Master's Thesis

Predicting the Location of Injured People in Disaster Zones using Deep Learning

Sangwoo Ha

Department of Computer Science and Engineering

Graduate School of UNIST

2020

Predicting the Location of Injured People in Disaster Zones using Deep Learning

Sangwoo Ha

Department of Computer Science and Engineering

Graduate School of UNIST

Abstract

A large-scale disaster such as earthquakes and tsunami can cause billion-dollar destruction to a city and kill many people. To mitigate the dead troll, fast disaster response to rescue survivors in a disaster zone is of paramount importance. However, it is difficult to find the location of the injured people in a disaster zone due to the debris and smoke in collapsed buildings as well as the disruption of communication networks. This can cause poor decisions of the disaster response team about where to deploy the rescue personnel and allocate the resource. Therefore, we propose to develop an AI system to predict the location of injured people in a disaster area.

In this research, our system has three major parts: (1) the prediction of the density of injured people in a grid; and (2) the strategy of the rescue team to search for injured people; and (3) the deployment the rescue team to search the location of the most density injured people area according to the first and second part. In the first part, we developed a deep learning software package that consists of state-of-the-art deep learning techniques such as attention module and data annotation to predict the density of injured civilians. Our work uses a disaster simulator called *RoboCup Rescue Simulation (RCRS)*. To predict the density of injured people in RCRS, we train the machine learning model using the two cases of the image data: (1) single image frame such as a satellite image; and (2) multiple image sequence frame such as disaster video clip. Furthermore, we evaluate our ML model in the other two domains: (1) the prediction of the location of crime in Chicago; and (2) the prediction of the location of RSNA Pneumonia.

In the second part, we propose the Treasure Hunt Problem. In RCRS, the rescue team has to search more than one injured people and it is a complicated multi-agent problem. Therefore, study a simpler problem called the Treasure Hunt Problem, in which there is only one rescue crew search the only one injured civilian. In this problem, we assume that the location of the treasure is determined based on the probability distribution, and the ML model predicts the distribution of probability that treasure exists for each coordinate within the map. To solve this problem, we propose two search strategies that makes use of the ML model to improve the effectiveness of a search mission: (1) *the probabilistic greedy search* that the hunter searches preferentially for the cell with the highest probability of existing treasure given by ML model; and (2) *the probabilistically admissible heuristic A* search* that the hunter searches the cell determined by heuristic A* search with the probability of existing treasure given by ML model.

In the last part, we merge the first and second parts to search for the location of the most density injured people area. To predict the location, we predict the number of injured people with several ML models used in the first part and we convert the injured people density predicted to the probability distribution. And the rescue team search the most density injured people area according to the search strategy of the second part based on this probability distribution.

Contents

List of Figures

I Introduction

This master's thesis reports on the evaluation of the AI system to predict the location of injured people in disaster zones with the path-finding search strategy and machine learning model. The AI system is developed as parts of help the disaster management especially USAR with the disaster simulator called RCRS, which will be introduced before describing the problem statement, research objective, context, and further document structure

1.1 Urban Search and Rescue (USAR)

Disaster causes damage, ecological disruption, loss of human life, deterioration of health and health services on a scale sufficient to warrant an extraordinary response from outside the affected community or area. They are usually difficult to predict and it is even more challenging to prevent them from happening. These characteristics demand disaster management strategies to be in place for the mitigation of damaging consequences when a disaster happens. Urban Search and Rescue (USAR) [1], which is one of the most important tasks in disaster management strategies. The goal of USAR is to rescue the number of people as many as possible at the least amount of time while minimizing the risk to the rescuers.

1.2 The RoboCup Rescue Simulation (RCRS)

The RoboCup Rescue Simulation (RCRS) is an official simulator used RoboCup competition, a world-class robot competition. It is a large-scale multi-agent system that aims to study earthquake disaster response and support the emergency decision making by the rescue crew. In the RCRS, the simulator simulates the earthquake occurs in the city according to a specific disaster scenario.

1.3 Problem Statement

This research aims to address the following three questions: (1) How to train the best machine learning model for making predictions about the location of the injured people in RCRS? (2) Given a machine learning model for predicting the location of injured people, how to find a path to search for injured people in the least amount of time? (3) What is the optimal trade-off between the precision of the machine learning model for locating injured people and the time for searching for injured people?

1.4 Research Objective

The research objective should formulate a means of providing a solution to the research problem. As a starting point, this paragraph compiles a set of solution requirements. Research the objective is subsequently formulated. The research objective should work towards satisfying two solution requirements: (1) It should minimize the human burden as minimizing the time

required for the rescue in the USAR task; and (2) It should evaluate the requirements of disaster relief as a real-world solution on criteria. The following paragraphs construct a research objective by examining these solution requirements more closely. Subsequent to the considerations above, the research objective can be formulated as follows: (1) Predict the location of injured people in disaster zones with the machine learning model; and (2) Find the path to reach the injured people in disaster zones at the least amount of time with the search strategy which using the machine learning model. The first solution requirement is satisfied because the machine learning model is used for both objectives. And the second solution requirement is satisfied with the evaluation method of research objectives that should close to the real-world. This will be addressed in the research method section.

1.5 Research Method

The research method will involve developing an AI system to predict the location of injured people in a disaster area. The AI system has three major parts: (1) the prediction of the density of injured people in a grid; and (2) the strategy of the rescue team to search for injured people; and (3) the deployment the rescue team to search the location of the most density injured people area according to the first and second part.

In the first part, we also developed the deep learning software package that consists of state of the art deep learning technique such as attention module and data annotation to predict the density of injured civilians. Our work uses the virtual disaster simulator called *RoboCup Rescue Simulation (RCRS) [2]* because of hard to get the actual disaster data set such as satellite images of a disaster zone. The RCRS is a large-scale multi-agent system that aims to study disaster response in an earthquake. Its main purpose is to provide emergency decision support by integration of disaster information, prediction, and planning. Furthermore, RCRS shows virtual disaster situations as image data which represent information of disaster situation such as rescue team, fire, building collapse debris. Since one of the deep learning models, convolutional neural networks (CNNs) are able to extract the geometric structure of a target so that it is expected to be an advantage of this kind of geometric information to predict. And we divide the simulation screenshot image of the disaster situation by grid and the machine learning model predict the number of injured people in each cell to predict the density of injured civilian in RCRS. Therefore, we train the machine learning model using simulation screenshot images to predict the number of injured people in each grid cell in RCRS. After predicting the number of injured people in each cell, the AI system deploys the rescue team to search for injured people according to the density prediction.

In the second part, we propose the Treasure Hunt Problem to study the search strategy to search the individual injured people at the least amount of time according to the density predict in the first part. In this problem, to search the individual injured people, we consider the machine learning model is given to predict the location of injured people. And in the real disaster situation, the machine learning model has to predict the location of more than one

civilian who needs rescue. However, this is a complicated multi-agent problem. To simplify the problem, we assume two cases: (1) the rescue team searches the only one injured people and this person cannot move; and (2) the machine learning model predicts the injured people exist or not at each location as the probability distribution. To solve this problem, we propose the search strategy to find the injured people based on the probability distribution predict from the machine learning model: (1) *The probabilistic greedy search* that the hunter searches preferentially for the cell with the highest probability of existing treasure given by machine learning model; and (2) *The probabilistically admissible heuristic A* search* that the hunter searches the cell determined by heuristic A* algorithm with the probability of existing treasure given by machine learning model model.

In the last part, we merge the first and second parts to search for the location of the most density injured people area. In the second part, the rescue team search injured people according to the specific search strategy with the machine learning model. And this machine learning model used in the search strategy is based on the multivariate Gaussian distribution. However, in this part, we replace the machine learning model used in the search strategy to the machine learning model studied in the first part. We convert the injured people density predicted by the machine learning model to the probability distribution. And the rescue team searches the most density injured people area according to the search strategy of the second part (i.e., *The probabilistic greedy search* and *The probabilistically admissible heuristic A* search*) based on this probability distribution.

1.6 Document Structure

This thesis is organized as follows. After presenting the related work in Section II, we present the technical background of this study in Section III. And as the first part of our thesis, we explain how to predict the number of injured people in Section IV. In Section V, as the second part of our thesis, we present the Treasure Hunt Problem and explain our search strategy to solve this problem. And In Section VI, as the last part of our thesis, we merge the first and second parts to search for the location of the most density injured people area. Last, we summarize and present future works of study in Section VII.

II Related Work

The main purpose of our ongoing research is minimizing the time to rescue people in a disaster situation. And it is also clear that search problems are related to rescue problems. Stone [3] deriving the optimal search plan, this article discusses the relationship between locally optimal and uniformly optimal plans. In [3], an approach to search planning is presented. This discussion demonstrates the application of search theory to actual search planning. The six steps of search planning listed in this work are the computation of the prior target location distribution, estimation of sensor capabilities, determination of the detection function, search-plan development, updating for search feedback, and estimation of search effectiveness.

The goal of USAR is to rescue the greatest number of people in the shortest amount of time while minimizing the risk to the rescuers. And there are software packages that can help achieve this goal efficiently. One of the most famous software packages, the RoboCup rescue simulator [2] (RCRS) is a large-scale multi-agent system that aims to study disaster response in an earthquake. Its main purpose is to provide emergency decision support by integration of disaster information, prediction, and planning. There are many multi-agent research problems that can be investigated using the RoboCup Rescue simulation package [2]. Furthermore, there's a competition called RoboCup Rescue Simulation League that uses this simulation. The goal of this competition is to take this technological and scientific challenge and extend current rescue strategies with planning, learning, and information exchange capabilities needed to coordinate their efforts and to accomplish the rescue mission as a team. Over the years the winning entries in the competition showed a strong focus on highly optimized computations for multi-agent planning and model-based prediction of the outcome of the ongoing incidents. Several techniques for multi-agent strategy planning and team coordination in dynamic domains have also been developed based on the rescue simulator. Task allocation problems inherently found in this domain were first described in [4]. The solutions developed for the distributed constraint optimization problems (DCOPs) encountered during the simulations were evaluated in [5]. The coalition formation with spatial and temporal constraints (CFST) model and the state-of-the-art DCOP algorithms used for solving CFSTs were given in [6]. The problem of scheduling rescue missions was identified and described together with a real-time executable solution based on genetic algorithms in [7]. Furthermore, there has been substantial work on building information infrastructure and decision support systems for enabling incident commanders to efficiently coordinate rescue teams in the field. For example, Schurr et al. introduced a system based on software developed in the rescue competitions for the training and support of incident commanders in Los Angeles [8].

In the search problem, the case of the target does not respond to the searcher's action and does not move any location. In this study, the Treasure Hunt Problem matches this case of search problem and there is much research to solve this type of problem. The objective of this type of problem is often to maximize the probability of detection or to minimize the cost (or time) of the search. the single searcher should find the target to minimize the time within

an environment of defined coordinates. And, we do not consider continuous time and space. Cost in search problems, which is an important time in disaster relief. Therefore we consider the situation in which the expected cost is to be minimized. Black [9] addresses this problem when the probability of detection for each cell is constant. In his paper, the problem definition follows this: a single target is one of several regions and cannot move, give the prior probability that the target is in each region and there is conditional miss probability, the probability that a target in some region will not be detected on a single search there. The cost per search in each region, what sequence should regions be searched to minimize the expected cost of the search? His solution is the application of Bayes rule, the policy with the minimum expected cost is searcher always search in the region for which the posterior probability (given the failure of earlier search) of finding object divided by the cost is maximum. Chung et al. [10] propose a formulation of the spatial search problem, where a mobile searching agent seeks to locate a stationary target in a given search region or declare that the target is absent. The objective is to minimize the expected time until this search decision of the target's presence (and location) or absence is made. In this paper, they present a Bayesian formulation of the probabilistic search of an area and investigate several search strategies, including some that are motivated by natural searching-systems.

In the game theory, Princess Monster game [11] is a trace avoidance game played by two players in the region. The monster looks for the princess. The time to search for the princess is a reward. Both monsters and princesses are in the darkroom, and they know a room of boundaries. The monster can catch the princess if the princess within a certain range of monsters and this range is smaller than the size of the darkroom. This game was a well-known open problem until it was solved by Shmuel Gal in the late 1970s [12]. His optimal strategy for the princess is to move the princess to any place in the room, stop for a while at a time interval that is not too short or too long, then go to another independent place and repeat the process. And his optimal strategy for the monster subdivides the rooms into narrow squares, randomly choosing rectangles, and then searching for them in a specific way. And after a while, they chose a different rectangle at random randomly and independently, and so on. However, using machine learning with the probability of this problem, no one approaches this way in Princess and Monster problem. Especially, research on optimal parameters for using less accurate machine learning models has not yet been conducted. Hence, we focus on finding the relationship probability-based machine learning model's accuracy and grid size where the hidden object is located in the Treasure Hunt Problem.

III Technological Background

This section describes the technological background that is relevant to this research. The goal of this study is to predict the location of injured people using the machine learning model. Therefore, in this section, we address the technologies related to machine learning, especially related to deep learning.

3.1 Deep Learning

Deep learning, which is defined as a set of machine learning algorithms that attempts to achieve high levels of abstraction through a combination of different nonlinear transducers methods and summarize key contents or functions in large amounts of data or complex data, can be described as a machine learning field taught from a large scale. A lot of research is being done to express it in a form that a computer can understand when there is any data (e.g., pixel information in the case of an image, etc.). The 2012 Deep Learning project by Andrew Ng and Google at Stanford University succeeded in cat recognition among more than 10 million videos uploaded to YouTube using 16,000 computer processors, more than 1 billion natural networks and deep natural networks (DNN). In addition, Microsoft and Facebook are also making impressive achievements by acquiring research teams or running their own development teams.

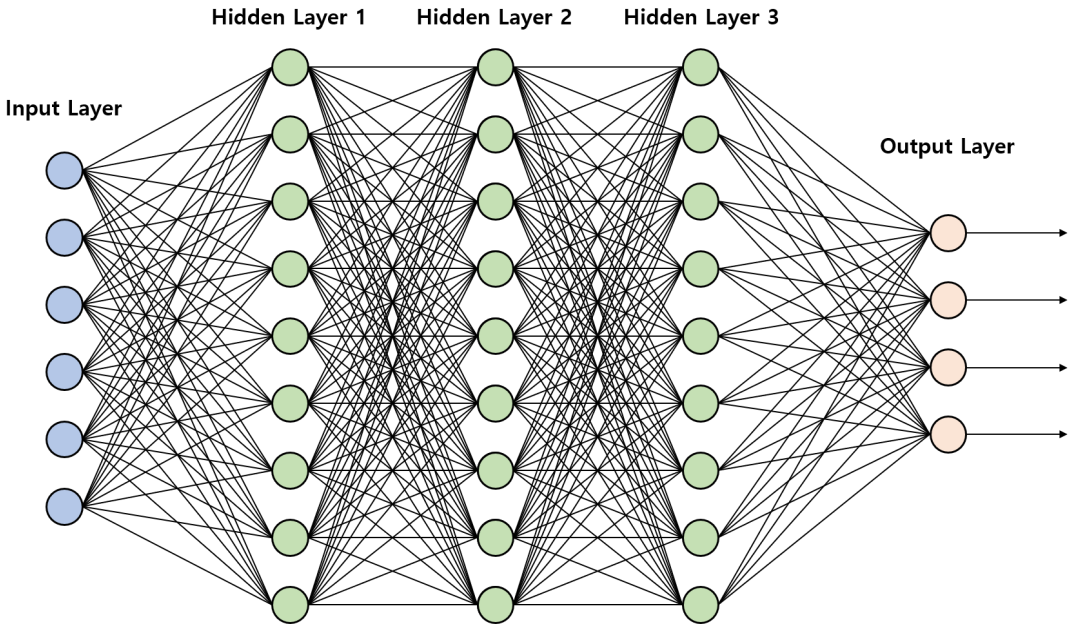


Figure 1: Illustration of Deep Learning Algorithms

3.2 Deep Neural Network

Deep Neural Network (DNN) is an artificial neural network (ANN) consisting of several hidden layers between the input layer (input layer) and the output layer. Deep neural networks can model complex non-linear relationships, as can typical artificial neural networks. For example, in an in-depth neural network structure for an object identification model, each object can be represented by a hierarchical configuration of the basic elements of the image. At this time, the additional layers can aggregate the features of the gradually assembled lower layers. This feature of deep neural networks allows more complex data to be modeled with fewer units (units, nodes) than similarly performed artificial neural networks. Examples include the application of deep neural network structures in language modeling. In the case of the synthetic Neural Network (CNN), not only is it well applied in the field of computer vision, but it is also well documented for each successful application case. More recently, it has been assessed that the synthetic neural network has been applied in the area of acoustic modeling (ASR) for automatic speech recognition (ASR) and has been more successful than existing models. Deep neural networks can be learned by standard error-reverse propagation algorithms. At this time, weights can be updated using the stochastic gradient descent using the following equation.

$$\Delta w_{ij}(t+1) = \Delta w_{ij}(t) + \eta \frac{\partial C}{\partial w_{ij}} \tag{1}$$

where is η the learning rate and C is the cost function. The cost and activation function is determined by method of learning which is supervised learning, unsupervised learning, reinforcement learning, etc. For example, when a classification problem with the supervised learning, the activation and cost functions are typically determined by the softmax function and the cross entropy function, The softmax function is defined as $p_j = \frac{\exp(x_j)}{\sum_k \exp(x_k)}$ and p_j is the class probability, x_j and x_k is the total input of j and k. The cross entropy is defined as $C = -\sum_j d_j \log p_j$. where d_j represents the target probability for the output unit j and p_j is the probability output for after apply the activation function to the j. In this study, to predict the density of the injured, we generally approach the linear regression problem rather than the problem of multi-class classification. The activation function and the cost function are determined by the Relu (Rectified Linear Unit) function and the root mean square error (RMSE). The Relu function is defined as

$$f(x) = x^+ = max(0, x) \tag{2}$$

Where x $>$ 0 which is a straight line with a slope of 1 and x $<$ 0, the output value is always zero. And the root mean square error defined as

$$RMSE(\theta_1, \theta_2) = \sqrt{MSE(\theta_1, \theta_2)} = \sqrt{E((\theta_1 - \theta_2)^2} = \sqrt{\frac{\sum_{i=1}^{n}(x_{1,i} - x_{2,i})^2}{n}} \tag{3}$$

θ_1 and θ_2 are the random vector to be compared with.

3.3 Convolutional Neural Network

In deep learning, a convolutional neural network is a deep neural network class that most commonly applies to visual image analysis. CNNs are regularized versions of multilayer perceptrons. Typically, multi-layer perceptrons refer to a fully connected network in which each neuron in one layer is connected to all neurons in the next layer. A standard method of regularization is to add some form of magnitude measurement of weights to the loss function. However, the approach to regularization is different because CNN uses tiered patterns within data to assemble more complex patterns using smaller, simpler patterns.

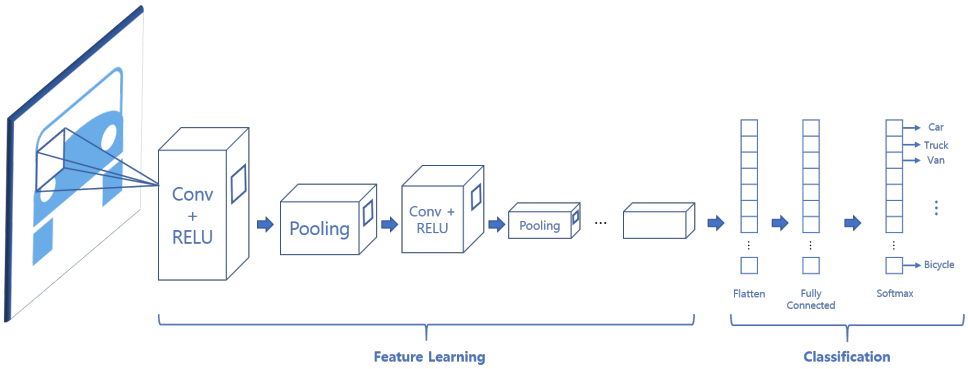


Figure 2: Illustration of Convolutional Neural Network (CNN)

Convolutional networks are given a hint by biological processes, and the connection patterns between neurons are similar to the animal's visual night tissue. CNNs use relatively little preprocessing compared to other image classification algorithms. This means that filters designed manually with conventional algorithms are learned by the network. The prior knowledge or independence from the human effort in designing this function is a great advantage.

3.4 Recurrent Neural Network

Recurrent Neural Network (RNN) is an artificial neural network class in which connections between nodes form a flow graph along a time sequence. By this, it can represent time-dynamic behavior. Unlike feedforward networks, RNN can handle input sequences using internal states. This allows applying to tasks such as linked handwriting recognition, voice recognition, etc. The term "recurrent neural network" is used indiscriminately to refer to two similar wide-area networks in a typical structure, one with finite impulses and the other with infinite impulses. Both network classes represent temporal dynamic behavior. The finite impulse recurrent network is a directed acyclic graph that can be deployed and replaced strictly feedforward neural network, while an infinite impulse recurrent network is a directed cyclic graph that cannot be unrolled. Both finite impulse and infinite impulse recursive networks can have additional memory states, and memory can be directly controlled by the neural network. The storage can also be replaced

by other networks or graphs if hat incorporates time delays or has feedback loops. This controlled state is called the gated state or gated memory and the long short-term memory networks [13] (LSTMs) and gated recurrent units. In this study, we use LSTM to predict the density of injured in video-based prediction.

