Machine Learning for Search and Rescue Decision Support

Thomas Laperrière-Robillard, Michael Morin, and Irène Abi-Zeid*

[1]Operations and Decision Systems, Université Laval. Québec, Canada
* irene.abi-zeid@osd.ulaval.ca

Abstract

Maritime search and rescue operations are important humanitarian activities for locating and rescuing survivors in distress at sea. This paper investigates the application of supervised learning techniques to improve search operations planning by estimating the probability of success of a search operation. Four models are evaluated in the study: random forest, K-Nearest Neighbors, Support Vector Machines Regression and Neural Networks. The results show that integrating machine learning can significantly reduce computation time for the allocation of search resources. This can enhance *SAR Optimizer*, the current optimization and evaluation module used in the Canadian Coast Guard decision support system. By improving the quality of search recommendations, our approach has the potential to improve SAR operations and, ultimately, save lives.

Keywords: Machine learning; Simulation; Maritime search and rescue; Canadian Coast Guard; Decision support system.

1. Introduction

Search and rescue (SAR) involves locating and aiding individuals who are in distress or facing immediate danger. Canada is responsible for one of the most challenging and vast SAR zones spanning 18 million square kilometers of land and water. One of the Canadian Coast Guard (CCG) roles is to save and protect lives in the maritime environment. It coordinates an average of 7,000 incidents per year with a rate of 97% of lives saved (Fisheries and Oceans, 2009). Maritime SAR operations in Canada are managed by three joint rescue control centers and two sub-control centers. SAR mission coordinators (SMC) are highly trained individuals who are tasked with the planning, coordination, control and management of operations. One of the biggest challenges for SMCs is deciding where to send search resources. Search planning is time-critical, as survivors must be found quickly due to the rapid decrease of survival rates (Xu et al., 2011).

In order to support SMCs in search planning, the CCG developed the Advanced Search Planning Tool (ASPT) decision support system (DSS), also known as CANSARP (Abi-Zeid, et al., 2019). This DSS includes *SAR Optimizer* (Abi-Zeid, et al., 2019), a search planning module involving simulation and optimization based on search theory (Stone et al., 2016). The output of *SAR Optimizer* is a search plan, namely the assignment of available search and rescue units (SRU) to rectangles, each enclosing a parallel search pattern. In the optimization module, the figure of merit to be maximized is the probability of success (POS), defined as the probability of finding the search object. The DSS evaluates multiple combinations of SRU and search rectangles to propose a best POS search plan in the planning time allowed.

The POS of a search plan is computed by simulating, at each time step, the positions of the SRUs and their proximity to the object's estimated position. However, a simulation approach is quite costly in terms of computation and time, in a context where a search plan must be produced within minutes, which constrains the number of search plans that can be assessed as potential solutions. This motivated us to explore whether and how machine learning (ML) can help make the POS computations faster and increase the number of candidate search plans evaluated. This study examines and compares the performances of four supervised ML algorithms to estimate the probability of success (POS) in search planning (Laperrière-Robillard et al. 2022).

2. Background

After being notified of an incident, the SMC creates, in the ASPT DSS, a SAR case containing all available information about the emergency, the characteristics of the vessel, the number of people involved, the last known point, possible sightings, relevant communications, etc. The next step is to perform a stochastic drift simulation based on Monte Carlo where 5,000 particles, equally likely to be the search object, are seeded using a bivariate Gaussian distribution with a standard deviation specified by the user. The particles are then moved by simulation in time and space according to a drift model that takes into account search object characteristics, surface currents, and winds. The result is a drift model providing the positions of the particles at each time step over a simulation horizon. The simulated trajectories of the particles represent equiprobable trajectories of the search object (Breivik and Allen, 2008). Subsequently, the SMC identifies available SRUs that can be tasked with performing the search operations. Using the drift model and the SRU information, *SAR Optimizer* recommends a search plan. Figure 1 provides a fictitious example of search plans (a parallel pattern and enclosing rectangle) for three SRUs. Since the optimization process terminates once the predefined time limit has been reached, there is no guarantee that all candidate search plans have been simulated and evaluated. The quality of the, possibly sub-optimal, recommended search plans depends not only on the total number of search patterns assessed but also on the sequence in which they were evaluated.

