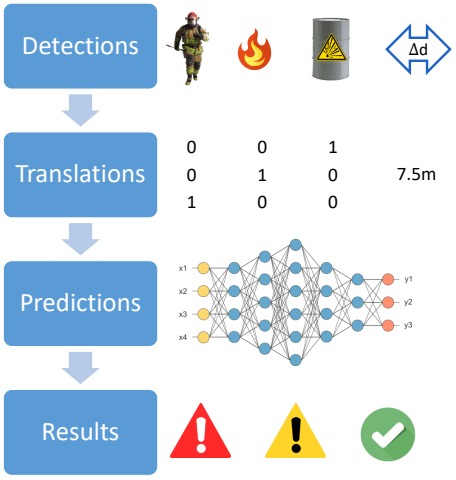

Fig. 4: General functioning of the Deep Neural Network: detections and distance between detections are translated and fed into a neural network which leads to classification of events into dangerous, potentially dangerous or safe occurrences.

5.1 Datasets

The datasets used for the evaluation are built based on previously defined rules, created and generated under the supervision of a fire department expert active in the 3DSafeGuard-VL project. The expert was an ICT responsible for the fire department and cooperated in the design of different incident response protocols. In these rules, two detected objects and a specific distance between them result in different possible outputs, an alert in case of a danger situation, a warning in case of a potentially unsafe situation or nothing when no threats are detected. A sample of select entries in the dataset is provided in Table 2. The datasets are built only for evaluation purposes and the whole system is not limited to the detections specified below. As explained in Section 3 an autonomous incident-scanning drone provides the decision support system with real-time object detection capability.

Three different datasets are created to evaluate the change in accuracy and in response time when more rules are trained. Dataset 1 trains the decision support system on 190 different rules, based on 20 possible detected objects. Dataset 2 trains the decision support system on 435 rules, based on 30 possible detected objects and Dataset 3 trains the system on 990 rules, based on 45 possible detected objects. The total number of rules per dataset is based on the number of possible detected objects, namely $\sum_{i=1}^{x-1} i$ where x is the number of possible detected objects. An overview of the datasets is showed in Table 3.

Accuracy is important to the incident responders because they base their decisions in part on the predictions made by the decision support system. A

lower accuracy means that there is a higher possibility of a wrong prediction. The three datasets have a different amount of rules to study their impact on the accuracy. With more rules incident responders can get more contextual information about the incident because more objects are scanned and analyzed. The response time on the other hand is also of great importance to evaluate the decision support system is capable of providing decisions in real-time or near real-time as a function of the number of rules in the system.

Table 2: Sample of dataset entries

Detection 1	Detection 2	Distance	Result
Firefighter	Explosion	10	Alert
Firefighter	Fire	5	None
Victim	Fire	10	Warning
Victim	Explosion	5	Alert
Fire	Explosion	2	Alert

Table 3: Overview of the different datasets. (# pos. detections shows how many different objects can be identified such as a firefighter, fire, smoke, etc., # training entries shows on how many entries of the dataset training is done and # evaluation items shows on how many entries the accuracy of the model is determined.)

Name	# Rules	# pos. detections	# training items	# evaluation items
Dataset 1	190	20	7600	1520
Dataset 2	435	30	18560	3712
Dataset 3	990	45	40760	8152

5.2 Results

Several experiments were performed using the three different datasets summarized in Table 3. First, the training phase is evaluated with a varied range of training steps, going from 100 to 40000 in steps of 100 in function of the obtained accuracy. This training is done on a Nvidia GTX 980 Ti GPU with CUDA 9.0 and TensorFlow 1.8 installed and the results are visualized in Figure 5. The accuracy of the DNN is at the highest levels starting from 4500 training steps. More training steps do not guarantee better accuracy in general. An important

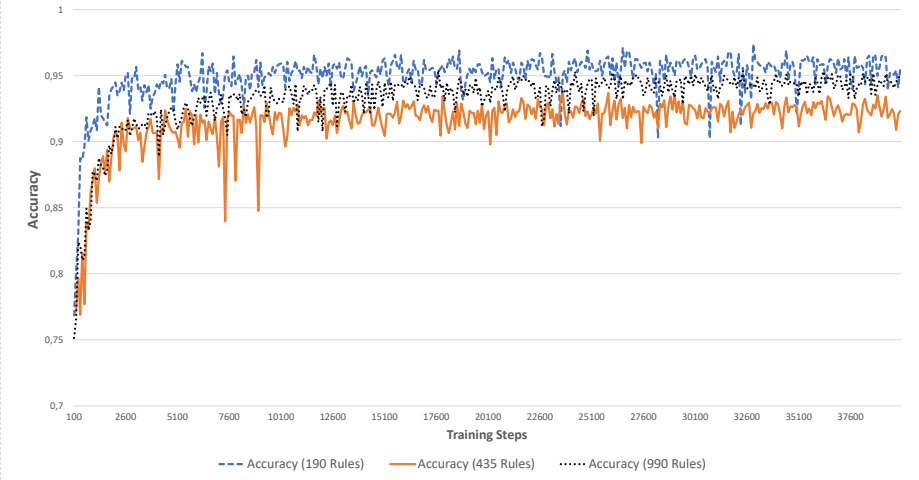

Fig. 5: Accuracy of the trained model as a function of the training steps for the three datasets consisting of 190, 435 and 990 rules.

conclusion from Figure 5 is that a model with less rules has better accuracy than a model with more rules. This behavior can be explained because there are less possibilities within the NN itself, causing a lower fault rate and thus a higher accuracy.