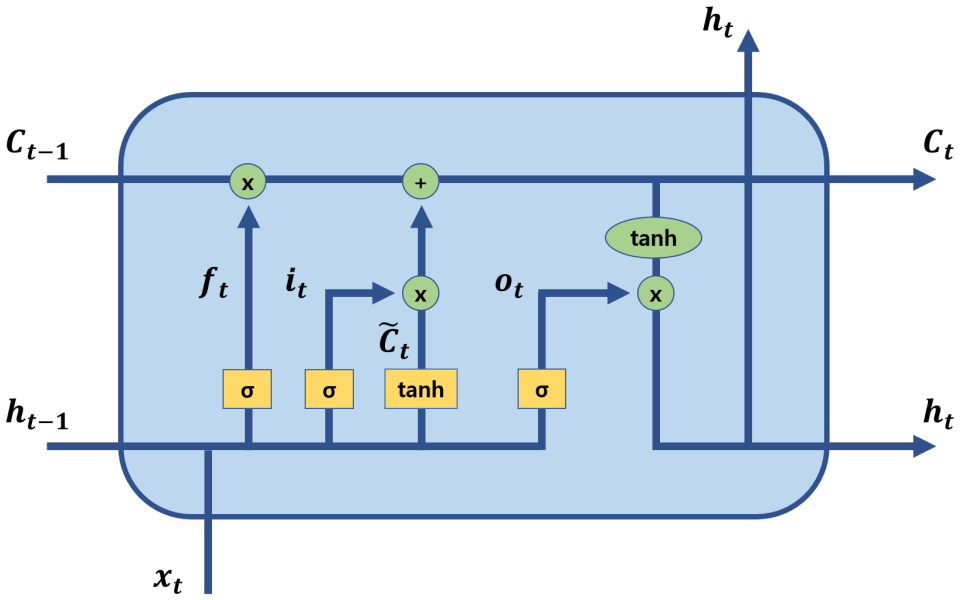


Figure 3: Illustration of Long Short-Term Memory (LSTM)

LSTM (Long Short-Term Memory) is a deep learning system that prevents the vanishing gradient problem. Long short-term memory (LSTM) is a deep learning system that avoids the vanishing gradient problem. LSTM augment by the recursive gate, usually called the "forget" gate. LSTM prevents backpropagated errors from vanishing gradient problem. Instead, errors can flow backward through unlimited numbers of virtual layers unfolded in space. In other words, LSTM can learn about tasks that require memory for events that occur before thousands or even millions of discrete time steps earlier. Problem specific LSTM similar status can be evolved. LSTM works with long delays between critical events and can handle mixed signals from low-frequency and high-frequency. Unlike previous models based on concepts such as the Hidden Markov Model (HMM), LSTM can learn to recognize language that fits the situation.

IV Locating Injured People After an Earthquake

Disasters such as earthquakes and tsunami can cause significant destruction to a city and hurt many people. To reduce the amount of the dead troll, fast disaster response to rescue survivors in a disaster zone is of paramount importance. However, the problem is all the current method to manage disaster environment is all done by human and their work burden is too much to save the people as much as possible. Especially, search for the location of people who need rescue in the disaster zone spend a considerable amount of time, which is one of the most important issues in disaster relief. The one of the solution to solve this problem, if we can predict the location of injured people in a disaster situation, it can help the rescue team to deploy the rescue team more quickly and accurately so that the time to save people can significantly reduce. Therefore, in this study, we developed a software package for predicting the location of injured people in an earthquake situation based on deep learning. However, there are limitations of conducting an experiment in a real disaster situation and collecting real disaster data set to train the machine learning model. Therefore, we predict the hidden injured in the virtual disaster simulator called RoboCup Rescue Simulation (RCRS) with deep learning.

4.1 The RoboCup Rescue Simulation (RCRS)

The objective of our study is to develop software packages to predict the location of the injured people using deep learning based on disaster simulation. To utilize deep learning techniques mainly, as many data sets as possible are needed to train the machine learning model. However, the data sets based on the actual situation of large-scale disasters are very difficult to obtain. It can make the problem of the machine learning model predict the injured people in a disaster zone. One way to solve this problem is to use virtual disaster simulation to train the machine learning model. Virtual disaster simulations allow generating enough data sets for training deep learning model on the assumption that computer resources are enough. Therefore, we survey the virtual disaster simulation suitable for studying the actual disaster response. The virtual disaster simulation should not only need to realistically simulate the disaster scenario but also can benchmark the resulting disaster response plan.

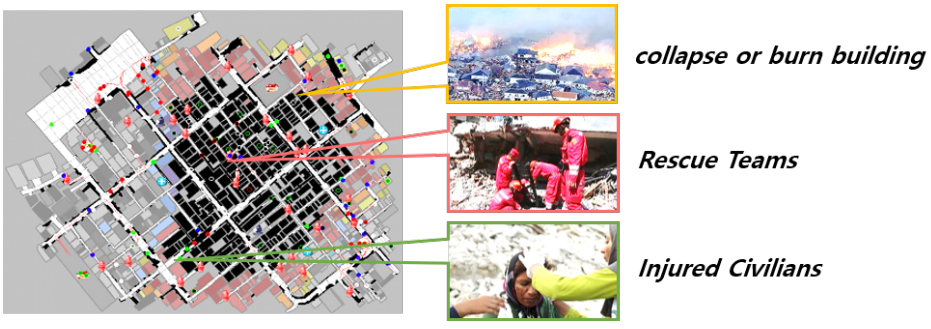


Figure 4: Illustration of RoboCup Rescue Simulation (RCRS)

The RoboCup Rescue Simulation (RCRS) is an official simulator used RoboCup competition, a world-class robot competition. It is a large-scale multi-agent system that aims to study earthquake disaster response and support the emergency decision making by the rescue crew. In the RCRS, the simulator simulates the earthquake occurs in the city according to a specific disaster scenario. The city where the earthquake will occur can be selected from the Open Street Map (OSM) because the RCRS has a program that converts maps on the OSM into maps to be used for simulation. And the earthquake disaster scenario in the simulator consists of time steps, buildings, roads, and people. In a disaster scenario, a fire starts at certain buildings, and some randomly selected buildings collapse and create debris. A fire cause injury and the debris cause the interference of the movement of the rescue team. And scenario progresses a unit of time called time step and the time step progresses, the fire spreads and people move. In the simulation, there are two groups of people: rescue teams and civilians. The rescue team consists of firefighters, police and emergency teams. The firefighters are responsible for firefighting, police removing debris from buildings, and emergency teams for transporting the injured civilians to shelters. Each rescue crew can communicate within a certain range, and according to the competition participants' disaster response plans and policies, rescue crews are moved.

4.2 Hidden Injured Problem

In RCRS, there are civilians in the disaster zone random location. And as the disaster progresses, the building collapses or fires cause injured civilians. Therefore, we predict the location of hidden injured civilians in RCRS. And the civilians have information called Health Point (HP) that shows how much they are injured status. Due to the disaster situation in simulators, likes fires spread or buildings collapse, civilians' HP going to lower. We set the HP threshold which determines the civilians are injured or not. If one civilian' HP lower than this threshold, we determine this civilian is injured. In this study, we propose the "Hidden Injured Problem" problem whose goal is to find the location of injured people. In this problem, we suppose the x and y are the coordinate of the map where the $1 \leq x \leq X$ and $1 \leq y \leq Y$. And the location of civilians is chosen randomly, the location of the rescue team and initial fire start building in the simulation is fixed. In addition, each time step $\boldsymbol{t}$ exists in the simulation, as time step increase, fires spread or building collapses become more severe, thus increasing the number of injured civilians. Accordingly, the rescue team moves and tracks the location of the injured people. Therefore, even if the initial location of civilians is random, the location of the occurrence of the injured civilian will have a constant pattern as the disaster scenario progresses.

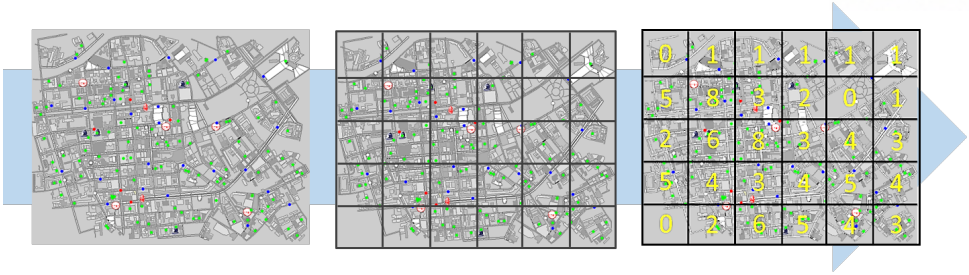


Figure 5: Predict the density of injured people in RCRS

In this study, we use the machine learning model $\boldsymbol{M}$ to predict the location of the injured. And predict the exact location of injured civilians requires considerable computational resources and complexity, so that we divide the simulation map into grid and predict the density of the injured people in each grid cell. This can significantly shorten computational resources, complexity and the time required for training. Furthermore, we expected that the accuracy of the prediction of the injured civilians location will also be increased. Therefore, we assume that the map divided by the grid size of $M \times N$, and the number of injured civilians in each grid cell $N_{i,j}$ where the $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, N$. And $N_{i,j}$ is the number of injured civilians who locate at (x_t, y_t) within the range of $\frac{X*(i-1)}{M} \leq x_t < \frac{X*i}{M}$ and $\frac{Y*(j-1)}{N} \leq y_t < \frac{Y*j}{N}$. And we predict the number of injured civilians in each grid cell when the given data D by time step with machine learning model $N_{i,j} = \boldsymbol{M}(D, \boldsymbol{t})$.

4.3 The Machine Learning Model

Machine learning techniques, and deep learning, in particular, have attracted much attention in recent years due to successful application to various actual problems that had been regarded as difficult for several decades due to the complexity of the problem. In [14], it was shown that deep learning surpassed traditional machine learning methods using handmade feature extraction in a famous 'natural' image classification task [15]. It has been suggested that deep neural networks can automatically learn the important features of a problem, such as the geometrical constraints of pixels in images. For example, a deep neural network that was trained on images of humans developed neurons that responded to the human face or body. In this study, we develop the machine learning model to the density of the injured civilians. The objective of the study is finding the best machine learning model to predict the density of the injured civilians in the disaster zone. In this study, we divide this goal into two main categories: *(1) predicting the location of the injured civilians in the RCRS using image-based data* such as satellite images in the disaster zone or simulation map images. *(2) predicting the location of the injured civilians in the RCRS using video-based data* such as CCTV video in the disaster zone or simulation map image sequence. And we expected that using Convolution Neural Network (CNN) with Recurrent Neural Network (RNN) is a suitable neural network to predict.

4.4 Image-Based Prediction

To predict the density of injured civilians in RCRS using image-based data, we train the machine learning model using simulation screenshot images. We consider that all frames of video clips in disaster situations are independent images. Therefore, we choose one frame randomly in images sequence of a disaster scenario in RCRS to train the machine learning model. The model's inputs are simulated images created from RCRS. And CNN extracts images features, fully-connected layer output the number of injured people vectors in each grid cell.

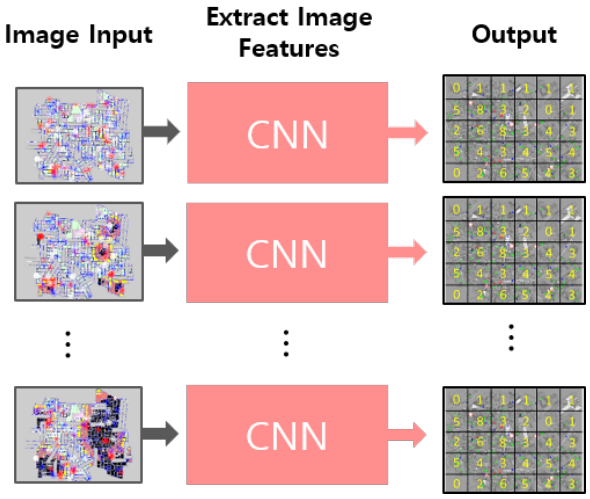


Figure 6: Illustration of CNN based machine learning model

In this study, we research the method of enhancing the performance of the machine learning model to predict the number of injured civilians in RCRS. And we evaluate the prediction performance of the different machine learning models to find the best performance of the machine learning model. To enhance the performance of the machine learning model, first, we find the best CNN model to extract the images feature. Second, we expect to see better performance by applying the attention mechanism to the best CNN model found in the first step. Finally, we apply our own data annotation method to simulation data set for training machine learning model.

4.5 Video-Based Prediction

Predicting the density of injured people in RCRS using video-based data, we train the machine learning model using a simulation screenshot image sequence. We consider that all frames of video clips are used for training. Therefore, we choose nine frames randomly in images sequence of a disaster scenario in RCRS to train the machine learning model and predict the density of the 10th frame which is the next frame of 9th of the nine frames. The machine learning model's inputs are simulated images created and filtered from Plug-in1 and extract images feature using

CNN. Afterward, the Long-short memory (LSTM) extract sequence image feature from CNN. And fully-connected layer output the number of injured people vectors in each grid cell. The different from image-based predictions is add the LSTM layer to sequence learning on video.

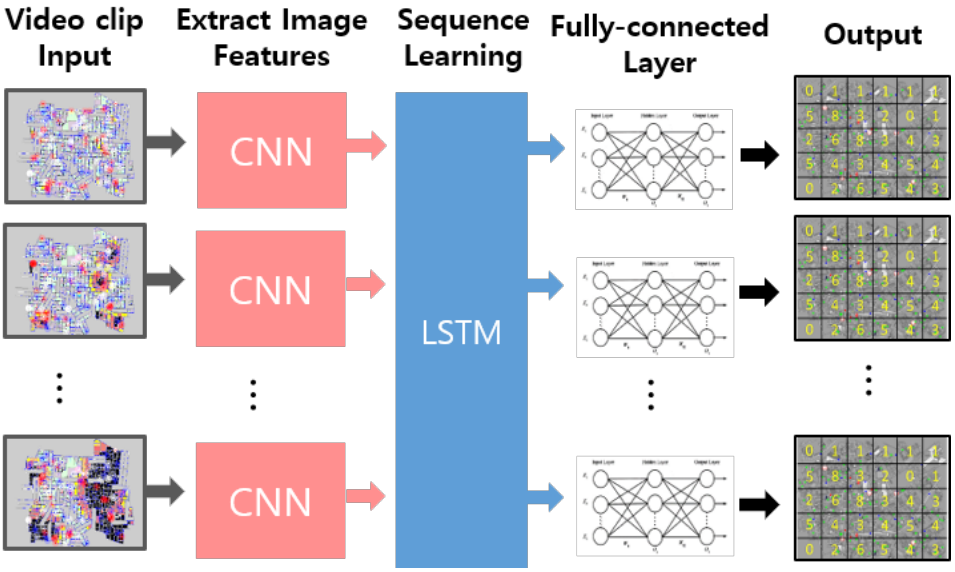


Figure 7: Illustration of CNN-LSTM based machine learning model

In this study, we research the method of enhancing the performance of the machine learning model to predict the number of injured civilians in RCRS. And we evaluate the prediction performance of the different machine learning models to find the best performance of the machine learning model. To enhance the performance of the machine learning model, first, we find the best CNN-LSTM model to extract the images feature. Second, we expect to see better performance by applying the attention mechanism to the best CNN-LSTM model found in the first step. Finally, we apply our own data annotation method to simulation data set for training machine learning model.

4.6 Find the Best CNN model

To enhance the performance of prediction, we need to find the best CNN model to extract the images feature. Therefore we evaluated performance using several CNN models that performed very well in recent image processing studies. In recent image processing related to deep learning researches has mainly investigated three important factors of networks to enhance the performance of CNN models: depth, width, and cardinality. ResNet [16] stacks the same topology of residual blocks along with skip connection to build an extremely deep architecture. Wide-ResNet [17] shows that width is another important factor to improve the performance of a model. Zagoruyko and Komodakis [17] propose to increase the width of a network based on the ResNet architecture. They have shown that a 28-layer ResNet with increased width can

outperform an extremely deep ResNet with 1001 layers on the CIFAR benchmarks. ResNet [18] and Xception [19] come up with to increase the cardinality of a network. They empirically show that cardinality not only saves the total number of parameters but also results in stronger representation power than the other two factors: depth and width. In this study, we evaluate our machine learning model with several CNN which is the state-of-art in image processing.

4.7 The Attention Mechanism

Apart from the CNN model of depth, width, and cardinality factors, we investigate a different aspect of the CNN architecture design, attention. The significance of attention has been studied extensively in the previous literature [20–25]. Our goal is to increase the focus image feature power of CNN by using attention mechanism: focusing on important features and suppressing unnecessary ones. Hu *et al.* [26] propose the *Squeeze and excitation (SE)* module which use global average-pooled features to compute channel-wise attention. Woo *et al.* [27] propose *Convolutional Block Attention Module (CBAM)*, which use max-pooled features as well that is a simple yet effective attention module for feed-forward convolutional neural networks. Furthermore, we proposed the new attention module called *Grid Convolutional Block Attention Module (GCBAM)*. This attention module divides the two-dimension image feature vector to a certain grid size. And use max-pooled features as well that is a simple yet effective attention module for feed-forward convolutional neural networks. In this study, we use SE, CBAM and GCBAM attention module to the model chosen by the previous experiment which performs well.

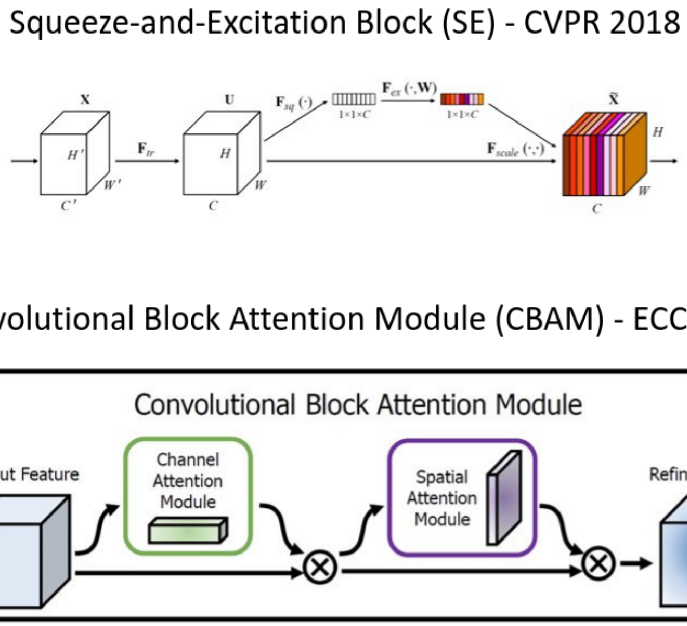


Figure 8: Illustration of SE and CBAM attention module

4.8 Feature-Highlight Data Annotation

In deep learning with image processing, the deep learning model train images according to the feature of images. And the accuracy of the deep learning model is affected by whether the model trains the feature of images well. In this study, we propose the data annotation method to help the model training more about the feature of images within the images for predicting the results well. In RCRS, injured civilians are occurred by fire or collapse of a building. And the rescue teams patrol to find injured civilians. Therefore, we expect that the important image of features to predict the location of injured civilians is the location of the object which relates to the injured civilians such as the location of collapsed buildings, locations of fires and rescue teams. Therefore, we study the data annotation method that highlighting the important location which is related to the result in the image. We expect that this annotation method allows the machine learning model to focus more on the feature of images in the training process. Therefore, we resizing, cropping, and highlighting of simulation image data which contribute to the increase in the efficiency of training and prediction accuracy of the deep learning model.