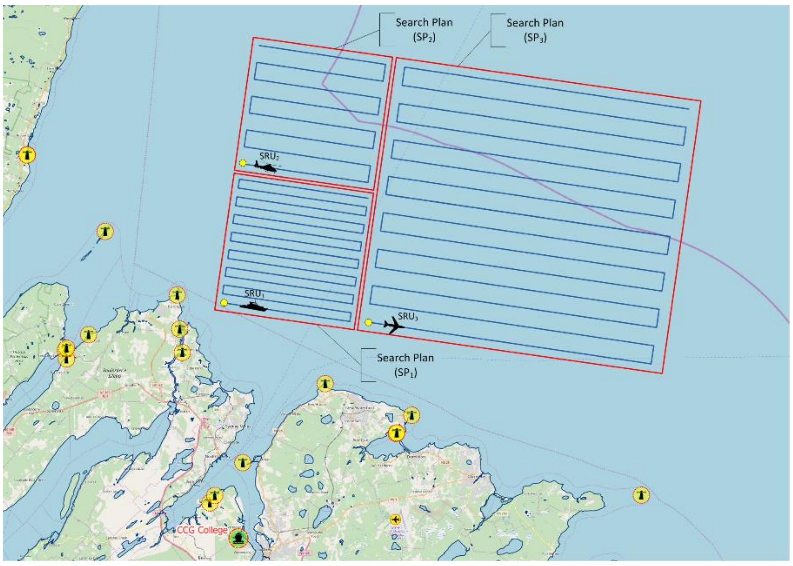

Figure 1: Example of a search plan with three SRUs conducting parallel search patterns in enclosed rectangles

Machine learning is a subfield of artificial intelligence with supervised learning as a particular case where a corpus of labeled learning examples is used to train a prediction model (Russell and Norvig, 2021). In supervised learning, the objective is to be able to predict a dependent variable (or label), here the POS, as a function of independent variables, referred to as learning features, here attributes of search plans. A dataset consisting of POS and attributes is used as learning examples. The learning phase, where a function transforming the attributes into a POS is followed by a prediction phase where this function is applied to obtain the value of a POS corresponding to unseen case attributes. A good model is one that is precise, meaning that it predicts correctly the POS of the training set, and that generalizes adequately, meaning that it predicts correctly the POS of unseen case attributes, evaluated using a test set. In many cases, fine-tuning of the hyperparameters of the ML algorithms is necessary. This consists of randomly picking a number of partitions of the training set *k* and repeating the training with each partition. The performance of the algorithm is obtained from the average performances over *n* sets of *k* partitions. Metrics to evaluate the performance of an algorithm include the mean absolute error (MAE) of the prediction (Kuhn and Johnson, 2013), computed here as the absolute difference between predicted and observed POS values.

3. Methods and Experiments

In order to evaluate an ML approach to predict a POS of a search plan, we followed a four-step process.

First, we generated the learning corpus by using *SAR Optimizer* to simulate and evaluate search plans. The drifting search object was a life raft and the SRU either a helicopter or a fixed-wing aircraft. The available search effort of a SRU is measured by the time spent searching. The experiments were conducted with two different effort levels, namely 3 and 6 hours. We assumed that a single SRU was on-scene searching. This setup resulted in 12 different scenarios (3 drifts × 2 SRUs × 2 effort levels). For each scenario, we generated, simulated, and evaluated the POS of almost 9,100 search plans. Therefore, our final corpus contained 12 sets of 9,100 evaluated search patterns.

We then applied four ML algorithms to each scenario, namely K-Nearest Neighbors (KNN), Support Vector Machines (SVM) Regression, Random Forest (RF), and Neural Networks (NN). We compared their prediction accuracy and the time they took to learn from data. Each model was trained using 70% of the dataset while keeping 30% for testing. This process was repeated 10 times with different partitions. The independent variables used to predict the POS included features of a search plan, namely the bearing, the area and the length and height of the enclosing rectangle, as well as the number and lengths of search and cross legs, and the starting coordinates of the search pattern.

Next, we tested how changing the size of the training set affects prediction accuracy for the ML model retained in the previous step. The goal was to find the best predictions while keeping training size small since generating the learning corpus is time consuming. In fact, it is not readily available from historical data since SAR incidents occur in different locations where the drifts have different characteristics. A smaller training set means faster data collection and training, which is important for real-life search missions. However, using too little data could lead to inaccurate predictions.