After training three different DNN-models from the datasets, these models are exported to another (less powerful) laptop. This laptop contains 8GB of RAM, an Intel-i5 processor and lacks a GPU. On this laptop, the models are exposed to predict 36 decisions, each based on 2 detections and the distance between them. This will indicate that a model can be exported to a local device in e.g. an intervention truck when communication with the back-end is not reliable or available. When the local device is connected to the drone, it is thus possible to have decision support on site. As shown in Figure 6, the response time is equally for the three different models. The total time the three models, consisting of 190, 435 and 990 rules, needed for making 36 predictions are 421ms, 408ms and 432ms respectively. It is proven that a model works concurrent and can handle multiple request at the same time. A possible explanation for the outliers in response time is that the request was stalled when the model was predicting another request.

Figure 7 shows the response time on 36 predictions for the three models on a device with a GTX 980Ti equipped. Compared to the device that lacks a GPU, the average response time is 23.59% faster for each individual response as for the 36 responses together. It is also remarkable, when using a system with a GPU, that the response time for 190 rules is higher in most cases than the response time for the models with 435 and 990 rules. When the model has more rules, more Nvidia-CUDA cores will be used in parallel, which result in lower response times. This is highly dependent on the specification of the used GPU.

Fig. 6: Required time (in ms) for calculating a prediction on a CPU based on 2 detections and the distance between them, evaluated on the three models consisting of 190, 435 and 990 rules. The request are sent after each other and the total time for predicting these 36 request are 421ms, 408ms and 432ms respectively.

Fig. 7: Required time (in ms) for calculating a prediction on a GPU based on 2 detections and the distance between them, evaluated on the three models consisting of 190, 435 and 990 rules. The request are sent after each other and the total time for predicting these 36 request are 315ms, 307ms and 306ms respectively.

Based on these results, it seems reasonable to train different decision support systems, all tailored to specific incidents such as a model in case of a diving incident, an earthquake, a fire, etc. In case of an incident, the request for decision support needs to specify which type of incident is occurring and based on that, the correct model is selected and more accurate predictions can be done resulting in improved reliability of the system. The incident responders will not notice the change from one model to another, and when switching to a model with more rules, there will be no change in response time. The drawback to this is cases where the decision support needs to aid with mixed incidents such as an earthquake followed by multiple fires and explosions. A possible solution is to introduce general detections and or rules that are available in every model or run multiple models in parallel.

6 Conclusions

This paper proposed the use of a decision support system for the classification of potential dangers based on data retrieved from a drone autonomously surveying an incident zone. Prototyped with TensorFlow, it was shown that it is possible to create a real-time decision support system that helps incident responders on scene to take split second decisions and potentially save lives. Based on the evaluation, it is proven that a decision support model can be deployed on a less powerful device (e.g. a laptop) with a minimal average increase of 100ms in response time. Also based on the evaluation, it is recommended to divide all the possible rules into different incident-specific models in order to gain improved accuracy between 2% and 5%. Incident responders do not need to know which model is used and will not notice when the model is changed in the back-end because the response time does not increase as function of the number of rules in a model. Remark that some common rules need to be included in each model to avoid having to switch between models too often.

As discussed in Section 3, the relearning of the model based on incident responder feedback from previous incidents is a candidate for future research. Also, information obtained through crowd sourcing platforms can provide even more contextual information, not provided by the autonomous drone and should be tied into the decision support system. One of the difficulties with crowd sourcing remains however autonomous selection of relevant data. Finally, a system will be introduced which will allow for more warning handling prioritization on top of the classification between danger, warning or a clear situation.

Acknowledgement

The authors would like to thank all partners namely, Xtendit Solutions, Fire.BE, DroneMatrix, CityMesh, Seris Security, SAIT and KU Leuven - Eavise, in the 3DSafeGuard-VL project and the Agency for Innovation by Science and Technology (Vlaio) for funding and support.