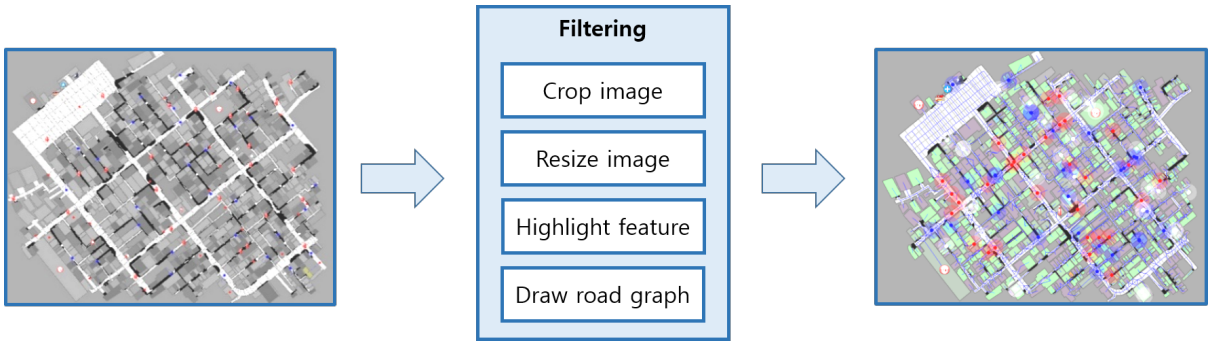


Figure 9: Feature-Highlight data annotation

4.9 Chicago Crime Location Prediction

In this study, we tested our machine learning model in another domain called Chicago Crime Dataset [1]. This dataset reflects reported incidents of crime that occurred in the City of Chicago from 2001 to the present. And this dataset includes latitude and longitude location information for crimes committed. In order to protect the privacy of crime victims, addresses are shown at the block level only and specific locations are not identified. However, since the approximate location is displayed, it is sufficient to use. In this study, we predict the crime location in Chicago with our machine learning model. To train the machine learning model, we create the image dataset which is converted Chicago crime location datasets in query form into satellite images. And we use the Chicago crime location dataset of longitude and latitude with google static map

[1] https://www.kaggle.com/chicago/chicago-crime

API [2] to convert the image. The picture below is an example of a data set that we created. In that picture, a blue mark indicates where the crime occurred and each image is taken using a satellite to capture an area within 320m x 320m of the city of Chicago.

Figure 10: Image of Chicago Crime dataset using Google static map API service

To train the machine learning model much better, we highlight the satellite image. We've highlighted data that is highly relevant to where the crime occurred. For example, each region highlighted the map in transparent color according to the average income of the area. We divided the map into 8x8 grid sizes (i.e., each cell size is 60m x 60m) and calculated the average income for each cell. Then we highlight the transparent color to each cell depends on the average income. And the income data is based on the Kaggle USA household income dataset [3]. Furthermore, we highlighted the important building location such as a government agency which is a police station, a fire station, a bus stop, etc. and a location for the occurrence of other crimes, such as speed and red light violation location in the map.

[2]https://developers.google.com/maps/documentation/maps-static/intro
[3]https://www.kaggle.com/goldenoakresearch/us-household-income-stats-geo-locations

Figure 11: Example of highlighting Chicago crime location satellite images

In this study, we also divide a Chicago Crime of Google map image dataset which we created into the grid size M x N. And we predict how much areas of each grid cell contain a crime location bounding box. In this problem, we assumed that the box which width and height to 40m and center coordinate is the crime occurrence coordinate is a crime location bounding box. And we suppose the several values: (1) X and Y axes of the Chicago Crime of Google map image coordinates are (x, y), (2) the coordinates of the crime location bounding box are (x_p, y_p). (3) the coordinates of each grid cell is (x_t, y_t) that within the range of $\frac{X*(i-1)}{M} \leq x_t < \frac{X*i}{M}$ and $\frac{Y*(j-1)}{N} \leq y_t < \frac{Y*j}{N}$. (4) The value of how much areas of each grid cell contain crime location bounding box is $N_{i,j}$ where the $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, N$. Therefore, we label $N_{i,j}$ to the area of (x_t, y_t) divided by the overlaps area of the crime location bounding box (x_p, y_p) and use our machine learning model with given image data to predict $N_{i,j}$.

4.10 RSNA Pneumonia Detection Challenge

In this study, we tested our machine learning model in another domain called RSNA Pneumonia Detection Challenge [4]. In this competition, the participant challenged to build an algorithm to detect a visual signal for pneumonia in medical images. Specifically, your algorithm needs to automatically locate lung opacities on chest radiographs. Pneumonia accounts for over 15% of all deaths of children under 5 years old internationally. In 2015, 920,000 children under the age of 5 died from the disease. In the United States, pneumonia accounts for over 500,000 visits to

[4]https://www.kaggle.com/c/rsna-pneumonia-detection-challenge/overview

emergency departments [28] and over 50,000 deaths in 2015 [29], keeping the ailment on the list of top 10 causes of death in the country. While common, accurately diagnosing pneumonia is a tall order. It requires the review of a chest radiograph (CXR) by highly trained specialists and confirmation through clinical history, vital signs, and laboratory exams. Pneumonia usually manifests as an area or areas of increased opacity [30] on CXR. However, the diagnosis of pneumonia on CXR is complicated because of a number of other conditions in the lungs such as fluid overload (pulmonary edema), bleeding, volume loss (atelectasis or collapse), lung cancer, or post-radiation or surgical changes. Outside of the lungs, fluid in the pleural space (pleural effusion) also appears as an increased opacity on CXR. When available, a comparison of CXRs of the patient taken at different time points and correlation with clinical symptoms and history is helpful in making the diagnosis. CXRs are the most commonly performed diagnostic imaging study. A number of factors such as positioning of the patient and depth of inspiration can alter the appearance of the CXR [31], complicating interpretation further. In addition, clinicians are faced with reading high volumes of images every shift. To improve the efficiency and reach of diagnostic services, the Radiological Society of North America (RSNA®) has reached out to Kaggle's machine learning community and collaborated with the US National Institutes of Health, The Society of Thoracic Radiology, and MD.ai to develop a rich dataset for this challenge.

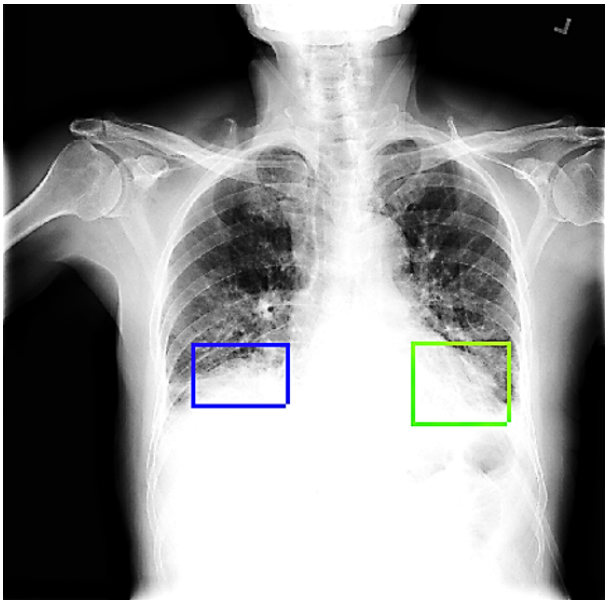


Figure 12: Sample Image of RSNA Pneumonia Detection challenge dataset

The dataset used in this challenge contains bounding box information for the patient pneumonia location. In this study, we divide a visual signal for pneumonia in the medical image which the dataset of this challenge into the grid size M x N. And we predict how much areas of each grid cell contain patient pneumonia's bounding box. In this problem, we suppose the several values: (1) X and Y axes of the medical image coordinates are (x, y), (2) the coordinates of the patient

pneumonia's bounding box are (x_p, y_p). (3) the coordinates of each grid cell is (x_t, y_t) that within the range of $\frac{X*(i-1)}{M} \leq x_t < \frac{X*i}{M}$ and $\frac{Y*(j-1)}{N} \leq y_t < \frac{Y*j}{N}$. (4) The value of how much areas of each grid cell contain patient pneumonia's bounding box is $N_{i,j}$ where the $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, N$. Therefore, we label $N_{i,j}$ to the area of (x_t, y_t) divided by the overlaps area of the patient's pneumonia bounding box (x_p, y_p) and use our machine learning model with given image data to predict $N_{i,j}$.

4.11 Experiment Results

In this section, we predict the density of injured people in RCRS and train the machine learning model using screenshot images of the simulation. We consider two cases of the image data to train the machine learning model: (1) single image frame such as a satellite image; and (2) multiple image sequence frame such as disaster video clip. Each case, we test the machine learning model to improve performance as follows: (1) comparison of the several state-of-art CNN models to find the best performance of the CNN model; and (2) comparison of the performance of the attention module model with the non-attention module model; and (3) comparison of the performance of the Feature-Highlight Data annotation with the non-Feature-Highlight Data annotation model. Furthermore, we evaluate our machine running model in the other two domains: (1) the prediction of the location of crime in Chicago; and (2) the prediction of the location of RSNA Pneumonia.

4.11.1 Image-Based Prediction: Find the Best CNN model

In this experiment, we predict the density of injured people in each grid cell within a single frame of simulation screenshot image. And we evaluate the several state-of-art CNN models to find the best performance of the CNN model. We use the disaster scenario that used at the RoboCup competition last year. This scenario is the earthquake that happens in Kobe, Japan that 163 civilians and 80 rescue crews exist within the 4km x 3km range. In this scenario, The rescue crews locate in a certain chosen building and civilians located in a randomly chosen building. And the fire starts randomly chosen six buildings. The scenario consists of 200 frames of the time step and we randomly choose one frame to train the machine learning model. We create a total of 11,000 disaster scenarios and we choose 10,000 scenarios to train the machine learning model, 1,000 scenarios to test the machine learning model. Both scenarios to train and test were not have overlapping scenarios.

For each scenario, we divide the simulation map into a grid size 4x4 and 8x8. And we compared the actual density of injured civilians in each grid cell (i.e., the ground truth) with the one predicted by the machine learning model and calculated the root mean squared error (RMSE) between the ground truth and the predicted numbers. We train the ResNet [16], Xception [19], Inception-ResNetV2 [32], InceptionV3 [33] and DenseNet [34] CNN based machine learning model to find the best performance of CNN model using Nvidia's GeForce RTX 2080

Ti graphics card. Each machine learning model trained the six epoch with the learning rate to 0.001 and the size of the input image is a 256x192 pixel.

According to the experiment, the machine learning model with Xception of RMSE is 2.1103 in the 4x4 grid which is the best performance. And the machine learning model with Inception-ResNetV2 of RMSE is 1.0508 in the 8x8 grid which is the best performance. Those models nearly double the performance of the worst-performing model. Except for the machine learning model with Inception-ResNetV2, performance did not improve as the number of parameters increased. MobileNet, on the other hand, showed better performance compared to fewer parameters. To enhance the model performance, we attach the attention module to the machine learning model in the next experiment. And for the efficient progress of the experiment, we choose three models to attach the attention module which perform well in both 4x4 and 8x8 grid based on the result of the experiment. In this study, we choose the machine learning model with Inception-ResNetV2 and Xception.

4.11.2 Image-Based Prediction: Attention mechanism

In this experiment, we compare the performance of the attention module model with the non-attention module model to see whether the attention module improves the performance of the machine learning model. And we use the same scenario used in the previous experiment. We evaluate *Squeeze and excitation (SE)*, *Convolutional Block Attention Module (CBAM)* and *Grid Convolutional Block Attention Module (GCBAM)* attention module with ResNet-50, Inception-ResNetV2 and Xception CNN based model. And *GCBAM* is a new attention module proposed in this study. For each scenario, we compared the actual density of injured civilians in each grid cell (i.e., the ground truth) with the one predicted by the machine learning model and calculated the root mean squared error (RMSE) between the ground truth and the predicted numbers.

According to the experiment, there is an improvement in performance for Inception-ResNetV2 with CBAM and GCBAM attention models. However, there is no or little improvement Xception and the model without the attention model still the best performance. We guess that due to the characteristics of the attention module that is focused more on the feature of the image seems to do not fit into model well. Therefore, we expect to improve the performance of the model by utilizing the Feature-Highlight data annotation method which is proposed in this study.

4.11.3 Image-Based Prediction: Feature-Highlight Data Annotation

In this experiment, we compare the performance of the Feature-Highlight data annotation with the non-version model. And we use the same scenario used in the previous experiment. We evaluate InceptionResNetV2 and Xception with SE, CBAM, and GCBAM attention module and Feature-Highlight data annotation. Furthermore, for each scenario compared the actual density of injured civilians in each grid cell (i.e., the ground truth) with the one predicted by the machine learning model and calculated the root mean squared error (RMSE) between the

ground truth and the predicted numbers.

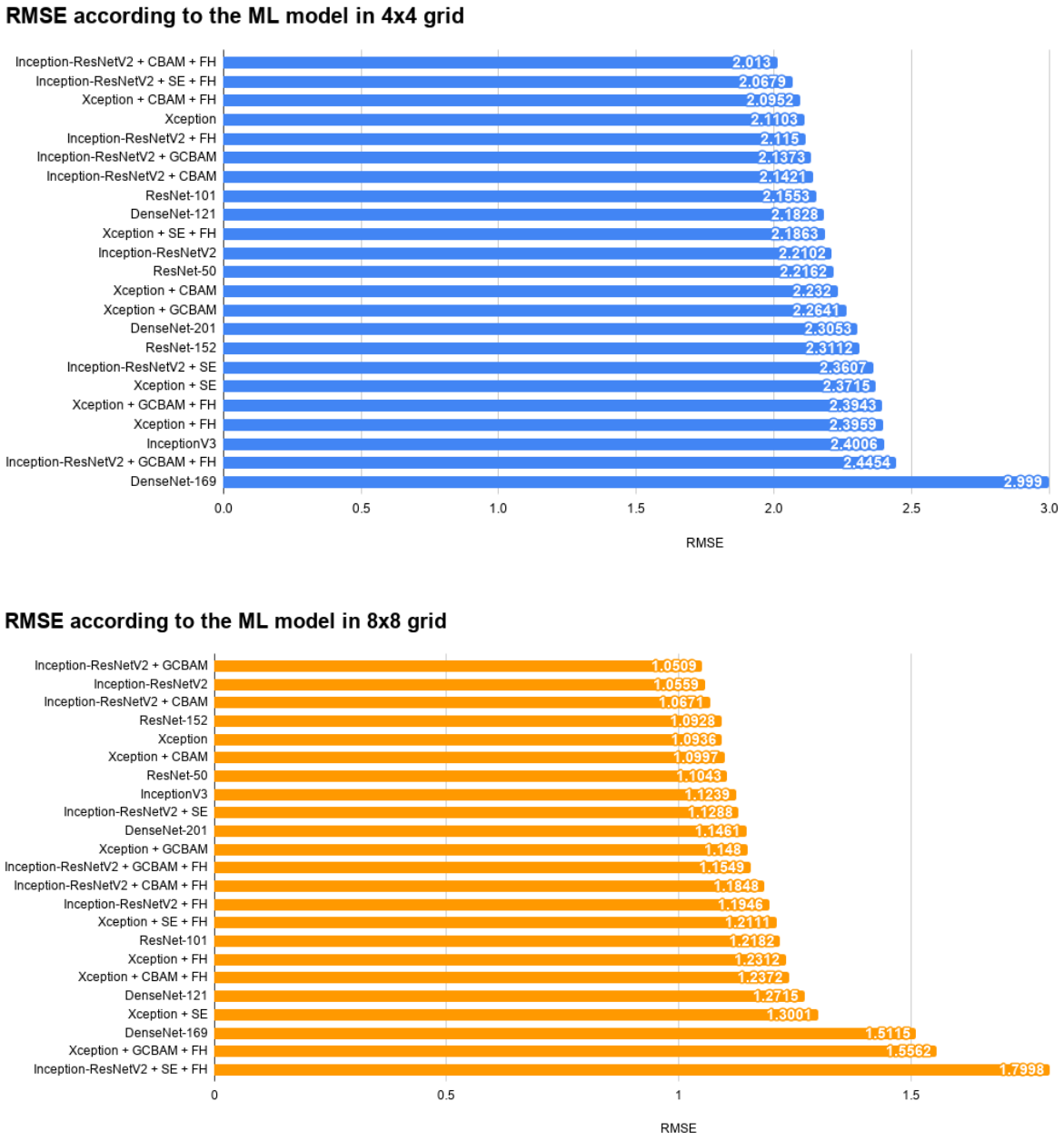


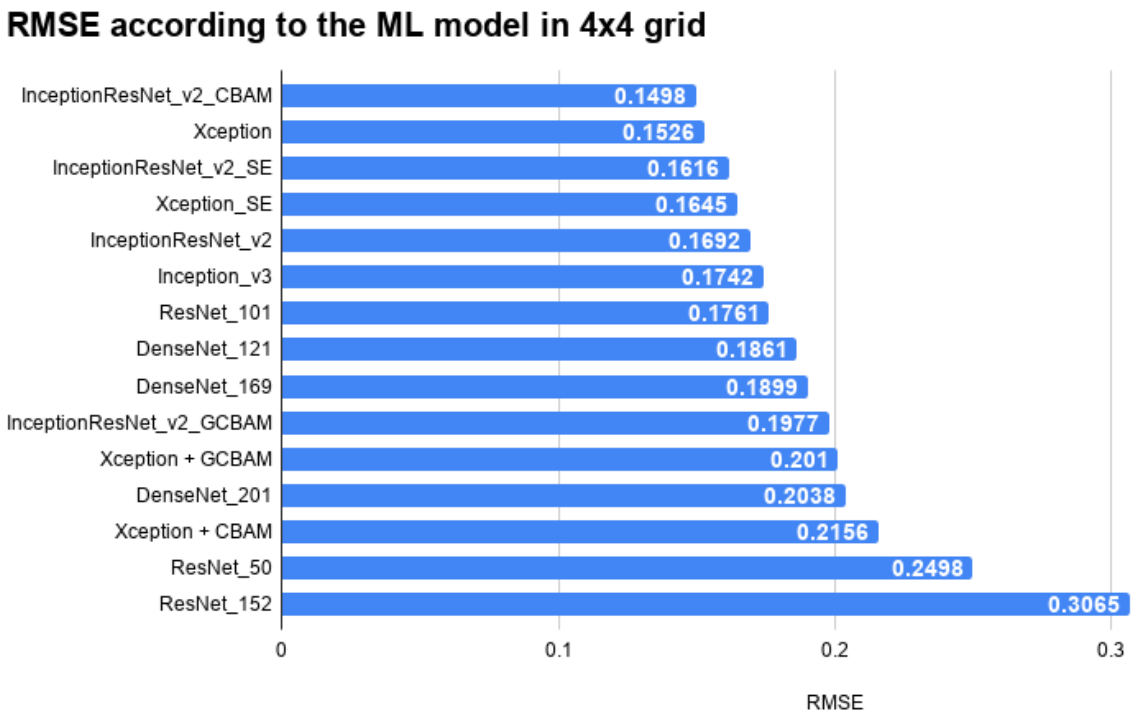
Figure 13: RMSE according to the ML model with Feature-Highlight data annotation and grid size

According to the experiment, the InceptionResNetV2 with CBAM attention module and Feature-Highlight data annotation method shown the best performance in the 4x4 grid which RMSE is 2.0130. And the InceptionResNetV2 with GCBAM attention module and Feature-Highlight data annotation method shown the best performance in the 8x8 grid which RMSE is 1.0509. Furthermore, nearly half of the machine learning model with Feature-Highlight data annota-

tion shown better performance than the original model. It means our Feature-Highlight data annotation method performed well in the RCRS.

4.11.4 Chicago Crime Location Prediction

To predict the crime location in Chicago, we create the image dataset based on the Kaggle's Chicago Crime dataset. A total of 4,300 images were created for model training, 700 images were created for model testing and we divide images into 4x4 and 8x8 grid. We use Nvidia's GeForce RTX 2080 Ti graphics card to train the machine learning model. In the optimizer part, we use Adam optimizer with a mean squared error loss function and the learning rate to 0.0001 with six epochs. As the first step of our study, we evaluate the machine learning model with the ResNet, DenseNet, Inception-ResNetV2, InceptionV3, and Xception CNN based models. And we also evaluate the Xception and Inception-ResNetV2 with SE, CBAM, GCBAM attention module models. In this experiment, we find the best performance of the machine learning model. For each step, we compared the actual the area of each grid cell divided by the overlaps area of the crime location bounding box (i.e., the ground truth) with the one predicted by the machine learning model and calculated the root mean squared error (RMSE) between the ground truth and the predicted numbers.