Finally, we selected the best ML model based on POS prediction quality, learning time, and training set size. The retained ML model was applied to predict the POS and rank candidate search plans in *SAR Optimizer* by decreasing POS (rectangle ordering heuristic). This allowed *SAR Optimizer* to evaluate, by simulation, the POS starting with the most promising search plans. In order to evaluate the benefits of using the ML model, we then compared the results of *SAR Optimizer* with and without ML based on the POS obtained. Hereafter, SAR Optimizer with the rectangle ordering heuristic is called *SAR Optimizer + ML*, and the standard version of SAR Optimizer is simply called *SAR Optimizer*.

4. Results

4.1. ML Algorithms comparison

The results of comparing the four ML models showed that although three out of four displayed relatively small differences in terms of POS prediction precision, the variability in terms of execution time was considerable. Although the SVM model had the shortest average learning time requiring only 0.37 minutes to train the model, its predictive performance was consistently inferior. The RF, NN and KNN models needed, on average, 51.9, 6.8 and 0.9 minutes respectively. Therefore, we retained the KNN algorithm as a candidate algorithm for predicting POS. Full results are available in (Laperrière-Robillard et al. 2022)

4.2. Prediction quality as a function of training size

Generating the learning set is an expensive operation. The average time needed to produce it for a scenario varies between 28 and 61 minutes when the training set size corresponds to 70% of the full dataset. This is obviously not acceptable since a search plan must be proposed in under 5 minutes. We therefore computed the average MAE for different training set sizes ranging from 455 search rectangles to 7,735. As expected, the prediction's quality improves (MAE decreases) as the size of the training set increases (Figure 2). However, we observed that using a training set of 455 rectangles resulted in an average POS estimation that deviated by approximately 0.012 from the actual ground-truth POS value. Considering that the highest POS values for search patterns range between 0.305 and 0.941, this margin of error is relatively small in practical terms, particularly in scenarios where the best POS exceeds 0.5.

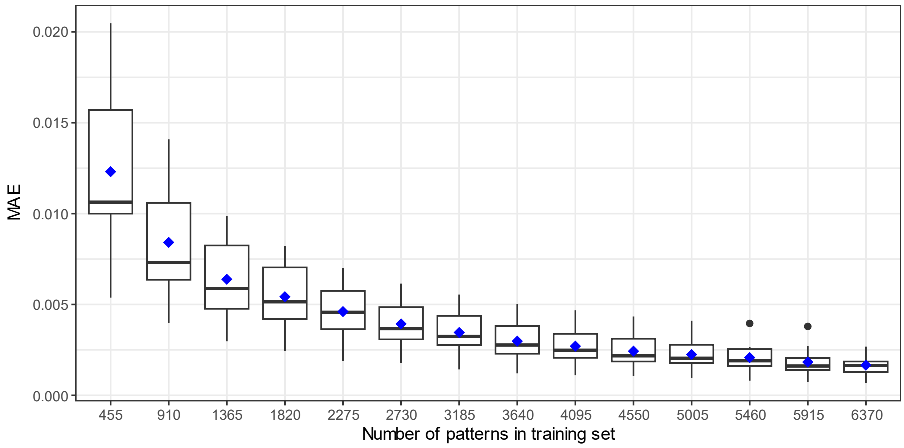

Figure 2: The mean average error of the predicted POS versus the number of plans in the training set **(adapted from Laperrière-Robillard et al., 2022)**

4.3. Comparing *SAR Optimizer* results with and without ML

Based on the previous results, we retained the KNN model with 455 search plans in the training set. We compared the performances of *SAR Optimizer* and *SAR Optimizer + ML* in terms of time needed to attain a best POS value. This comparison was computed over 30 runs representing learning with 30 different datasets of 455 patterns for 12 scenarios where we let *SAR Optimizer* run for 45 minutes. Each scenario is based on a drift (named A, B, or C) and involves a helicopter flying at 500 feet at 90 knots for 3 hours (case 1), a fixed wing flying at 1,000 feet at 120 knots for 6 hours (case 2), a helicopter flying at 90 knots for 3 hours (case 3), or a fixed wing flying at 1,000 feet at 120 knots for 6 hours. Table 1 shows that the total time to obtain the highest POS search plan was much higher without ML than with ML. This is explained by the fact that without ML, *SAR Optimizer* needed to evaluate a larger number of search plans before reaching the best one in the time allocated. Using the ML predicted POS as a heuristic to determine the order in which search plans were to be simulated proved very beneficial.