References

1. FLIR Boson, http://www.flir.it/cores/boson/
2. Sony Semiconductor Solutions Corporation - IMX274QC, https://www.sony-semicon.co.jp/products_en/new_pro/october_2016/imx274lqc_e.html
3. Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D.G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., Zheng, X., Brain, G.: TensorFlow: A System for Large-Scale Machine Learning TensorFlow: A system for large-scale machine learning. 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16) pp. 265–284 (2016). https://doi.org/10.1038/nn.3331, https://doi.org/10.1038/nn.3331
4. Abu-Elkheir, M., Hassanein, H.S., Oteafy, S.M.: Enhancing emergency response systems through leveraging crowdsensing and heterogeneous data. 2016 International Wireless Communications and Mobile Computing Conference, IWCMC 2016 pp. 188–193 (2016). https://doi.org/10.1109/IWCMC.2016.7577055, https://doi.org/10.1109/IWCMC.2016.7577055
5. Banerjee, A., Basak, J., Roy, S., Bandyopadhyay, S.: Towards a Collaborative Disaster Management Service Framework using Mobile and Web Applications. International Journal of Information Systems for Crisis Response and Management **8**(1), 65–84 (2016). https://doi.org/10.4018/IJISCRAM.2016010104, https://doi.org/10.4018/IJISCRAM.2016010104
6. Bishop, C.M.: Pattern Recognition and Machine Learning. Springer (2006). https://doi.org/10.1117/1.2819119
7. Burges, C.J.: A tutorial on support vector machines for pattern recognition. Data Mining and Knowledge Discovery **2**(2), 121–167 (Jun 1998). https://doi.org/10.1023/A:1009715923555, https://doi.org/10.1023/A:1009715923555
8. Chen, R., Sharman, R., Rao, H.R., Upadhyaya, S.J.: Coordination in Emergency Response Management. Communications of the ACM **51**(5), 66–73 (2008). https://doi.org/10.1145/1342327.1342340, https://doi.org/10.1145/1342327.1342340
9. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines: And Other Kernel-based Learning Methods. Cambridge University Press, New York, NY, USA (2000)
10. Dotson, G.S., Hudson, N.L., Maier, A.: A decision support framework for characterizing and managing dermal exposures to chemicals during Emergency Management and Operations. Journal of Emergency Management **13**(4), 359–380 (2015). https://doi.org/10.5055/jem.2015.0248, https://doi.org/10.5055/jem.2015.0248
11. Fürnkranz, J.: Pruning algorithms for rule learning. Machine Learning **27**(2), 139–172 (May 1997). https://doi.org/10.1023/A:1007329424533, https://doi.org/10.1023/A:1007329424533
12. Fürnkranz, J.: Separate-and-conquer rule learning. Artif. Intell. Rev. **13**(1), 3–54 (Feb 1999). https://doi.org/10.1023/A:1006524209794, https://doi.org/10.1023/A:1006524209794
13. Fürnkranz, J.: Round robin rule learning. In: Proceedings of the Eighteenth International Conference on Machine Learning. pp. 146–153. ICML '01, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA (2001), http://dl.acm.org/citation.cfm?id=645530.655685

14. Google Research: Tensorflow Serving (2017), tensorflow.github.io/serving/
15. Kotsiantis, S.B.: Supervised Machine Learning: A Review of Classification Techniques. Informatica **31**, 249–268 (2007). https://doi.org/10.1115/1.1559160, http://dl.acm.org/citation.cfm?id=1566770.1566773
16. Laskey, K.B.: CROWDSOURCED DECISION SUPPORT FOR EMERGENCY RESPONDERS **4444**(703)
17. Lim, T.S., Loh, W.Y., Shih, Y.S.: A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. Machine Learning **40**(3), 203–228 (Sep 2000). https://doi.org/10.1023/A:1007608224229, https://doi.org/10.1023/A:1007608224229
18. Morentz, J.W., Doyle, C., Skelly, L., Adam, N.: Unified incident command and decision support (UICDS): A department of homeland security initiative in information sharing. 2009 IEEE Conference on Technologies for Homeland Security, HST 2009 pp. 182–187 (2009). https://doi.org/10.1109/THS.2009.5168032, https://doi.org/10.1109/THS.2009.5168032
19. Murthy, S.K.: Automatic construction of decision trees from data: A multi-disciplinary survey. Data Mining and Knowledge Discovery **2**(4), 345–389 (Dec 1998). https://doi.org/10.1023/A:1009744630224, https://doi.org/10.1023/A:1009744630224
20. Nair, V., Hinton, G.E.: Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th International Conference on International Conference on Machine Learning. pp. 807–814. ICML'10, Omnipress, USA (2010), http://dl.acm.org/citation.cfm?id=3104322.3104425
21. Tijtgat, N., Ranst, W.V., Volckaert, B., Goedemé, T., De Turck, F.: Embedded Real-Time Object Detection for a UAV Warning System. 2017 IEEE International Conference on Computer Vision Workshops (ICCVW) pp. 2110–2118 (2017). https://doi.org/10.1109/ICCVW.2017.247, https://doi.org/10.1109/ICCVW.2017.247
22. Tijtgat, N., Volckaert, B., De Turck, F.: Real-Time Hazard Symbol Detection and Localization Using UAV Imagery. 2017 IEEE 86th Vehicular Technology Conference (VTC-Fall) pp. 1–5 (2017). https://doi.org/10.1109/VTCFall.2017.8288259, https://doi.org/10.1109/VTCFall.2017.8288259
23. Torfs, M.: 1 dead, 33 injured in Wetteren train derailment (may 2013), http://deredactie.be/cm/vrtnieuws.english/News/1.1620582#
24. Yildiz, O.T., Dikmen, O.: Parallel univariate decision trees. Pattern Recogn. Lett. **28**(7), 825–832 (May 2007). https://doi.org/10.1016/j.patrec.2006.11.009, http://dx.doi.org/10.1016/j.patrec.2006.11.009