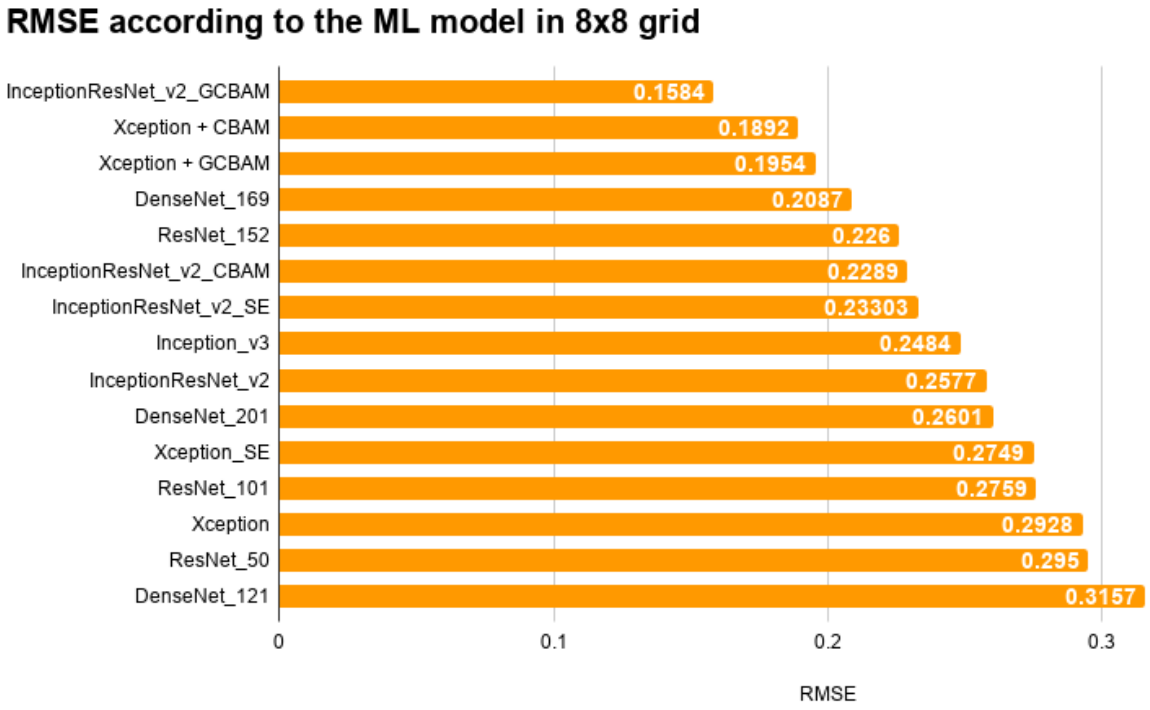


Figure 14: RMSE according to the ML model and grid size in Chicago Crime Dataset

In this experiment, almost more than half of the CNN based machine learning model improves the performance after attaching the attention module. In the grid size 4x4, Inception-ResNetV2 with the CBAM attention module is the best performance in which RMSE is 0.1498. In the grid size 8x8, Inception-ResNetV2 with our GCBAM attention module is the best performance in which RMSE is 0.1584. According to the experiment, the Inception-ResNetV2 with the CBAM attention module and the Inception-ResNetV2 with GCBAM attention module is not only the best performance of the machine learning model in Chicago Crime domain but also in the RCRS.

4.11.5 RSNA Pneumonia Detection Challenge

To predict the patient pneumonia location, we create the image dataset based on the Kaggle's pneumonia challenge dataset. A total of 6,000 images were created for model training, 600 images were created for model testing. We divide the pneumonia medical images into 16 and 64. It means the map divided into 4x4 and 8x8 grid size. We use Nvidia's GeForce RTX 2080 Ti graphics card to train the machine learning model. In the optimizer part, we use Adam optimizer with a mean squared error loss function and learning rate to 0.0001 with six epochs. As the first step of our study, we evaluate the ResNet, DenseNet, InceptionResNetV2, InceptionV3 and Xception CNN based machine learning model to find the best CNN model to predict. And we also evaluate the Xception and Inception-ResNetV2 with SE, CBAM, GCBAM attention module models. For each step, we compared the actual the area of each grid cell divided by

the overlaps area of the patient's pneumonia bounding box (i.e., the ground truth) with the one predicted by the machine learning model and calculated the root mean squared error (RMSE) between the ground truth and the predicted numbers.

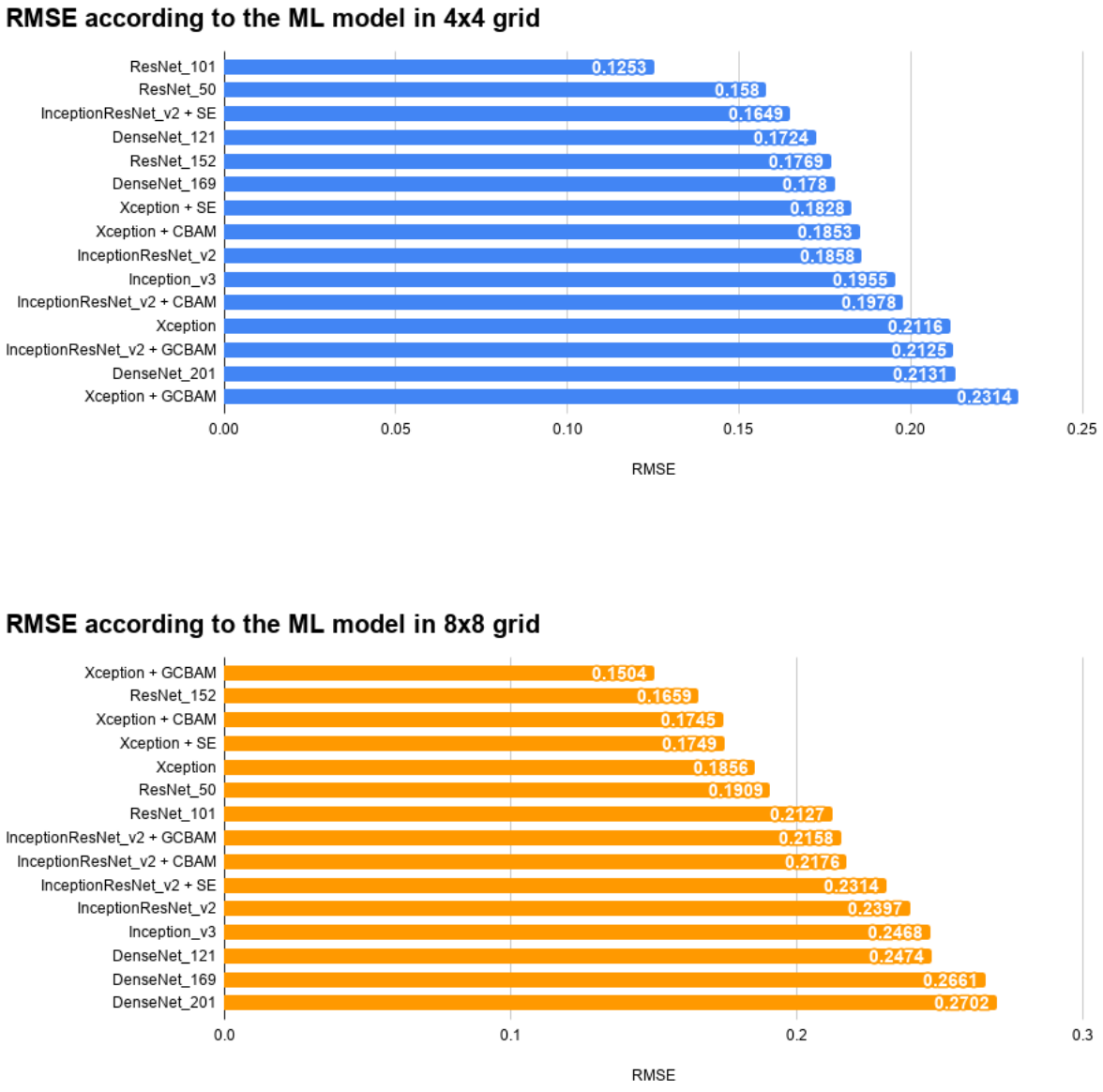


Figure 15: RMSE according to the ML model and grid size in RSNA Dataset

In this experiment, the ResNet-101 CNN based model is the best performance of the machine learning model in the 4x4 grid which RMSE is 0.1253. And the Xception with GCBAM attention module CNN based model is the best performance of the machine learning model in the 8x8 grid in which RMSE is 0.1504. According to the experiment, in the 8x8 grid size, the Xception with GCBAM attention module CNN based model is the best performance not only in the RSNA dataset but also RCRS. And in the 4x4 grid size, when we attach the attention module the performance is little increase or even decrease.

4.11.6 Video-Based Prediction: Find the best CNN model

In this experiment, we predict the density of injured people in each grid cell within the sequence of the screenshot image. And we evaluate the several state-of-art CNN models with LSTM layers to find the best performance of the CNN model. We use the disaster scenario that used at the RoboCup competition last year. This scenario is the earthquake that happens in Kobe, Japan that 163 civilians and 80 rescue crews exist within the 4km x 3km range. In this scenario, The rescue crews locate in a certain chosen building and civilians located in a randomly chosen building. And the fire starts randomly chosen six buildings. The scenario consists of 200 frames of the time step and we choose ten frames to train the machine learning model. The machine learning model predicts the density of injured civilians in the next frame of the images to train. And to train the machine learning model, we randomly choose one frame of image in each scenario and create the image sequence which includes the chosen frame with after eight frames. We create a total of 11,000 disaster scenarios and we choose 10,000 scenarios to train the machine learning model, 1000 scenarios to test the machine learning model. Both scenarios to train and test were not have overlapping scenarios. And each scenario contains 200 images, a total of 2,000,000 images to train.

For each scenario, we divide the simulation map into a grid size 4x4 and 8x8. And we compared the actual density of injured civilians in each grid cell (i.e., the ground truth) with the one predicted by the machine learning model and calculated the root mean squared error (RMSE) between the ground truth and the predicted numbers. We train the ResNet [16], ResNeXt [18], Xception [19], Inception-ResNetV2 [32], InceptionV3 [33] and DenseNet [34] CNN with LSTM layers based machine learning model to find the best performance of CNN model using Nvidia's GeForce RTX 2080 Ti graphics card. Each machine learning model trained the six epoch with the learning rate to 0.001 and the size of the input image is a 256x192 pixel.

According to the experiment, the ResNeXt-50 CNN based machine learning model is the best performance in the 4x4 grid which RMSE is 3.2219. And also the ResNeXt-50 CNN based machine learning model is the best performance in the 8x8 grid which RMSE is 1.4829. To enhance the model performance, we attach the attention module to the machine learning model in the next experiment. And for the efficient progress of the experiment, we choose three models to attach the attention module which perform well in both 4x4 and 8x8 grid based on the result of the experiment. In this study, we choose the machine learning model with ResNeXt-50 and ResNeXt-101.

4.11.7 Video-Based Prediction: Attention mechanism

In this experiment, we compare the performance of the attention module model with the non-attention module model to see whether the attention module improves the performance of the machine learning model. And we use the same scenario used in the previous experiment. We evaluate *Squeeze and excitation (SE)*, *Convolutional Block Attention Module (CBAM)* and

Grid Convolutional Block Attention Module (GCBAM) attention module with ResNeXt-50 and ResNeXt-101 CNN with LSTM layers based model. For each scenario, we compared the actual density of injured civilians in each grid cell (i.e., the ground truth) with the one predicted by the machine learning model and calculated the root mean squared error (RMSE) between the ground truth and the predicted numbers.

According to the experiment, there is an improvement in performance for ResNeXt-50 and ResNeXt-101 CNN with the LSTM layers based model in the 4x4 and 8x8 grid. However, in the 4x4 grid size, the ResNeXt-101 has improvement with CBAM and in the 8x8 grid size, the ResNeXt-50 has improvement with GCBAM. And the ResNeXt-50 still the best performance in the 4x4 grid, the ResNeXt-50 with GCBAM is the best performance in the 8x8 grid which RMSE is 1.4896. We guess that due to the characteristics of the attention module that is focused more on the feature of the image seems to do not fit into model well. Therefore, we expect to improve the performance of the model by utilizing the Feature-Highlight data annotation method which is proposed in this study.

4.11.8 Video-Based Prediction: Feature-Highlight Data Annotation

In this experiment, we compare the performance of the Feature-Highlight Data annotation with the non-Feature-Highlight Data annotation model. And we use the same scenario used in the previous experiment. We evaluate SE, CBAM and GCBAM attention module with ResNeXt-50 and ResNeXt-101 CNN with LSTM layers based model. And compare each model to apply Feature-Highlight data annotation method with non-version. Also, for each scenario compared the actual density of injured civilians in each grid cell (i.e., the ground truth) with the one predicted by the machine learning model and calculated the root mean squared error (RMSE) between the ground truth and the predicted numbers. In the graph below, FH means the Feature-Highlight annotation method.

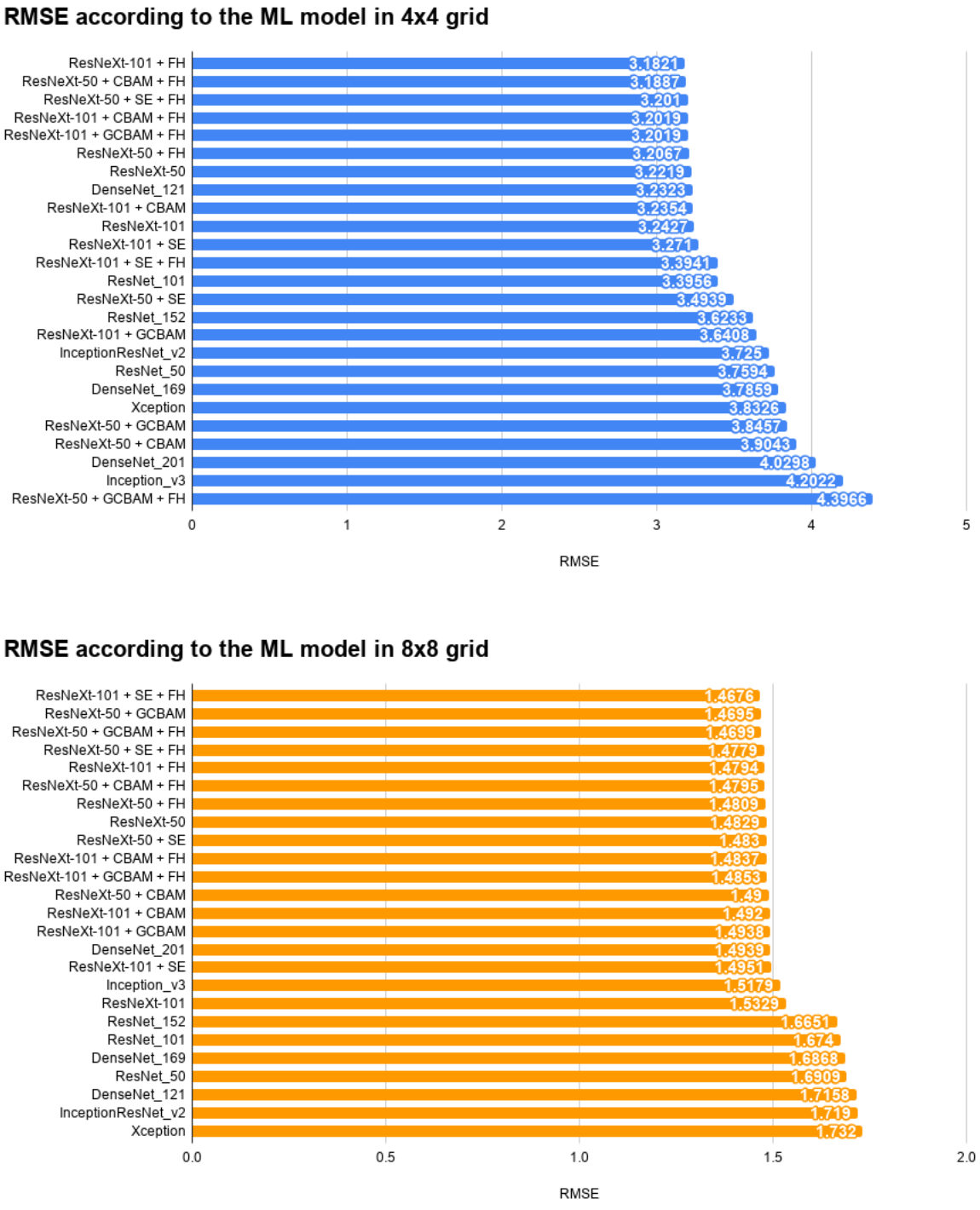


Figure 16: RMSE according to the ML model and grid size

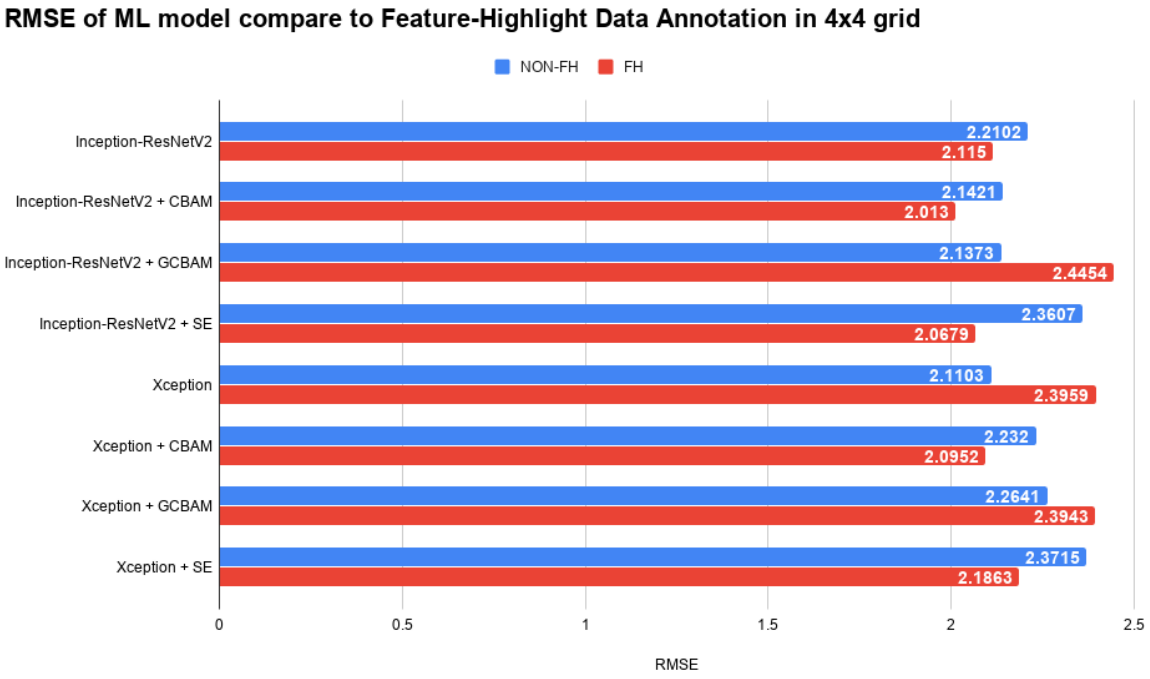
According to the experiment, the ResNeXt-101 with Feature-Highlight data annotation method shown the best performance in the 4x4 grid which RMSE is 3.1821. And the ResNeXt with SE attention module and Feature-Highlight data annotation method shown the best performance in the 8x8 grid which RMSE is 1.4876. Furthermore, more than half of the machine learning

model with FH data annotation shown better performance than the original model. It means our FH data annotation method performed well in the RCRS.

4.12 Conclusions

In this part, we predict the hidden injured in the virtual disaster simulator called RoboCup Rescue Simulation (RCRS) with deep learning. First, we evaluate the several state-of-art CNN models to find the best performance of the CNN model. We train the ResNet [16], ResNeXt [18], Xception [19], Inception-ResNetV2 [32], InceptionV3 [33] and DenseNet [34] CNN based machine learning model to find the best performance of CNN model. According to the experiment, the machine learning model with Xception of RMSE is 2.1103 in the 4x4 grid which is the best performance. And the machine learning model with Inception-ResNetV2 of RMSE is 1.051 in the 8x8 grid which is the best performance. To enhance the model performance, we attach the attention module to the machine learning model in the next experiment. And for the efficient progress of the experiment, we choose three models to attach the attention module which perform well in both 4x4 and 8x8 grid based on the result of the experiment.

Second, we compare the performance of the attention module model with the non-attention module model to see whether the attention module improves the performance of the machine learning model. And we use the same scenario used in the previous experiment. We evaluate *Squeeze and excitation (SE)*, *Convolutional Block Attention Module (CBAM)* and *Grid Convolutional Block Attention Module (GCBAM)* attention module with InceptionResNetV2 and Xception CNN based model. According to the experiment, in the 4x4 grid size, there is an improvement in Inception-ResNetV2. However, there is no improvement in Xception. In the 8x8 grid size, there is little improvement in Inception-ResNetV2 and still no improvement in Xception. Furthermore, the Xception still is the best performance in the 4x4 grid which RMSE is 2.1103. And the Inception-ResNetV2 with GCBAM attention module is the best performance in the 8x8 grid which RMSE is 1.0509. In this study, we consider why the attention module cannot increase the performance of the machine learning model, we guess that due to the characteristics of the attention module that is focused more on the feature of the image seem to do not fit into the model well. Therefore, we expect to improve the performance of the model by utilizing the Feature-Highlight data annotation method which is proposed in this study.