Table 1. Comparison of *SAR Optimizer* and *SAR Optimizer + ML* **(adapted from Laperrière-Robillard et al., 2022)**

	SAR Optimizer		***SAR Optimizer + ML***	
Scenario	Total time to best POS (min.)	Simulation rank of best plan POS	Average time to best POS (min.) with 95% CI	Median rank of best plan POS
A1	20.657	4,301	2.433 ± 0.149	17
A2	38.211	4,101	4.928 ± 0.378	24
A3	21.139	4,411	2.519 ± 0.146	29
A4	38.153	4,103	5.626 ± 0.485	72
B1	21.909	4,614	3.269 ± 0.517	112
B2	44.387	4,611	6.790 ± 0.985	130
B3	25.028	5,323	4.025 ± 0.812	201
B4	39.028	4,219	6.063 ± 0.991	90
C1	23.510	4,727	5.015 ± 1.279	340
C2	43.506	4,730	7.167 ± 1.135	154
C3	21.285	4,634	2.927 ± 0.355	80
C4	42.947	4,631	6.990 ± 1.362	154

5. Conclusions

In this paper, we set out to explore whether machine learning can improve *SAR Optimizer* in the Canadian maritime SAR planning DSS by developing higher POS search plans in less time. Our experiments showed that the K-Nearest Neighbors algorithm, trained on a relatively small training dataset, can provide accurate predictions within a reasonable timeframe. Given that *SAR Optimizer* can be stopped prematurely due to time constraints, we were able to improve the quality of the proposed search plans by guiding *SAR Optimizer* towards evaluating more promising search plans first, based on their ML-predicted POS.

Although training the ML model requires an initial set of simulated search models, we found that only a small subset is needed to effectively train the model. Our experimental results indicate that *SAR Optimizer + ML* outperforms the standard *SAR Optimizer*. Although we have specifically applied our approach to a maritime SAR decision support system, the method is generalizable and can be integrated into any simulation-based decision support system.

Our contribution consists of developing, testing and evaluating a new approach integrating ML to partially replace computationally expensive simulations in the Canadian SAR planning DSS. Future work includes extending our approach to multiple SRUs on scene.

References

Abi-Zeid, I., Morin, M. and Nilo, O. (2019). Decision Support for Planning Maritime Search and Rescue Operations in Canada. In Filipe, J., Smialek, M., Brodsky, A., Hammoudi, S. Eds Proceedings of International Conference on Enterprise Information Systems, Vol. 1, 316-327, Heraklion, Greece. https://doi.org/10.5220/0007730303280339

Fisheries and Oceans Canada (2009). Canadian Coast Guard Information Kit. 36 pages. DFO 2009-1563.

Kuhn, M., & Johnson, K. (2013). Applied predictive modeling (Vol. 26, p. 13). New York: Springer.

Laperrière-Robillard, T., Morin, M., & Abi-Zeid, I. (2022). Supervised learning for maritime search operations: An artificial intelligence approach to search efficiency evaluation. Expert Systems with Applications, 206, 117857.

Russell, S., Norvig, P., 2021. Artificial Intelligence: A Modern Approach, Global Edition 4th, 4th ed. Pearson.

Stone, L. D., Royset, J. O., & Washburn, A. R. (2016). Optimal Search for Moving Targets (Vol. 237). Springer International Publishing. https://doi.org/10.1007/978-3-319-26899-6

Xu, X., Turner, C. A., & Santee, W. R. (2011). Survival time prediction in marine environments. Journal of Thermal Biology, 36(6), 340–345. https://doi.org/10.1016/j.jtherbio.2011.06.009

Acknowledgments

The authors wish the ASPT Working group and the SMCs for their support during the project. Special thanks to the project manager, the technical team from the Canadian Coast Guard College in Sydney, and to Oscar Nilo for their support.

Funding: The SAR Optimizer decision support system was developed under Canadian Government, contract number FP802-150046 to Neosoft Technologies. This research was funded by Natural Sciences and Engineering Research Council of Canada [grant number RGPIN-2021-03495, DGECR-2021-00189] and by Fonds de recherche du Québec – Nature et technologies [grant number 2017-B3-199085].