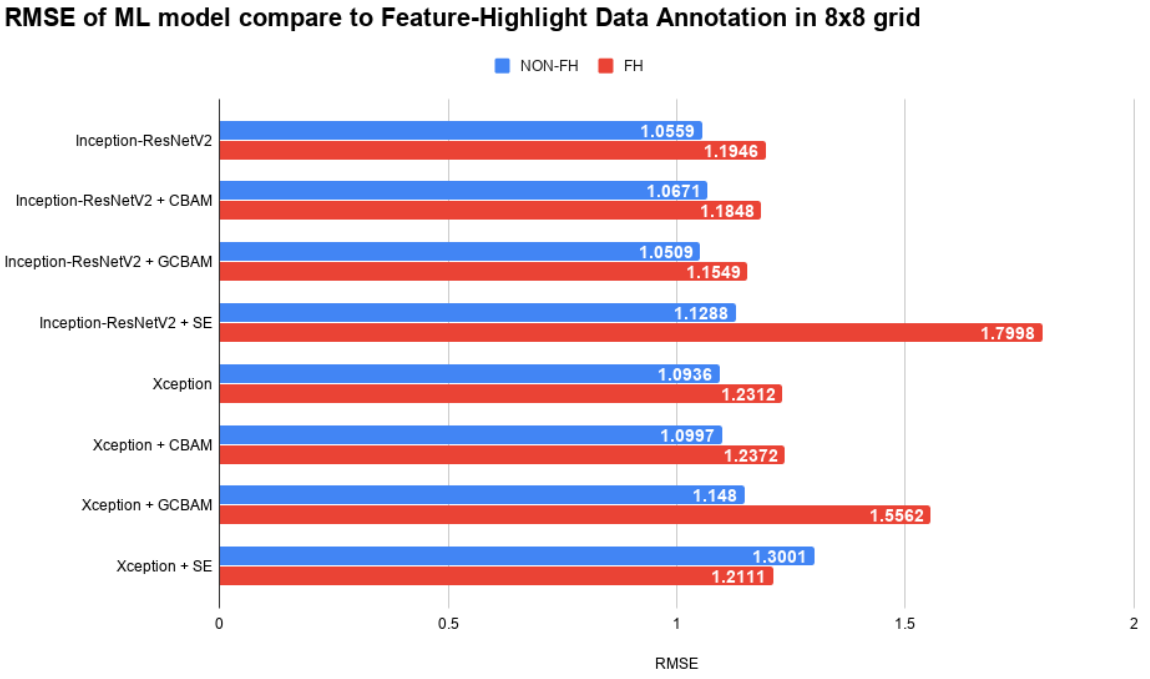


Figure 17: RMSE of ML model compare to FH and NON-FH

Third, we compare the performance of the Feature-Highlight data annotation and non-version. According to the experiment, there is improvement when we use Feature-Highlight data annotation with the attention module in the 4x4 grid except for the GCBAM attention module. However, in the 8x8 grid size, there is no improvement except for the Xception with the

SE attention module. Furthermore, the Inception-ResNetV2 with CBAM attention module and Feature-Highlight data annotation method shown the best performance in the 4x4 grid which RMSE is 2.013. And the Inception-ResNetV2 with GCBAM attention module and Feature-Highlight data annotation method shown the best performance in the 8x8 grid which RMSE is 1.051. And more than half of the machine learning model with FH data annotation shown better performance than the original model. It means our FH data annotation method performed well in the RCRS.

Last, we compare the other two domains called Chicago Crime Location Prediction and RSNA Pneumonia Detection Challenge. According to the experiment, in the Chicago Crime domain, in the grid size 4x4, Inception-ResNetV2 with the CBAM attention module is the best performance in which RMSE is 0.1498. In the grid size 8x8, Inception-ResNetV2 with our GCBAM attention module is the best performance in which RMSE is 0.1584. And the Inception-ResNetV2 with the CBAM attention module and the Inception-ResNetV2 with the GCBAM attention module is not only the best performance of the machine learning model in the Chicago Crime domain but also in the RCRS. However, in the RSNA domain, the ResNet-101 CNN based model is the best performance of the machine learning model in the 4x4 grid which RMSE is 0.1253. And the Xception with GCBAM attention module CNN based model is the best performance of the machine learning model in the 8x8 grid which RMSE is 0.1504. Because of the RSNA domain of characteristic, RSNA Pneumonia is not a perfectly hidden object and our machine learning model used the find the hidden object of goals. Therefore, the RSNA Pneumonia domain is more focus on detection so that the result is different.

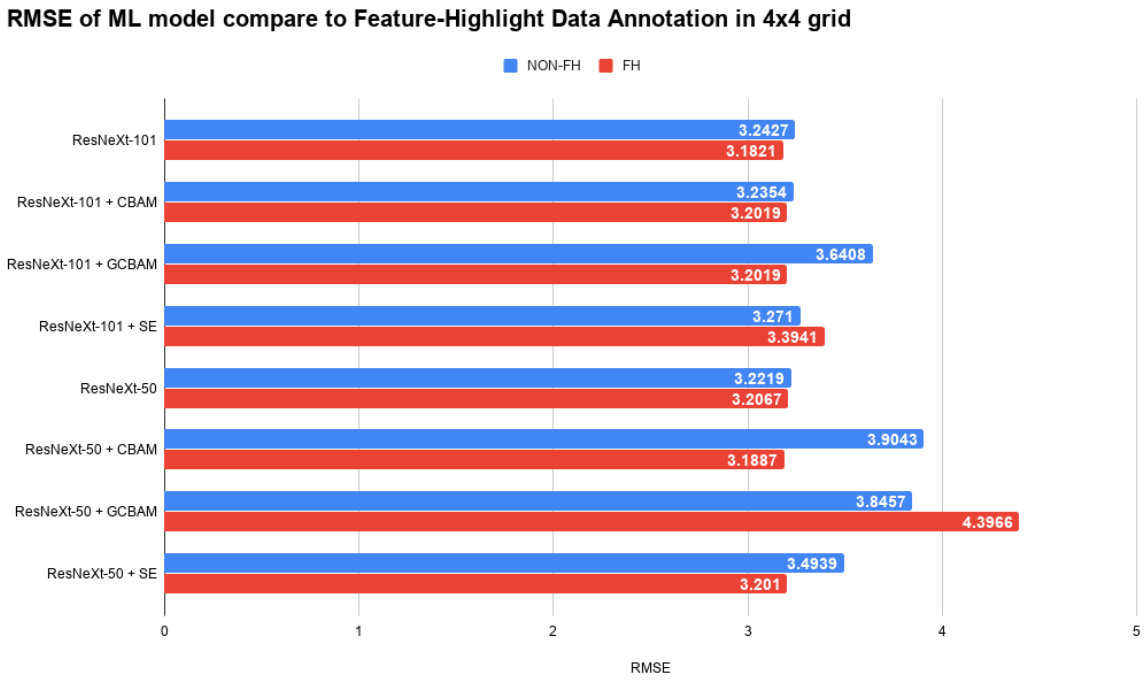
The best performance of ML model		
Domain	4x4	8x8
RCRS	Inception-ResNetV2 with CBAM	Inception-ResNetV2 with GCBAM
Chicago Crime	Inception-ResNetV2 with CBAM	Inception-ResNetV2 with GCBAM
RSNA Pneumonia	ResNet-101	Xception with GCBAM

Figure 18: The best performance of ML model in each domains

In the video part, we predict the density of injured people in each grid cell within the sequence of the screenshot image. First, we evaluate the several state-of-art CNN models with LSTM layers to find the best performance of the CNN model. We train the ResNet [16], ResNeXt [18], Xception [19], Inception-ResNetV2 [32], InceptionV3 [33] and DenseNet [34] CNN with LSTM layers based machine learning model to find the best performance of CNN model. According to the experiment, the ResNeXt-50 CNN based machine learning model is the best

performance in the 4x4 grid which RMSE is 3.2219. And also the ResNeXt-50 CNN based machine learning model is the best performance in the 8x8 grid which RMSE is 1.4829. To enhance the model performance, we attach the attention module to the machine learning model in the next experiment. And for the efficient progress of the experiment, we choose three models to attach the attention module which perform well in both 4x4 and 8x8 grid based on the result of the experiment. In this study, we choose the machine learning model with ResNeXt-50 and ResNeXt-101.

Second, we compare the performance of the attention module model with the non-version model to see whether the attention module improves the performance of the machine learning model. And we use the same scenario used in the previous experiment. We evaluate SE, CBAM and GCBAM attention module. According to the experiment, in the 4x4 grid size, the ResNeXt-101 has improvement with CBAM and in the 8x8 grid size, the ResNeXt-50 has improvement with GCBAM. And the ResNeXt-50 still the best performance in the 4x4 grid, the ResNeXt-50 with GCBAM is the best performance in the 8x8 grid which RMSE is 1.4896. We guess that due to the characteristics of the attention module that is focused more on the feature of the image seems to do not fit into model well. Therefore, we expect to improve the performance of the model by utilizing the Feature-Highlight data annotation method which is proposed in this study.



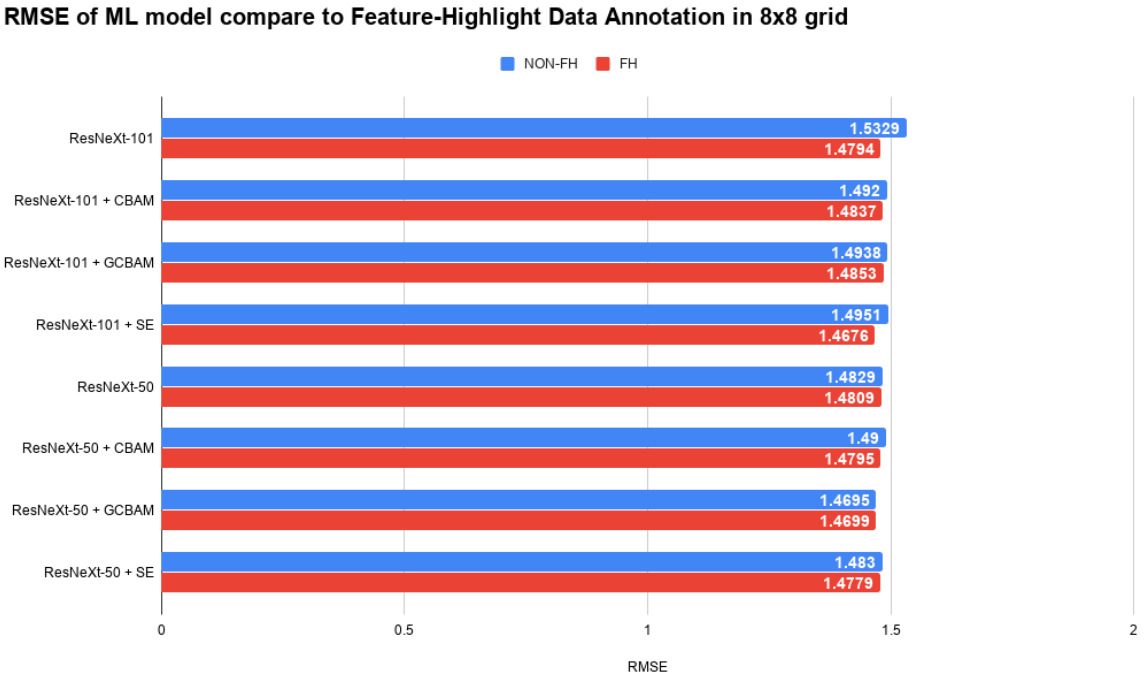


Figure 19: RMSE of ML model compare to FH and NON-FH

Last, We compare the performance of the Feature-Highlight data annotation with the non-Feature-Highlight Data annotation model. According to the experiment, in the 4x4 grid size, there is improvement when we use Feature-Highlight data annotation with the attention module except for the ResNeXt-101 with the SE attention module and ResNeXt-50 with GCBAM attention module. In the 8x8 grid size, there is a little improvement of all models when we use Feature-Highlight data annotation with the attention module. Furthermore, the ResNeXt-101 with Feature-Highlight data annotation method shown the best performance in the 4x4 grid which RMSE is 3.1821. And the ResNeXt with SE attention module and Feature-Highlight data annotation method shown the best performance in the 8x8 grid which RMSE is 1.4876. Furthermore, more than half of the machine learning model with FH data annotation shown better performance than the original model. It means our FH data annotation method performed well in the RCRS.

V Using ML Models in A* Search

Disasters such as earthquakes and tsunami can cause significant destruction to a city and hurt many people. To reduce the amount of the dead troll, fast disaster response to rescue survivors in a disaster zone is of paramount importance. But the problem is all the current method to manage disaster environment is all done by human and their work burden is too much to save the people as much as possible. Especially, find the location of people who need rescue in the disaster zone spend a considerable amount of time, which is one of the most important issues in disaster relief. However, if we can predict the location of injured people in a disaster situation, the time to save people can significantly reduce and this will a great positive effect on disaster relief. In this thesis, we propose the Treasure Hunt Problem to study the search strategy using in the disaster situation. In the disaster situation, the machine learning model has to predict the location of more than one civilian who needs rescue. But this is a complicated multi-agent problem. Therefore, we study a simpler problem called the Treasure Hunt Problem, in which there is only one hidden treasure. In this problem, the treasure is like an injured civilian in the disaster zone, but there is only one treasure in this problem

5.1 Treasure Hunt Problem

The treasure hunt problem is the problem that the hunter tries to find the treasure. In this problem, the treasure is locating in one cell, and the hunter is initially locating in another cell in the discretized map into a grid of a given size. The goal is this problem, the hunter should find the treasure as quickly as possible. And we divided this problem into two cases: (1) when the hunter moves the cell to find the treasure, there is no cost, (2) when the cell moves, there is a certain cost.

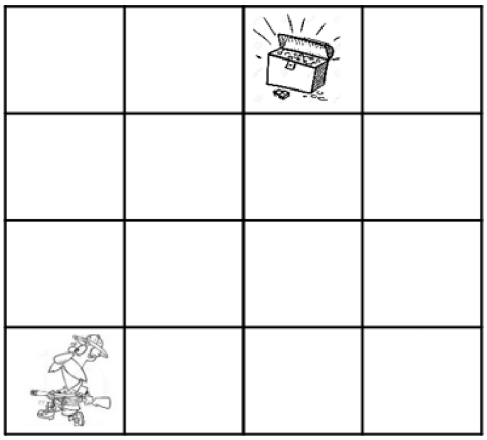


Figure 20: Example of hunter locate (1,1) and treasure locate (3,4) in 4x4 grid map

Here we assume that the hunter has knowledge for the first time. The hunter has a machine

learning model that predicts the probability of distribution of the treasure in a grid. However, the machine learning model is not perfect, and it predicts the probability distribution of the treasure in a grid whose cell size is larger than the actual grid's cell size. The goal is to find the treasure as quickly as possible, using both (1) machine learning to predict the location of the treasure and (2) a pathfinding algorithm that utilizes the machine learning model to find the treasure. To solve this problem, we build the treasure hunt problem simulator to simulate the hunter find the treasure using a machine learning model with a search strategy in the discrete space. To solve the Treasure Hunt Problem, we develop the simulator to simulate the hunter try to find the treasure location. In the simulator, the hunter is located to the (1,1) coordinates on a discrete map divided into grids and the treasure location (x_t, y_t) is randomly chosen according to a probability distribution $\boldsymbol{P}$. The probability distribution $\boldsymbol{P}$ is the probability of locate treasure in each cell $f(x, y)$ where the x and y are the coordinates for each cell in a discrete map divided into grids. And we assume that the map divided by the grid size of $M_P \times M_P$, where the $1 \leq x \leq M_P$ and $1 \leq y \leq M_P$.

5.2 The Machine Learning Model

In the simulator, the hunter tries to find the treasure using probability of existing treasure derived from machine learning (ML) model. And the ML model generator generates ML models for the agent using a probability distribution $\boldsymbol{Q}$. The probability distribution $\boldsymbol{Q}$ is the probability of existing treasure that ML model predict in each cell $f'(x, y)$ where the x and y are the coordinates for each cell in a discrete map divided into grids. Furthermore, the different between the probability distribution $\boldsymbol{P}$ and $\boldsymbol{Q}$ that we assume that the machine learning model is not perfect, and it predicts the probability distribution of the treasure in a grid whose cell size is larger than the actual grid's cell size. It means in the probability distribution $\boldsymbol{Q}$, the map divided by the grid size of $M_Q \times M_Q$, where the $1 \leq x \leq M_Q$, $1 \leq y \leq M_Q$ and $M_Q \leq M_P$. To evaluate and compare machine learning models with different prediction accuracy, we use Kullback–Leibler divergence to measure the distance between the probability distribution $\boldsymbol{P}$ for putting the treasure in a map and the probability distribution $\boldsymbol{Q}$ given to the agent as a ML model:

$$D_{KL}(P \parallel Q) = \sum_{i=1}^{M} \sum_{j=1}^{N} P(i,j) \, log\left(\frac{P(i,j)}{Q(i,j)}\right) \tag{4}$$

where i and j is the coordination of discrete space, M and N is the maximum of X and Y-axis of discrete space. In this equation, the $P(i, j)$ is the probability of locate treasure derived from probability distribution $\boldsymbol{P}$ and $Q(i, j)$ is probability of existing treasure derived from probability distribution $\boldsymbol{Q}$. Therefore, when the $D_{KL}(P \parallel Q) = 0$, the probability distribution $\boldsymbol{P}$ is equal to $\boldsymbol{Q}$ and we said $\boldsymbol{Q}$ is *truthful*.

5.3 The Search Strategy

In the treasure hunt problem, the hunter tries to find the treasure using probability with the search strategy. And we recall our assumption, the hunter has a machine learning model that predicts the probability of distribution of the treasure in a grid. With this probability distribution, the hunter uses two search strategies: (1) *The probabilistic greedy search* that the hunter searches preferentially for the cell with the highest probability of existing treasure given by ML model. In this search strategy, we focus on the first case problem (1) that the hunter can move from one cell to any other cell without the cost. This means the agent can visit one cell that is not adjacent to the current cell instantly. However, the agent still needs to spend the time to explore a cell. (2) *The probabilistically admissible heuristic A* search* that the hunter searches the cell determined by heuristic A* algorithm with the probability of existing treasure given by the ML model. In this search strategy, the heuristic value of A* search is determined by the method proposed in this thesis.

5.3.1 The Probabilistically Admissible Heuristic A* Search

In the A* search, if an admissible heuristic is adapted in an A* search algorithm, then this algorithm would eventually find an optimal solution to the goal. And the search strategy to become admissible heuristic, the estimated cost $h(n)$ must be lower than or equal to the actual cost (i.e., optimal cost) $h^*(n)$ of reaching the goal state $h(n) \leq h^*(n)$. In the treasure hunt problem, the ML model gives the probability of existing treasure (i.e., goal state) $p_{x,y}$ to each cell. According to this probability, we can probabilistically measure the actual cost of reaching the treasure $h^*(n|P = p_{x,y})$. For example, in the Figure **??**, if the cost of the moving between cells is 1, the probability of having a treasure in (3,3) is 30%, so the actual cost of reaching the treasure is 4 in 30%. And the probability of having a treasure in (3,2) and (2,3) is 40%, so the actual cost of reaching the treasure is 3 in 40%.

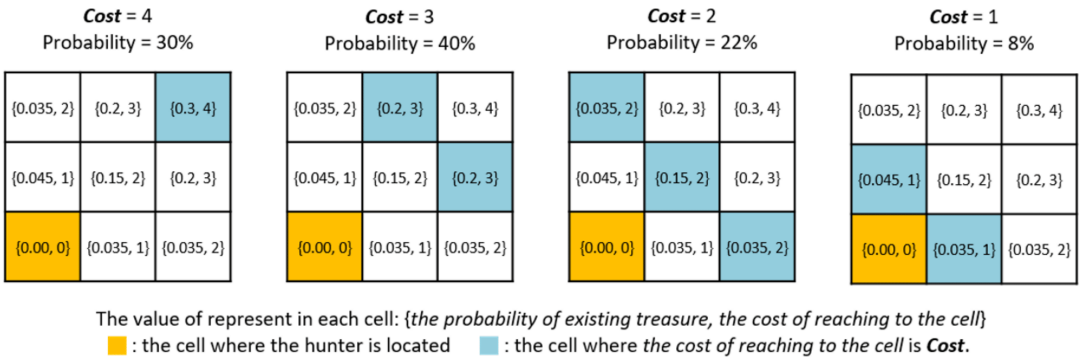


Figure 21: Example of the probabilistically measure the actual cost of reaching the treasure

And the heuristic value $\boldsymbol{h}$ is the estimated cost of reaching the treasure and if the heuristic

value is lower than or equal to the actual cost of reaching the treasure, the search strategy with heuristic value $\boldsymbol{h}$ is admissible. Therefore, the actual cost of reaching the treasure can probabilistically measure, we can say that the estimated cost is probabilistically to be less than or equal to the actual cost of reaching the treasure $h(n) \leq h^*(n|P = p_{x,y})$. It means that the search strategy is probabilistically admissible according to the determined heuristic value $\boldsymbol{h}$. To determine the heuristic value $\boldsymbol{h}$, we defined the threshold probability value called λ that is a value that guarantees the minimum probability of admissible of the search strategy. Therefore, the hunter searches the treasure with the *Probabilistically Admissible Heuristic A** search strategy in this below following order:

1. Define the vector $\boldsymbol{C}$ which consist of $C_{x,y} = \{p_{x,y}, D_{x,y}\}$, where x, y is the coordinate of the map divided by grid, $p_{x,y}$ is the probability of existing treasure $f'(x, y)$ and $D_{x,y}$ is the Manhattan distance of coordinate (x, y) to the hunter location.

2. Insert the $C_{x,y}$ which has the highest $p_{x,y}$ of the vector $\boldsymbol{C}$ into the queue $\boldsymbol{Q}$ and delete the $C_{x,y}$ in the vector $\boldsymbol{C}$ until it is larger than the threshold probability value λ .

3. Select the lowest $D_{x,y}$ in queue $\boldsymbol{Q}$ as the heuristic value $\boldsymbol{h}$ and the hunter start to A* search according to the heuristic value $\boldsymbol{h}$ that derived from the queue.

4. After search, set the $p_{x,y}$ of the searched cell to zero, and normalizes $p_{x,y}$ of the other remaining cells.

5. Go back to the (1) and repeat the search process until find the treasure.

5.4 Assumption

In this study, there is two assumptions for the Treasure Hunt Problem to make a more realistic problem. First, we assume that the machine learning model is not perfect, and it predicts the probability distribution of the treasure in a grid whose cell size is larger than the actual grid's cell size. Therefore, the ML model generator generates ML models for the agent using a probability distribution $\boldsymbol{Q}$, and the map divided by the grid size of $M_Q \times M_Q$, where the $1 \leq x \leq M_Q$, $1 \leq y \leq M_Q$. However, the machine learning model is not perfect, so that M_Q can lower than M_P. If the $M_Q \leq M_P$, we divide the each grid cell of probability in $\boldsymbol{Q}$ to calculate $D_{KL}(P \parallel Q)$. It means the probability of each grid cell in $\boldsymbol{Q}$ to $\frac{Q(i,j)}{(M_P/M_Q)^2}$. For example, if the treasure locate in 64x64 grid cell and the machine learning model predict the treasure location probability in 32x32 grid cell, the probability of each grid cell in $\boldsymbol{Q}$ which $Q(i, j)$ to $\frac{Q(i,j)}{4}$. Second, we assume that the agent can move from one cell to another cell in no time when we use *The probabilistic greedy search* as the search strategy. However, the agent still needs to spend time to explore a cell. And *The probabilistically admissible heuristic A* search* strategy relaxes the second assumption.

5.5 Experiment Results

In the second part, we propose the Treasure Hunt Problem. In RCRS, the rescue team has to search more than one injured people and it is a complicated multi-agent problem. Therefore, study a simpler problem called the Treasure Hunt Problem, in which there is only one rescue crew search the only one injured civilian. In this problem, we assume that the location of the treasure is determined based on the probability distribution, and the ML model predicts the distribution of probability that treasure exists for each coordinate within the map. To solve this problem, we propose two search strategies that makes use of the ML model to improve the effectiveness of a search mission: (1) *the probabilistic greedy search* that the hunter searches preferentially for the cell with the highest probability of existing treasure given by ML model; and (2) *the probabilistically admissible heuristic A* search* that the hunter searches the cell determined by heuristic A* search with the probability of existing treasure given by ML model.

5.5.1 The Probability Distribution

In this study, the hunter is located to the (1,1) coordinates on a discrete map divided into grids and the treasure location (x_t, y_t) is randomly chosen according to a probability distribution $\boldsymbol{P}$. The probability distribution $\boldsymbol{P}$ is the probability of locate treasure in each cell $f(x, y)$ where the x and y are the coordinates for each cell in a discrete map divided into grids. In this experiment, we assume that probability distribution $\boldsymbol{P}$ as the Multivariate Gaussian distribution. And the $f(x, y)$ follow below the Multivariate Gaussian distribution with the parameter μ and Σ.

$$f(x,y) = \frac{1}{2\pi\sigma_X\sigma_Y\sqrt{1-\rho^2}} \exp\left(-\frac{1}{2(1-\rho^2)}\left[\frac{(x-\mu_X)^2}{\sigma_X^2} + \frac{(y-\mu_Y)^2}{\sigma_Y^2} - \frac{2\rho(x-\mu_X)(y-\mu_Y)}{\sigma_X\sigma_Y}\right]\right) \quad (5)$$

where ρ is the correlation between x and y with $\sigma_X > 0$ and $\sigma_Y > 0$. And the mean and covariance matrix, where $\mu_X > 0$ and $\mu_Y > 0$ should positive integer

$$\mu = \begin{pmatrix} \mu_X \\ \mu_Y \end{pmatrix}, \quad \Sigma = \begin{pmatrix} \sigma_X^2 & \rho\sigma_X\sigma_Y \\ \rho\sigma_X\sigma_Y & \sigma_Y^2 \end{pmatrix} \quad (6)$$

The parameters vector of Multivariate Gaussian distribution μ and Σ chosen different depending on the experiment. And normalize the value at the center of a cell across the grid, such that the sum of all values in all cells is 1. And the probability distribution $\boldsymbol{Q}$ follow the Multivariate Gaussian distribution as well with the different parameter μ' and Σ'. The parameters vector μ' and Σ' chosen also different depending on the experiment. And normalize the value at the center of a cell across the grid, such that the sum of all values in all cells is 1.

5.5.2 The Probabilistic Greedy Search

In this experiment, we evaluate the search time according to the different grid sizes and the machine learning model accuracy with *The Probabilistic Greedy Search Strategy*. To evaluate our search strategy, We simulate the hunter find the treasure in 64x64 grid map, and the treasure location randomly chosen according to a multivariate Gaussian distribution $\boldsymbol{P}$ with the parameter is $\mu = \begin{pmatrix} 50 \\ 50 \end{pmatrix}$, $\Sigma = \begin{pmatrix} 8 & 0 \\ 0 & 0 \end{pmatrix}$. And the hunter has a machine learning model to predict the probability of distribution of the treasure in a grid. The hunter's ML model is a multivariate Gaussian distribution $\boldsymbol{Q}$ as well with the parameter that makes the $D_{KL}\left(P \parallel Q\right)$of probability distribution $\boldsymbol{P}$ and $\boldsymbol{Q}$ to be 1, 4, 8, 16, 24, 32, 40, 48, 64 and 96. The hunter searches preferentially for the cell with the highest probability of existing treasure given by the ML model. In this experiment, we evaluate the search time according to the grid size of the different ML models. In the graph of Figure. 22, the search time which is the value of the y-axis is the number of cells that the hunter has searched until a treasure is found multiplied to the time to search one cell. And we assume that the time to search one cell is $\frac{1}{Gridsize}$. We can say that the smaller the search time, the higher the performance of the model. We have tested the grid size to 1x1, 4x4, 8x8, 16x16, 32x32 and 64x64. For each grid size, we took the average of the search time of the 10,000 times of simulation and plotted a graph within 1x1 to 64x64 with different colors according to the different $D_{KL}\left(P \parallel Q\right)$of ML models.

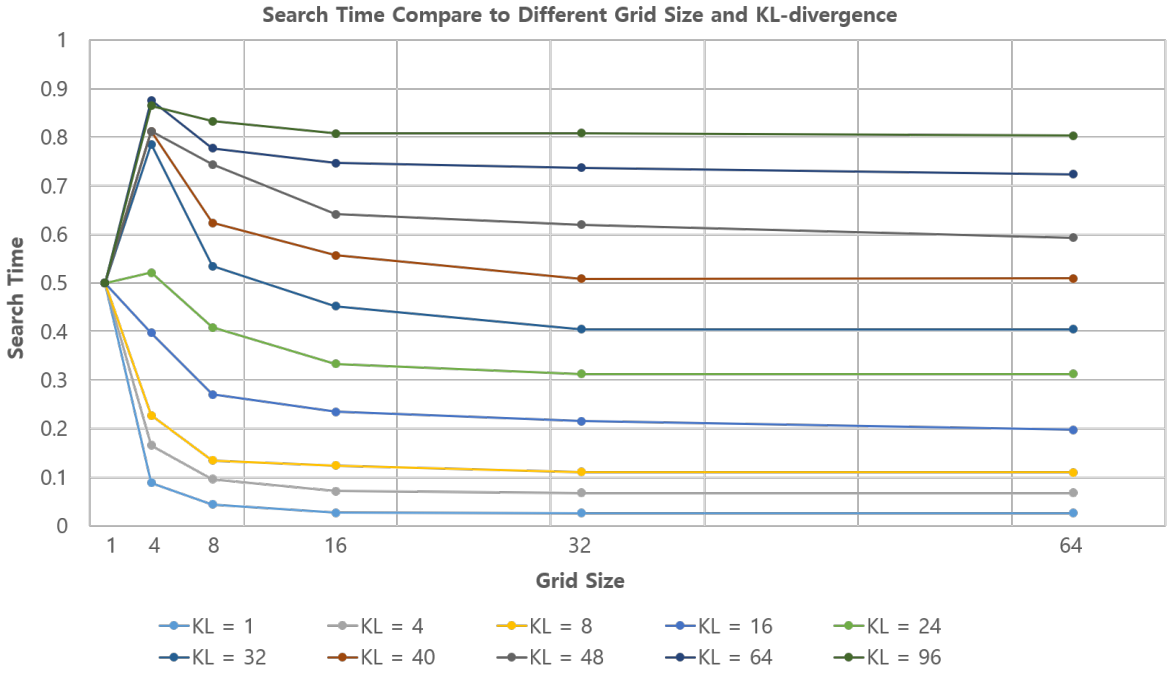


Figure 22: Search time according to $D_{KL}\left(P \parallel Q\right)$ and grid size of the probabilistic greedy search

The result shows that where the larger $D_{KL}\left(P \parallel Q\right)$, the average search time larger. In common sense, if the machine learning model accuracy lower, the search time going to lower too. And

the result also followed the same results as common sense. And recall our assumption, *the probabilistic greedy search* jumps the cell without cost so that it is hard to observe the efficiency of grid size and accuracy. Therefore, we test the machine learning model with *the probabilistically admissible heuristic A* search*

5.5.3 The Probabilistically Admissible Heuristic A* Search: Find the Optimal λ

In this experiment, we find the optimal parameter of *the probabilistically admissible heuristic A* search* to evaluate the search time of this search strategy. The probabilistically admissible heuristic A* search has the parameter called the λ which is how much admissible this search strategy will allow. Therefore, as the first step of the experiment, We simulate the hunter find the treasure in 32x32 grid map, and the treasure location randomly is chosen according to a multivariate Gaussian distribution $\boldsymbol{P}$ with the parameter is $\mu = \begin{pmatrix} 25 \\ 25 \end{pmatrix}$, $\Sigma = \begin{pmatrix} 8 & 0 \\ 0 & 8 \end{pmatrix}$. And The hunter's ML model is a multivariate Gaussian distribution $\boldsymbol{Q}$ as well with the parameter that makes the $D_{KL}(P \parallel Q)$of probability distribution $\boldsymbol{P}$ and $\boldsymbol{Q}$ to be 2, 4, 6, 8, 16, and 32. The hunter searches *the probabilistically admissible heuristic A* search* with the probability of existing treasure given by the ML model. We evaluate the search time according to the threshold probability value λ to 10%, 20%, 30% ...100% and the different ML models. And as the first step of the experiment, we test only 16x16 grid size of the machine learning prediction $\boldsymbol{Q}$. In this experiment, the search time is the number of cells that the agent has searched until a treasure is found, multiplied to the time to search one cell. And we assume that the time to search one cell is $\frac{1}{Gridsize}$.

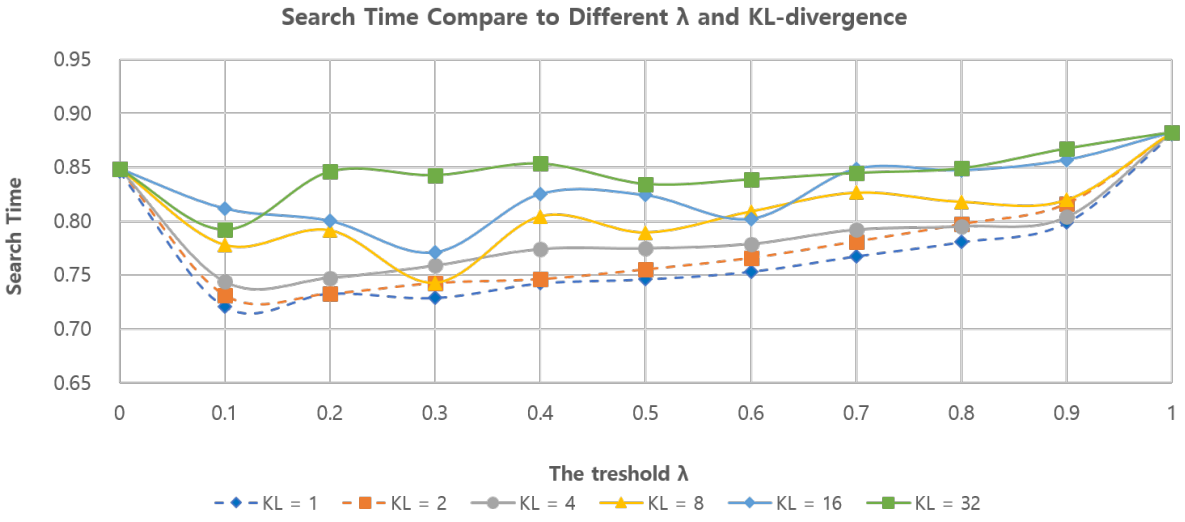


Figure 23: Search time according to $D_{KL}(P \parallel Q)$ and λ

The result shows that the existing optimal λ that minimum time to search the treasure ac-

cording to the model accuracy $D_{KL}(P \parallel Q)$. For example, when the machine learning model of $D_{KL}(P \parallel Q)$is 1, 2, 4 and 32, there is one global optimal λ value which is 10%. The result shows that after the search time is decreased from 0% to 10% and then increases after that. And when the machine learning model of $D_{KL}(P \parallel Q)$is 8 and 16, there is one global optimal λ value which is 30%.

Table 1: The optimal λ for each grid size compare to different $D_{KL}(P \parallel Q)$

	Grid Size					
$D_{KL}(P \parallel Q)$	1 x 1	2 x 2	4 x 4	8 x 8	16 x 16	32 x 32
1	10%	10%	10%	10%	10%	10%
2	10%	10%	10%	10%	10%	10%
4	10%	0%	10%	10%	10%	10%
8	10%	0%	20%	10%	30%	10%
16	10%	10%	10%	10%	30%	30%
32	10%	10%	70%	10%	10%	30%

Furthermore, not only to 32x32 grid sizes but also to other grid sizes shown the similar graph patterns which represent there is optimal λ exist according to the experiment. This shows that the optimal value of threshold λ, a parameter for *The Probabilistically Admissible Heuristic A* Search* exists. Therefore, we found the optimal λ for each grid size compare to different $D_{KL}(P \parallel Q)$. And the table 1 shows that the different optimal λ according to the different accuracy of the machine learning model.

5.5.4 The Probabilistically Admissible Heuristic A* Search: Find the Optimal Grid Size

In this experiment, we evaluate the search time according to the different grid size with optimal λ. We simulate the hunter find the treasure in a 32x32 grid map, and the treasure location randomly is chosen according to a multivariate Gaussian distribution $\boldsymbol{P}$ and $\boldsymbol{Q}$ with the parameter is same as the previous experiment. We evaluate the search time according to the grid size of the different ML models. And the λ which is the parameter of *the probabilistically admissible heuristic A* search* to set the optimal λ. The search time which is the value of the y-axis is the number of cells that the hunter has searched until a treasure is found multiplied to the time to search one cell. We have tested the grid size to 1x1, 2x2, 4x4, 8x8, 16x16 and 32x32 of the machine learning prediction $\boldsymbol{Q}$. For each grid size, we took the average of the search time of the 10,000 times of simulation with the different λ and plotted a graph within 1x1 to 64x64 with different colors according to the different $D_{KL}(P \parallel Q)$of ML models.

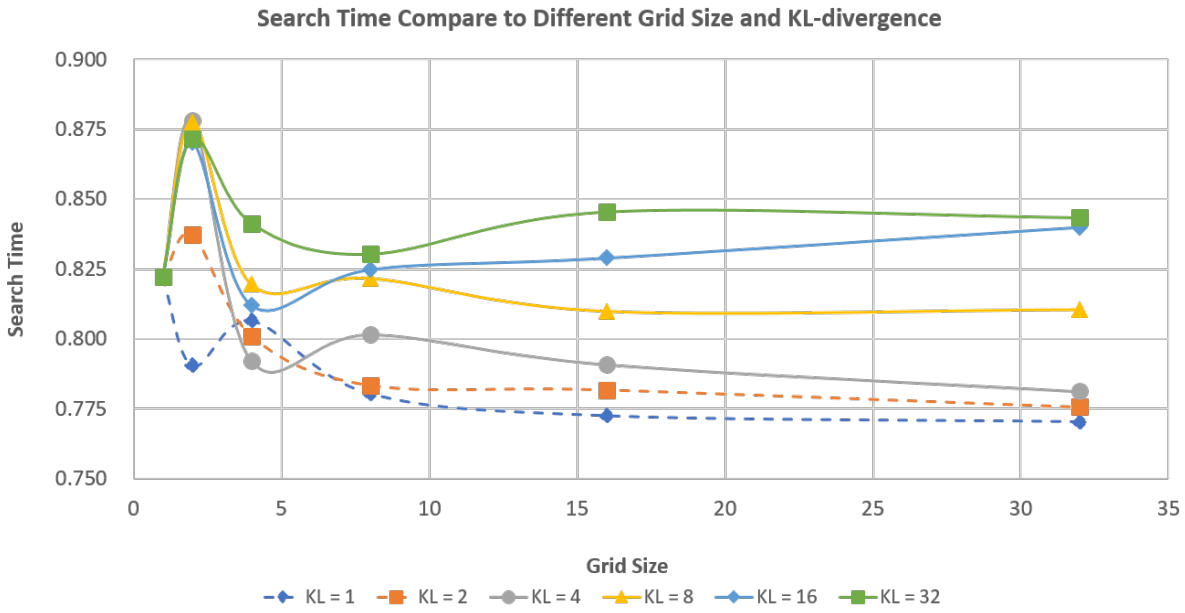


Figure 24: Search time according to $D_{KL}(P \parallel Q)$ and grid size

The result shows that the grid size which shows the minimum search time is different depending on the accuracy of the model. On average, the higher the accuracy of the machine learning model on all grid sizes, the shorter the search time. When the KL (i.e., $D_{KL}(P \parallel Q)$) is 32, it means the low accuracy of the machine learning model, the average search time is 0.845. And when the KL is 1, it means the high accuracy of the machine learning model, the average search time is 0.775. However, each machine learning model with different accuracy shows that there is a certain grid size that shows the global or local minimum search time.

5.6 Conclusions

In this part, we study the Treasure Hunt Problem with the search strategy that *the probabilistic greedy search* and *the probabilistically admissible heuristic A* search*. The probability distribution used in the locating of treasure and the machine learning model is Multivariate Gaussian distribution, and we test the different parameters of this distribution. In this part, we study the relationship of the accuracy of the machine learning model and grid size with the Treasure Hunt problem. The grid size represents the precision of prediction and the KL-divergence of the probability distribution of the machine learning model and locating the treasure represent the accuracy of prediction. Therefore, we test the two search strategies with different KL-divergence and grid size. The results in the two search strategies showed two different aspects: *the probabilistic greedy search* and *the probabilistically admissible heuristic A* search*.

The probabilistic greedy search of result shows that where the larger $D_{KL}(P \parallel Q)$, the average of search time larger. In common sense, if the machine learning model accuracy lower, the search time going to lower too. And the result also followed the same results as common sense. However, after the grid size 32x32, the search time was only a small change. And when the

machine learning model of accuracy (i.e., KL-divergence of ML model) is too low or too high, there is a little variation in search time with grid size. It means that after some point of grid size (i.e., the precision of the prediction), increasing the size of the grid is less efficient depending on the accuracy of the machine learning model. And recall our assumption, *the probabilistic greedy search* jumps the cell without cost so that it is hard to observe the efficiency of grid size and accuracy. Therefore, we test the machine learning model with *the probabilistically admissible heuristic A* search.*

The probabilistically admissible heuristic A search* of result shows that the grid size which shows the minimum search time is different depending on the accuracy of the model. On average, the higher the accuracy of the machine learning model on all grid sizes, the shorter the search time. When the KL (i.e., $D_{KL}(P \parallel Q)$) is 32, it means the low accuracy of the machine learning model, the average search time is 0.845. And when the KL is 1, it means the high accuracy of the machine learning model, the average search time is 0.775. However, each machine learning model with different accuracy shows that there is a certain grid size that shows the global or local minimum search time. For example, the machine learning model in which KL is 16, after the 4x4 grid size, the search time is longer when the grid size larger. And the machine learning model in which KL is 32, after the 8x8 grid size, the search time is longer when the grid size larger. Otherwise, the machine learning model in which KL is lower than 8, there is a local point of grid size which is low search time but the shorter search time when the grid size larger. It means that when the KL (i.e., the accuracy of the model) is low, there is an optimal grid size (i.e., the precision of model) with minimum search time. And the KL is high, there is a local optimal grid size. However, the larger the grid size, the less search time. According to this experiment, when the search strategy using a machine learning model, there is optimal precision depends on the accuracy of the machine learning model. And as we expected, the optimal point is shown when the accuracy of the machine learning model is low. In the last part, we apply this study to the first part to optimize the precision in a deep learning model to locate the injured people in disaster zones.

VI Tuning Precision of Locations in Deep Learning Model for Locating Injured People

In the last part, we merge the first and second parts to search for the location of the most density injured people area. In the first part, we predicted the density of injured people in the disaster zone divided into the grid. In the second part, we proposed the Treasure Hunt Problem, in which there is only one rescue crew search the only one injured civilian. And the goal of the third part is to find the most density injured area based on the number of injured people which predicted in the first part and the search strategy proposed in the second part. Therefore, in the Treasure Hunt Problem, only one rescue crew (i.e., the rescue team) search the only one injured civilian (i.e., the most density injured area). And the search strategy of rescue crew is the *The probabilistic greedy search* and *The probabilistically admissible heuristic A* search* which based on the probability distribution of converted the injured people density predicted by the machine learning model.

6.1 Treasure Hunt with Hidden Injured Problem in RCRS

In the second part, the rescue team search injured people according to the specific search strategy with the machine learning model. And this machine learning model used in the search strategy is based on the multivariate Gaussian distribution. However, in this part, we replace the machine learning model used in the search strategy to the machine learning model studied in the first part. We convert the injured people density predicted by the machine learning model to the probability distribution. And the rescue team search the most density injured people area according to the search strategy of the second part based on this probability distribution. In this part, first, we predict the density of injured people in the RCRS using the first part of the machine learning model. As a first part of assumption, the map divided by the grid size of $M \times N$, and the number of injured civilians in each grid cell $N_{i,j}$ where the $i = 1, 2, \ldots, M$ and $j = 1, 2, \ldots, N$. Then we divide each predicted number of injured people $N_{i,j}$ to the total number of injured people T is $\sum_{i=1}^{M} \sum_{j=1}^{N} N_{i,j}$. Therefore, the probability of existing treasure at each grid cell $P_{i,j}$ is $\frac{N_{i,j}}{T}$.

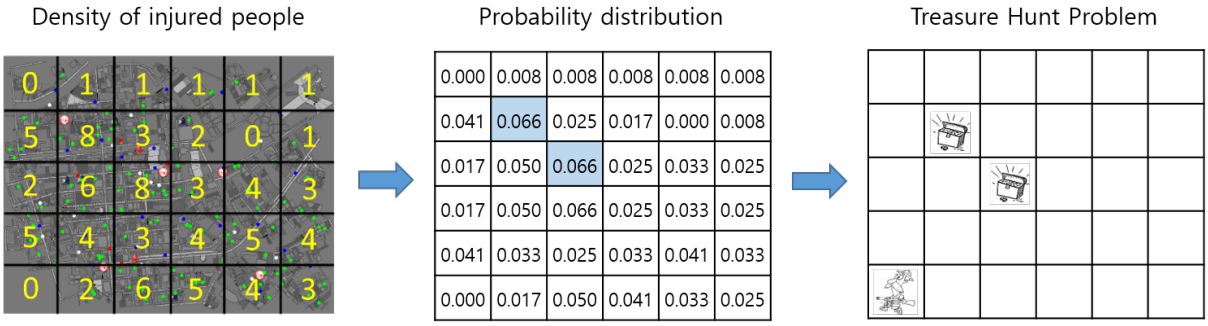


Figure 25: Treasure Hunt problem with RCRS

As the Treasure Hunt problem of the second part, the hunter is located to the (1,1) coordinates on a discrete map divided into grids and the treasure location (x_t, y_t) is the most density injured people grid cell location. And the hunter tries to find the treasure using the probability of existing treasure derived from the machine learning (ML) model. And the ML model generator generates ML models for the agent using a probability distribution $\boldsymbol{Q}$. The probability distribution $\boldsymbol{Q}$ is the probability of existing treasure that ML model predict in each cell $f'(x, y)$ where the x and y are the coordinates for each cell in a discrete map divided into grids. And in this part we set the probability of existing treasure that ML model predict in each cell $f'(x, y)$ to $P_{i,j}$. And the hunter uses two search strategies: (1) *The probabilistic greedy search* that the hunter searches preferentially for the cell with the highest probability of existing treasure given by the ML model. In this search strategy, we focus on the first case problem (1) that the hunter can move from one cell to any other cell without the cost. This means the agent can visit one cell that is not adjacent to the current cell instantly. However, the agent still needs to spend the time to explore a cell. (2) *The probabilistically admissible heuristic A* search* that the hunter searches the cell determined by heuristic A* algorithm with the probability of existing treasure given by the ML model.

6.2 The Machine Learning Model and The Search Strategy

To predict the density of injured people in RCRS using image-based data, we train the machine learning model using simulation screenshot images. We consider that all frames of video clips in disaster situations are independent images. Therefore, we choose one frame randomly in images sequence of a disaster scenario in RCRS to train the machine learning model. The model's inputs are simulated images created from RCRS. And CNN extracts images features, fully-connected layer output the number of injured people vectors in each grid cell. In this part, we use several machine learning models used in part one to predict the number of injured people. We used five different machine learning models for prediction based on the accuracy: (1) Inception-ResNet-V2 with CBAM attention module and Feature-Highlight data annotation; and (2) Xception with GCBAM attention module; and (3) DenseNet-169; and (4) DenseNet-201; and (5) ResNeXt-101.

Table 2: The accuracy of ML model compare to different grid size

	Grid Size					
ML model	1 x 1	2 x 2	4 x 4	8 x 8	16 x 16	32 x 32
Inception-ResNetV2+CBAM+FH	7.4211	5.2128	2.0130	1.0477	0.5595	0.2950
Xception+GCBAM	7.4371	4.7445	2.5771	1.0701	0.5756	0.2766
DenseNet-201	14.5679	4.4213	3.4664	1.3065	0.5945	0.3382
ResNeXt-101	45.6001	11.6522	3.9617	1.7411	0.8134	0.3827

As same as part one, we create a total of 11,000 disaster scenarios and we choose 10,000 scenarios to train the machine learning model, 1,000 scenarios to test the machine learning model. For each scenario, we divide the simulation map into a grid size 1x1, 2x2, 4x4, 8x8, 16x16 and 32x32. And we compared the actual density of injured civilians in each grid cell (i.e., the ground truth) with the one predicted by the machine learning model and calculated the root mean squared error (RMSE) between the ground truth and the predicted numbers. The RMSE of the machine learning model shown as table 2, Inception-ResNet-V2 with the CBAM attention module and Feature-Highlight data annotation model has the best performance and ResNeXt-101 model has the worst performance. And we predict the number of injured citizens in each grid cell with five models of different accuracy, and then we search the most density injured people grid cell using the two search strategy as same as the part two: *The probabilistic greedy search* and *The probabilistically admissible heuristic A* search.* This problem is the same as the Treasure Hunt problem, however, the hunter converts to the rescue team and the treasure convert to the most density injured people grid cell.

6.3 Experiment Results

In the last part, we merge the first and second parts to search for the location of the most density injured people area. To predict the location, we predict the number of injured people with several ML models used in the first part and we convert the injured people density predicted to the probability distribution. And the rescue team search the most density injured people area according to the search strategy of the second part based on this probability distribution.

6.3.1 The Probabilistic Greedy Search

In this experiment, we simulate the hunter to find the treasure in the 32x32 grid map, and the treasure location is the most density of injured people grid cell. And the hunter has a machine learning model to predict the probability distribution of the treasure in a grid. This probability distribution is converted from the injured people density predicted by the machine learning model. The hunter's ML model is the four different machine learning models for prediction based on the accuracy: (1) Inception-ResNet-V2 with CBAM attention module and Feature-Highlight

data annotation; and (2) Xception with GCBAM attention module; and (3) DenseNet-169; and (4) ResNeXt-101. The hunter searches preferentially for the cell with the highest probability of existing treasure given by the ML model. In this experiment, we evaluate the search time according to the grid size of the different ML models. In the graph of Figure. 26, the search time which is the value of the y-axis is the number of cells that the hunter has searched until a treasure is found multiplied to the time to search one cell. And we assume that the time to search one cell is $\frac{1}{Gridsize}$. We can say that the smaller the search time, the higher the performance of the model. We have tested the grid size to 1x1, 4x4, 8x8, 16x16 and 32x32. For each grid size, we took the average of the search time of the 10,000 times of simulation and plotted a graph within 1x1 to 32x32 with different colors according to the different ML models.

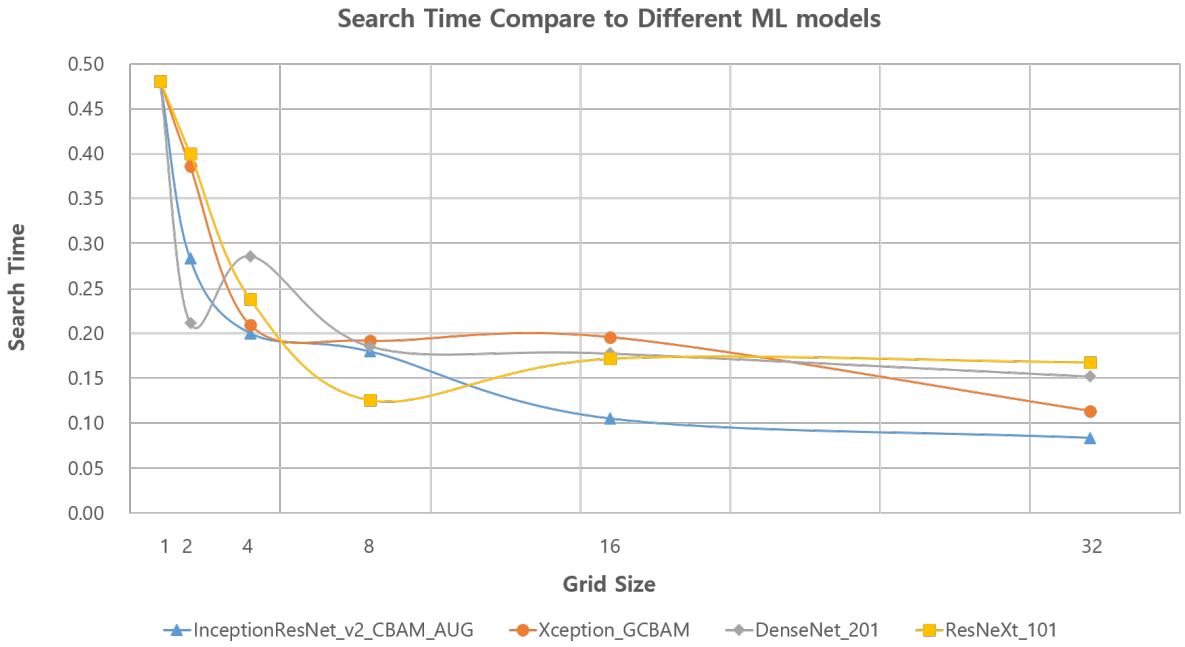


Figure 26: Search time according to the different ML model and grid size

6.3.2 The Probabilistically Admissible Heuristic A* Search: Find the Optimal λ

In this experiment, we evaluate the search time according to the different grid sizes and the machine learning model accuracy with *The Probabilistic Greedy Search Strategy*. And we simulate the hunter find the treasure in 32x32 grid map, and the treasure location is the most density of injured people grid cell. The hunter has a machine learning model to predict the probability distribution of the treasure in a grid. This probability distribution is converted from the injured people density predicted by the machine learning model. The hunter's ML model is the four different machine learning models for prediction based on the accuracy: (1) Inception-ResNet-V2 with CBAM attention module and Feature-Highlight data annotation; and (2) Xception with

GCBAM attention module; and (3) DenseNet-169; and (4) ResNeXt-101. The hunter searches *the probabilistically admissible heuristic A* search* with the probability of existing treasure given by the ML model. We evaluate the search time according to the threshold probability value λ to 10%, 20%, 30% ... 100% and the different ML models. In this experiment, the search time is the number of cells that the agent has searched until a treasure is found, multiplied to the time to search one cell. And we assume that the time to search one cell is $\frac{1}{Gridsize}$. We can say that the smaller the search time, the higher the performance of the model. First of all, We have tested the grid size to 32x32. For each grid size, we took the average of the search time of the 10,000 times of simulation and plotted a graph within λ is 10%, 20%, 30% ... 100% with different colors according to the different accuracy of ML models.

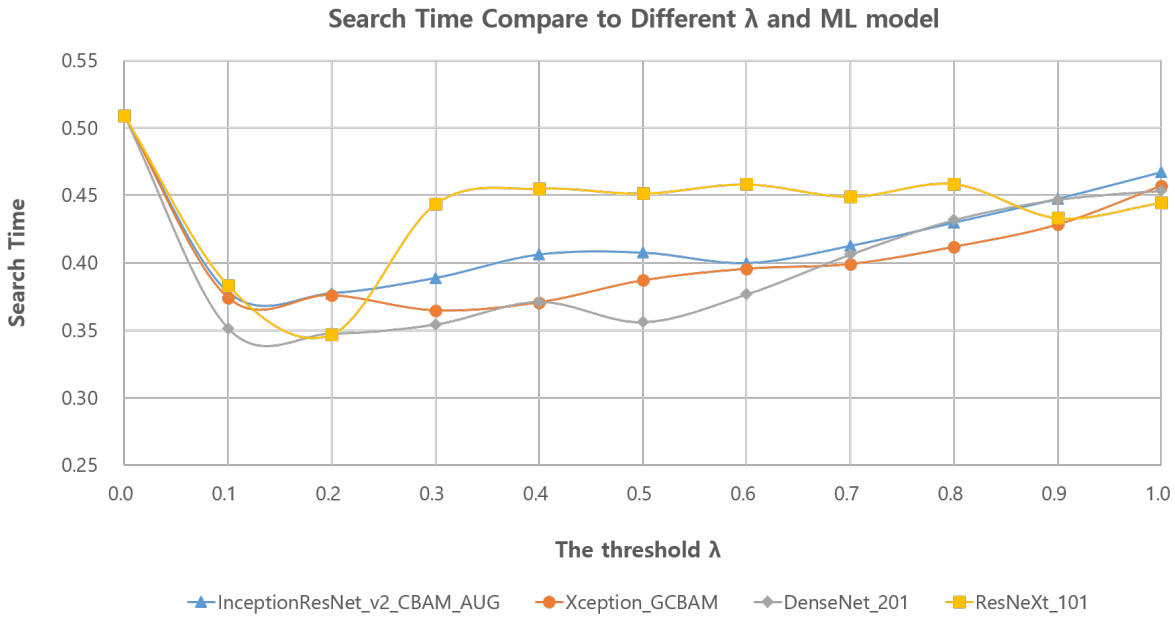


Figure 27: Search time according to $D_{KL}(P \parallel Q)$ and λ

The result shows that the existing optimal λ that minimum time to search the treasure according to the model accuracy $D_{KL}(P \parallel Q)$. And the optimal λ can be multiple or one. In this experiment, we plotted the four different ML model with the grid size to 8x8. Figure **??**, when the InceptionResNet-v2 with CBAM attention, model, the global optimal λ is 10% where λ decreases from 0% to 10% and then increases after that. And when the Xception with GCBAM attention model, the global optimal λ is 30% and the local optimal λ is 10%, 20%, and 40%. The other models of global optimal λ is 20%.

Table 3: The optimal λ for each grid size compare to different $D_{KL}(P \parallel Q)$

	Grid Size					
$D_{KL}(P \parallel Q)$	1 x 1	2 x 2	4 x 4	8 x 8	16 x 16	32 x 32
Inception-ResNetV2 + CBAM + FH	70%	80%	20%	20%	20%	40%
Xception + GCBAM	70%	80%	40%	30%	10%	50%
DenseNet-201	70%	10%	20%	20%	20%	50%
ResNeXt-101	70%	80%	10%	20%	20%	60%

Furthermore, not only to 16x16 grid sizes but also to other grid sizes shown the similar graph patterns which represent there is optimal *lambda* exist according to the experiment. This shows that the optimal value of threshold λ, a parameter for *The Probabilistically Admissible Heuristic A* Search* exists. Therefore, we found the optimal λ for each grid size compare to different ML models. And the table 3 shows that the different optimalaccording to the different accuracy of the machine learning model

6.3.3 The Probabilistically Admissible Heuristic A* Search: Find the Optimal Grid Size

In this experiment, we evaluate the search time according to the different grid size with optimal λ. We simulate the hunter find the treasure in the 32x32 grid map, and the treasure location is the most density of injured people grid cell. And the hunter has the machine learning model to predict the probability distribution of the treasure in a grid. This probability distribution is converted from the injured people density predicted by the machine learning model. The hunter's ML model is the four different machine learning models used in the previous experiment. The hunter searches *the probabilistically admissible heuristic A* search* with the probability of existing treasure given by the ML model. And We evaluate the search time according to the grid size of the different ML models. And the λ which is the parameter of *the probabilistically admissible heuristic A* search* to set the optimal λ. We have tested the grid size to 1x1, 2x2, 4x4, 8x8, 16x16 and 32x32. For each grid size, we took the average of the search time of the 10,000 times of simulation and plotted a graph within 1x1 to 32x32 with different colors according to the different ML models.

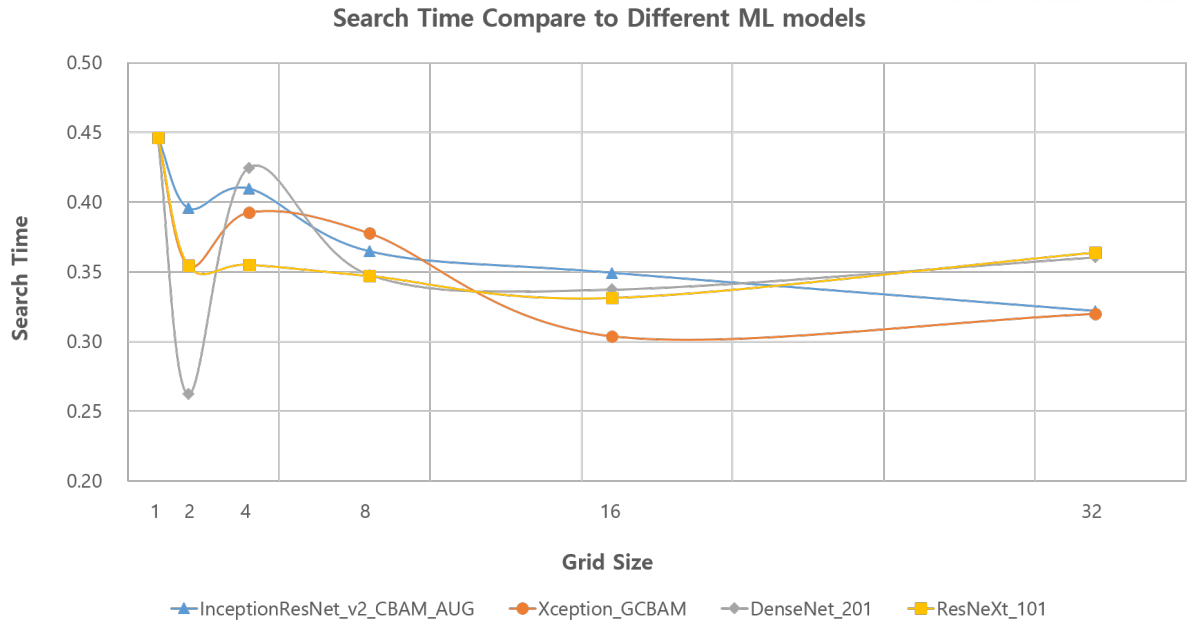


Figure 28: Search time according to $D_{KL}(P \parallel Q)$ and grid size

6.4 Conclusions

In this part, we study the optimizing the grid size in the deep learning model to predict. We test the search strategy that *the probabilistic greedy search* and *the probabilistically admissible heuristic A* search* with the machine learning model to predict the injured people. And optimize the grid size according to the accuracy of the machine learning model used in part one. In this part, we study the relationship of the accuracy of the machine learning model and grid size with the search strategy studied in the Treasure Hunt problem with the deep learning model to predict the injured people. The grid size represents the precision of prediction and the RMSE of the machine learning model represents the accuracy of prediction. Therefore, we test the two search strategies with the different ML model and grid size. The results in the two search strategies showed two different aspects: *the probabilistic greedy search* and *the probabilistically admissible heuristic A* search.*

The probabilistic greedy search with the machine learning model to predict the injured people of the result shows that when the accuracy of the machine learning model is low (i.e., DenseNet-201 and ResNeXt-101 model), there is an optimal grid size which is the minimum search time. And the accuracy of the machine learning model is high (i.e., InceptionResNet-v2 and Xception), when the grid size is larger, the search time is less. In the DenseNet-201 model, the grid size of 2x2 is the local optimal grid size that the search time is 0.21. And in the ResNeXt-101 model, the grid size of 8x8 is the global optimal grid size that the search time is 0.13. On the other hand, when the accuracy of machine learning is high, there is no optimal grid size and the larger grid size, the less search time. In the InceptionResNet-v2 with the CBAM attention module and

the Xception with the CBAM attention module, the search time keeps reduction when the grid size is larger. In part two, we confirm that when the accuracy of machine learning is low, there is an optimal grid size and in this part, We were able to see the same result. Moreover, using the machine learning model to predict the injured people used in part one, the same results are shown. And recall our assumption, *the probabilistic greedy search* jumps the cell without cost so that it is hard to observe the efficiency of grid size and accuracy. Therefore, we test the machine learning model used in part one with *the probabilistically admissible heuristic A* search*

The probabilistically admissible heuristic A search* of result (Figure.**??**) with the machine learning model to predict the injured people shows similar to the result of *The probabilistic greedy search*. When the InceptionResNet-v2 with the CBAM attention module model, the grid size is larger, the search time is less. One different from the result of *The probabilistic greedy search*, the Xception with the GCBAM attention module, shows the global optimal grid size 16x16. Because of the *The probabilistically admissible heuristic A* search*, it requires more accuracy to show the larger grid size, the less search time. And the DenseNet-201 and the ResNeXt-101, the global and local optimal gird size shown as 2x2 and 16x16. According to this experiment, when the search strategy using a machine learning model, there is optimal precision depends on the accuracy of the machine learning model. Furthermore, as we expected, the optimal point is shown when the accuracy of the machine learning model is low when we test the deep learning model to locate the injured people in disaster zones.

VII Summary and Future Work

In this study, we propose to develop an AI system to predict the location of injured people in a disaster area. In this research, our system has three major parts: (1) the prediction of the density of injured people in a grid; and (2) the strategy of the rescue team to search for injured people; and (3) the deployment the rescue team to search the location of the most density injured people area according to the first and second part. In the first part, we developed a deep learning software package that consists of state of the art deep learning technique such as attention module and data annotation to predict the density of injured civilians. Our work uses a disaster simulator called *RoboCup Rescue Simulation (RCRS)*. To predict the density of injured people in RCRS, we train the machine learning model using the two cases of the image data: (1) single image frame such as a satellite image; and (2) multiple image sequence frame such as disaster video clip. Furthermore, we evaluate our ML model in the other two domains: (1) the prediction of the location of crime in Chicago; and (2) the prediction of the location of RSNA Pneumonia. In this part, we find the best machine learning model in the RCRS problem compare to the different state-of-art deep learning models. And we attach the *Squeeze and excitation (SE)*, *Convolutional Block Attention Module (CBAM)* and *Grid Convolutional Block Attention Module (GCBAM)* attention module to the machine learning model to improve the performance. Furthermore, we propose the *Feature-Highlight Data annotation* method that attention module can more focus on the feature of images. And we compare the machine learning model with our Feature-Highlight data annotation method and non-version, we confirm that our Feature-Highlight data annotation improves the performance of the machine learning model with the attention module. And we evaluate our best performance of the machine learning model to other domains and confirm that the best performance of the machine learning model also well performs on the other domains which deal with the same problem as RCRS.

In the second part, we propose the Treasure Hunt Problem. In RCRS, the rescue team has to search more than one injured people and it is a complicated multi-agent problem. Therefore, study a simpler problem called the Treasure Hunt Problem, in which there is only one rescue crew search the only one injured civilian. In this problem, we assume that the location of the treasure is determined based on the probability distribution, and the ML model predicts the distribution of probability that treasure exists for each coordinate within the map. To solve this problem, we propose two search strategies that makes use of the ML model to improve the effectiveness of a search mission: (1) *the probabilistic greedy search* that the hunter searches preferentially for the cell with the highest probability of existing treasure given by ML model; and (2) *the probabilistically admissible heuristic A* search* that the hunter searches the cell determined by heuristic A* search with the probability of existing treasure given by ML model. In this experiment, we find that when the KL (i.e., the accuracy of the model) is low, there is an optimal grid size (i.e., the precision of model) with minimum search time. And the KL is high, there is a local optimal grid size. However, the larger the grid size, the less search time.

According to this experiment, when the search strategy using a machine learning model, there is optimal precision depends on the accuracy of the machine learning model. And as we expected, the optimal point is shown when the accuracy of the machine learning model is low. In the last part, we apply this study to the first part to optimize the precision in a deep learning model to locate the injured people in disaster zones.

In the last part, we merge the first and second parts to search for the location of the most density injured people area. To predict the location, we predict the number of injured people with several ML models used in the first part and we convert the injured people density predicted to the probability distribution. And the rescue team search the most density injured people area according to the search strategy of the second part based on this probability distribution. The result shows that when the accuracy of the machine learning model is low (i.e., DenseNet-201 and ResNeXt-101 model), there is an optimal grid size which is the minimum search time. And the accuracy of the machine learning model is high (i.e., InceptionResNet-v2 and Xception), when the grid size is larger, the search time is less. Therefore, we evaluate the search time according to the different ML model, we find the optimal grid size which showed the best performance (i.e., minimum search time) depend on the different accuracy of ML models.

In the future, we intend to our attention module and data annotation to apply the other domain (i.e., CIFAR). In this study, we evaluate our attention module (GCBAM) and Feature-Highlight annotation in several domains. However, the RSNA domain is not enough to evaluate our research. Demonstrating the performance improvements of our research in other diverse domains will have a great positive impact on expanding research direction and demonstrating performance. Furthermore, we intend to our machine learning model to evaluate the other city. In this study, we evaluate our machine learning model to predict the city of Japan, Kobe. This is the urban city used in the RoboCup competition. However, RCRS can simulate not only the city but also any other region. Therefore, if we evaluate our machine learning model not only in Kobe but also in cities in many other regions, this can reduce bias towards model performance evaluation.

We intend to expand the Treasure Hunt Problem to search not only one civilian but also more than one civilians. In this study, we evaluate our path-finding strategy to search only one injured person. Since there are several injuries in actual disaster situations, our research may not be enough to apply to real situations. Therefore, if we expand our search strategy to evaluate the search for multiple injured people in disaster zones, It will show that the Treasure Hunt problem could be more generalized, and some assumptions could be eliminated to evaluate the more accurate and realistic search strategy. And we intend to study the relationship between *the probabilistically admissible heuristic A* search* of parameters and search time with Treasure Hunt Problem. In this study, we turned the parameter of the search strategy (i.e., threshold λ) and grid size by the result of the experiment. Therefore, from further study, if we can derive these parameters mathematically, not experimentally, we can evaluate the performance of our search strategy and the machine learning model more precisely as mathematically experiment result.

Furthermore, It can lead to improved performance of models and search strategies. Therefore, we also can compare to our search strategy and other strategies in the other domains.

Lastly, we expand the scope of part three. In the study, the goal of the third part is the rescue crew search the areas that the most density of injured people so that the crew can save the injured people as much as possible at the least amount of time. However, in the real world, the rescue crew may find the other injured people while searching. Therefore, expand our research scope, our search strategy allow the rescue crew to save the other injured people while searching the most density of injured people area. This will be a study on the dilemma between the capacity of the rescue crew and the time to rescue all injured people. And this study will have a great positive impact on the performance of the efficiency of our search strategy.

References

[1] C. Goodman and D. Hogan, "Urban search and rescue," *Disaster medicine*, p. 79, 2007.

[2] C. Skinner and S. Ramchurn, "The robocup rescue simulation platform," in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*. International Foundation for Autonomous Agents and Multiagent Systems, 2010, pp. 1647–1648.

[3] L. D. Stone, "The process of search planning: current approaches and continuing problems," *Operations Research*, vol. 31, no. 2, pp. 207–233, 1983.

[4] R. Nair, T. Ito, M. Tambe, and S. Marsella, "Task allocation in the robocup rescue simulation domain: A short note," in *Robot Soccer World Cup*. Springer, 2001, pp. 751–754.

[5] P. R. Ferreira, F. Dos Santos, A. L. Bazzan, D. Epstein, and S. J. Waskow, "Robocup rescue as multiagent task allocation among teams: experiments with task interdependencies," *Autonomous Agents and Multi-Agent Systems*, vol. 20, no. 3, pp. 421–443, 2010.

[6] S. D. Ramchurn, A. Farinelli, K. S. Macarthur, and N. R. Jennings, "Decentralized coordination in robocup rescue," *The Computer Journal*, vol. 53, no. 9, pp. 1447–1461, 2010.

[7] A. Kleiner, M. Brenner, T. Bräuer, C. Dornhege, M. Göbelbecker, M. Luber, J. Prediger, J. Stückler, and B. Nebel, "Successful search and rescue in simulated disaster areas," in *Robot Soccer World Cup*. Springer, 2005, pp. 323–334.

[8] N. Schurr, P. Patil, F. Pighin, and M. Tambe, "Using multiagent teams to improve the training of incident commanders," in *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*. ACM, 2006, pp. 1490–1497.

[9] W. L. Black, "Discrete sequential search," *Information and control*, vol. 8, no. 2, pp. 159–162, 1965.

[10] T. H. Chung and J. W. Burdick, "Analysis of search decision making using probabilistic search strategies," *IEEE Transactions on Robotics*, vol. 28, no. 1, pp. 132–144, 2011.

[11] R. Isaacs, *Differential games: a mathematical theory with applications to warfare and pursuit, control and optimization*. Courier Corporation, 1999.

[12] S. Gal, "Search games with mobile and immobile hider," *SIAM Journal on Control and Optimization*, vol. 17, no. 1, pp. 99–122, 1979.

[13] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.

[15] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[17] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.

[18] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[19] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[20] V. Mnih, N. Heess, A. Graves *et al.*, "Recurrent models of visual attention," in *Advances in neural information processing systems*, 2014, pp. 2204–2212.

[21] J. Ba, V. Mnih, and K. Kavukcuoglu, "Multiple object recognition with visual attention," *arXiv preprint arXiv:1412.7755*, 2014.

[22] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014.

[23] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhudinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *International conference on machine learning*, 2015, pp. 2048–2057.

[24] K. Gregor, I. Danihelka, A. Graves, D. J. Rezende, and D. Wierstra, "Draw: A recurrent neural network for image generation," *arXiv preprint arXiv:1502.04623*, 2015.

[25] M. Jaderberg, K. Simonyan, A. Zisserman *et al.*, "Spatial transformer networks," in *Advances in neural information processing systems*, 2015, pp. 2017–2025.

[26] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7132–7141.

[27] S. Woo, J. Park, J.-Y. Lee, and I. So Kweon, "Cbam: Convolutional block attention module," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 3–19.

[28] P. Rui and K. Kang, "National hospital ambulatory medical care survey: Emergency department summary tables," 2014.

[29] S. L. Murphy, J. Xu, K. D. Kochanek, S. C. Curtin, and E. Arias, "Deaths: final data for 2015," 2017.

[30] T. Franquet, "Imaging of community-acquired pneumonia," *Journal of thoracic imaging*, vol. 33, no. 5, pp. 282–294, 2018.

[31] B. Kelly, "The chest radiograph," *The Ulster Medical Journal*, vol. 81, no. 3, p. 143, 2012.

[32] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, "Inception-v4, inception-resnet and the impact of residual connections on learning," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[33] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[34] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

VIII Appendix

This section is the appendix to describe in detail the result of the experiment.

8.1 Image-based Prediction in RCRS

This table describes the result of RMSE that the state-of-art deep learning model with SE, CBAM and GCBAM attention module in RCRS. And the machine learning model predicts the density of injured people using the single-frame image.

Description	Parameters	Grid Size	RMSE
ResNet-50 [16]	21.30M	4x4	2.216247
	21.33M	8x8	1.104256
ResNet-101	41.98M	4x4	2.155278
	42.01M	8x8	1.218160
ResNet-152	57.94M	4x4	2.311191
	57.97M	8x8	1.092777
DenseNet-121 [34]	7.05M	4x4	2.182839
	7.10M	8x8	1.271531
DenseNet-169	12.67M	4x4	2.999014
	12.75M	8x8	1.511546
DenseNet-201	18.35M	4x4	2.305295
	18.44M	8x8	1.146064
Inception-ResNetV2 [32]	54.37M	4x4	2.210150
	54.44M	8x8	1.050896
Inception-ResNetV2 + SE	71.40M	4x4	2.360719
	71.48M	8x8	1.128776
Inception-ResNetV2 + CBAM	71.41M	4x4	2.142055
	71.47M	8x8	1.067117
Inception-ResNetV2 + GCBAM	71.47M	4x4	2.137344
	71.54M	8x8	1.054906
InceptionV3 [33]	21.84M	4x4	2.400644
	21.93M	8x8	1.123869
Xception [19]	20.89M	4x4	2.110339
	21.00M	8x8	1.093639
Xception + SE	22.38M	4x4	2.371548
	22.47M	8x8	1.300128
Xception + CBAM	22.38M	4x4	2.231953
	22.48M	8x8	1.099724

Xception + GCBAM	22.38M	4x4	2.264135
	22.50M	8x8	1.148024

8.2 Image-based Prediction in RCRS: Feature-Highlight and Attention module

This table describe the result of RMSE that the Inception-ResNetV2 and Xception with our Feature-Highlight data annotation and attention module. The FH-RMSE means the result of RMSE that the machine learning model use our Feature-Highlight data annotation.

Description	Parameters	Grid Size	RMSE	FH-RMSE
Inception-ResNetV2 [32]	54.37M	4x4	2.216150	2.114966
	54.44M	8x8	1.114428	1.194564
Inception-ResNetV2 + SE	71.40M	4x4	2.360719	2.067913
	71.48M	8x8	1.128776	1.799768
Inception-ResNetV2 + CBAM	71.41M	4x4	2.142055	2.013006
	71.47M	8x8	1.067117	1.184809
Inception-ResNetV2 + GCBAM	71.47M	4x4	2.137344	2.445398
	71.54M	8x8	1.054906	1.154894
Xception [19]	20.89M	4x4	2.179926	2.395909
	21.13M	8x8	1.102236	1.231165
Xception + SE	22.38M	4x4	2.371548	2.186312
	22.47M	8x8	1.300128	1.211052
Xception + CBAM	22.38M	4x4	2.231953	2.095201
	22.48M	8x8	1.099724	1.237198
Xception + GCBAM	22.38M	4x4	2.264135	2.394272
	22.50M	8x8	1.148024	1.556173

8.3 Chicago Crime Location Prediction

This table describe the result of RMSE that the state-of-art deep learning model with SE, CBAM and GCBAM attention module in Chicago Crime dataset domain.

Description	Parameters	Grid Size	RMSE
ResNet-50 [16]	21.30M	4x4	0.249773
	21.33M	8x8	0.295023
ResNet-101	41.98M	4x4	0.176076
	42.01M	8x8	0.275893
ResNet-152	57.94M	4x4	0.306450

	57.97M	8x8	0.226034
DenseNet-121 [34]	7.05M	4x4	0.186130
	7.10M	8x8	0.315652
DenseNet-169	12.67M	4x4	0.189858
	12.75M	8x8	0.208664
DenseNet-201	18.35M	4x4	0.203783
	18.44M	8x8	0.260121
Inception-ResNetV2 [32]	54.36M	4x4	0.169207
	54.43M	8x8	0.257702
Inception-ResNetV2 + SE	25.14M	4x4	0.161596
	25.24M	8x8	0.233030
Inception-ResNetV2 + CBAM	25.15M	4x4	0.149808
	25.25M	8x8	0.228853
Inception-ResNetV2 + GCBAM	25.17M	4x4	0.197706
	25.27M	8x8	0.158368
InceptionV3 [33]	21.83M	4x4	0.174180
	21.93M	8x8	0.248352
Xception [19]	20.89M	4x4	0.152559
	21.00M	8x8	0.292797
Xception + SE	21.23M	4x4	0.164500
	22.34M	8x8	0.274904
Xception + CBAM	21.21M	4x4	0.215634
	23.01M	8x8	0.189147
Xception + GCBAM	21.21M	4x4	0.200959
	23.01M	8x8	0.195369

8.4 RSNA Pneumonia Detection Challenge

This table describe the result of RMSE that the state-of-art deep learning model with SE, CBAM and GCBAM attention module in RSNA Pneumonia detection challenge domain.

Description	Parameters	Grid Size	RMSE
ResNet-50 [16]	21.30M	4x4	0.157981
	21.33M	8x8	0.190919
ResNet-101	41.98M	4x4	0.125325
	42.01M	8x8	0.212713
ResNet-152	57.94M	4x4	0.176896
	57.97M	8x8	0.165897
DenseNet-121 [34]	7.05M	4x4	0.172405

	7.10M	8x8	0.247393
DenseNet-169	12.67M	4x4	0.177953
	12.75M	8x8	0.266117
DenseNet-201	18.35M	4x4	0.213121
	18.44M	8x8	0.270242
Inception-ResNetV2 [32]	54.37M	4x4	0.185789
	54.44M	8x8	0.239721
Inception-ResNetV2 + SE	71.40M	4x4	0.164949
	71.48M	8x8	0.231415
Inception-ResNetV2 + CBAM	71.41M	4x4	0.197762
	71.47M	8x8	0.217587
Inception-ResNetV2 + GCBAM	71.47M	4x4	0.212451
	71.54M	8x8	0.215849
InceptionV3 [33]	21.84M	4x4	0.195492
	21.93M	8x8	0.246776
Xception [19]	20.89M	4x4	0.211563
	21.00M	8x8	0.185649
Xception + SE	22.38M	4x4	0.182790
	22.47M	8x8	0.174921
Xception + CBAM	22.38M	4x4	0.185293
	22.48M	8x8	0.174513
Xception + GCBAM	22.38M	4x4	0.231399
	22.50M	8x8	0.150365

8.5 Video-based Prediction in RCRS

This table describe the result of RMSE that the state-of-art deep learning model with SE, CBAM and GCBAM attention module in RCRS. And the machine learning model predict the density of injured people using the multi-frame image.

Description	Parameters	Grid Size	RMSE
ResNet-50 [16]	21.59M	4x4	3.221900
	21.68M	8x8	1.482911
ResNeXt-50 + SE	28.41M	4x4	3.493942
	28.57M	8x8	1.482978
ResNeXt-50 + CBAM	28.41M	4x4	3.904281
	28.57M	8x8	1.490002
ResNeXt-50 + GCBAM	28.43M	4x4	3.845660
	28.59M	8x8	1.469520

ResNet-101	42.27M	4x4	3.395557
	42.36M	8x8	1.673974
ResNeXt-101 + SE	52.10M	4x4	3.271026
	52.26M	8x8	1.495123
ResNeXt-101 + CBAM	52.10M	4x4	3.235420
	52.26M	8x8	1.492024
ResNeXt-101 + GCBAM	52.16M	4x4	3.640757
	52.50M	8x8	1.493750
ResNet-152	58.22M	4x4	3.623339
	58.31M	8x8	1.665147
DenseNet-121 [34]	7.34M	4x4	3.232334
	7.49M	8x8	1.715784
DenseNet-169	12.95M	4x4	3.785897
	12.52M	8x8	1.686833
DenseNet-201	18.64M	4x4	4.029794
	18.79M	8x8	1.493864
ResNeXt-50 [18]	23.08M	4x4	3.759389
	23.52M	8x8	1.690866
ResNeXt-101	42.58M	4x4	3.242727
	42.40M	8x8	1.532850
Inception-ResNetV2 [32]	54.64M	4x4	3.724961
	54.78M	8x8	1.719019
InceptionV3 [33]	22.12M	4x4	4.202212
	22.28M	8x8	1.517941
Xception [19]	21.18M	4x4	3.832643
	21.38M	8x8	1.732041

8.6 Video-based Prediction in RCRS: Feature-Highlight and Attention module

This table describe the result of RMSE that the Inception-ResNetV2 and Xception with our Feature-Highlight data annotation and attention module. The FH-RMSE means the result of RMSE that the machine learning model use our Feature-Highlight data annotation.

Description	Parameters	Grid Size	RMSE	FH-RMSE
ResNet-50 [16]	21.59M	4x4	3.221900	3.221900
	21.68M	8x8	1.482911	1.482911
ResNeXt-50 + SE	28.41M	4x4	3.493942	3.201041
	28.57M	8x8	1.482978	1.477944

ResNeXt-50 + CBAM	28.41M	4x4	3.904281	3.188726
	28.57M	8x8	1.490002	1.479471
ResNeXt-50 + GCBAM	28.43M	4x4	3.845660	4.396645
	28.59M	8x8	1.469520	1.469921
ResNet-101	42.27M	4x4	3.395557	3.182134
	42.36M	8x8	1.673974	1.479393
ResNeXt-101 + SE	52.10M	4x4	3.271026	3.394058
	52.26M	8x8	1.495123	1.467555
ResNeXt-101 + CBAM	52.10M	4x4	3.235420	3.201932
	52.26M	8x8	1.492024	1.483710
ResNeXt-101 + GCBAM	52.16M	4x4	3.640757	3.201932
	52.50M	8x8	1.493750	1.485297

Acknowledgements

I would like to thank my thesis advisor Tsz-Chiu Au of the Electrical and Computer Engineering at Ulsan National Institute of Science and Technology. The door to Prof. Au office was always open whenever I ran into a trouble spot or had a question about my research or writing. He consistently allowed this paper to be my own work, but steered me in the right the direction whenever he thought I needed it.