# Individual Prediction Modelling for Air Traffic Control using Supervised Learning

Master of Science Thesis

L.J.A. Kloosterman

Faculty of Aerospace Engineering
Department of Control & Operations
Delft University of Technology

# Individual Prediction Modelling for Air Traffic Control using Supervised Learning

## Master of Science Thesis

by

L.J.A. Kloosterman

| | | | |
|---|---|---|---|
| Student number: | 4697006 | | |
| Project duration: | October 1, 2020 – January 13, 2022 | | |
| Thesis committee: | Prof. dr. ir. M. Mulder, | TU Delft, | examiner |
| | Dr. ir. C. Borst, | TU Delft, | examining supervisor |
| | Dr. ir. E. van Kampen, | TU Delft, | examining supervisor |
| | Dr. ir. E. Mooij, | TU Delft, | external examiner |
| | Ir. T. Nunes, | TU Delft, | non-examining supervisor |

TU Delft

# Contents

# List of Figures

# List of Tables

# Introduction

Historically the aviation industry has shown a steady growth which is expected to continue in the long-term according to SESAR JU [76]. In order for this growth to be able to continue, Air Traffic Control (ATC) has to be able to accommodate these extra flights. However, as argued by Majumdar et al. [46] the airspace capacity is already limited by the workload an Air Traffic Controller (ATCo) can manage. Multiple ATC solutions are being developed making the role of the ATCo redundant in the future, but do not include a transition plan from the current system. Hence, as argued by Hoekstra et al. [29], fully automated ATC solutions are not feasible in the near future. On top of this, transferring to an automated system in which the controller is not considered often leads to worse performance as argued by Bainbridge [6], as humans gain more knowledge of the system by interacting with it. In order to increase airspace capacity in the shorter term, an automated system should support the ATCo with decision making instead of replacing him.

An example of a decision support system is the Solution Space Diagram (SSD) which was introduced in the work of Velasco [47]. It is a visual tool which gives ATCos a clear overview of all conflict-free velocity vectors that could be issued to a pilot. Multiple studies have shown that the SSD has a significant effect on the reduction of controller workload [47, 19, 34, 9]. Furthermore, Westin et al.[85] introduced the notion of strategic conformance which aims to make automation individual-sensitive to the controller currently using the system. Enough evidence was provided to reasonably hypothesize that an advised solution, from automation, would be more readily accepted if it matches the individual problem-solving style of the controller. Combining the SSD and strategic conformance, Van Rooijen et al.[59] introduced a Convolutional Neural Network (CCN) which is trained on the actions of an individual controller. Using the SSD as input to the network allows the sytem and the controller to work from a shared mental model. With experiments, Van Rooijen et al.[59] showed that trained models can reasonably predict resolutions using three separate CNNs for command, type, and magnitude. In addition, it was found that a model achieves higher predictive performance if the controller has a more consistent problem-solving style.

## Problem Statement

This research of van Rooijen et al. [59] can be seen as a proof of concept into the capability of an individual-sensitive trained CNN to predict controller resolutions given a SSD as input. This research will expand on this, by evaluating how such model should be constructed in order to establish an optimal human-machine system which will be accepted by the controller. The following research question is defined to pursue this goal:

***”How can acceptance of Convolutional Neural Networks for Conflict Detection & Resolution be improved for use by Air Traffic Controllers?”***

It is hypothesized that, an ATCo is more likely to accept an advisory CNN during Conflict Detection & Resolution (CD&R) if an automation system is able to predict resolutions more accurately. Therefore, this research will explore how such system should be constructed with the sole objective of increasing predictive performance.This will be done from two different viewpoints:

- the most efficient use of a CNN in the context of CD&R;
- the optimization of the input to the CNN, i.e. feature engineering;

## Methodology

In order to answer the research question a literature review will be presented first. In which the state-of-the-art regarding CD&R, acceptance of automation, CNNs and SSDs will be elaborated on. After this, two experiments are conducted in order to analyse the viewpoints presented above. Both experiments follow the same four phases as presented in figure 1 being Simulation, Data Generation, Training and

Evaluation. First, a simulation is done in which resolutions are applied to conflict situations in order to create data. This data is transformed into data sets which consist of SSDs describing the situation and the corresponding resolutions. After this, a CNN is trained such that it can predict the resolutions given the input. Finally, a test data set is used to evaluate the performance of the model.

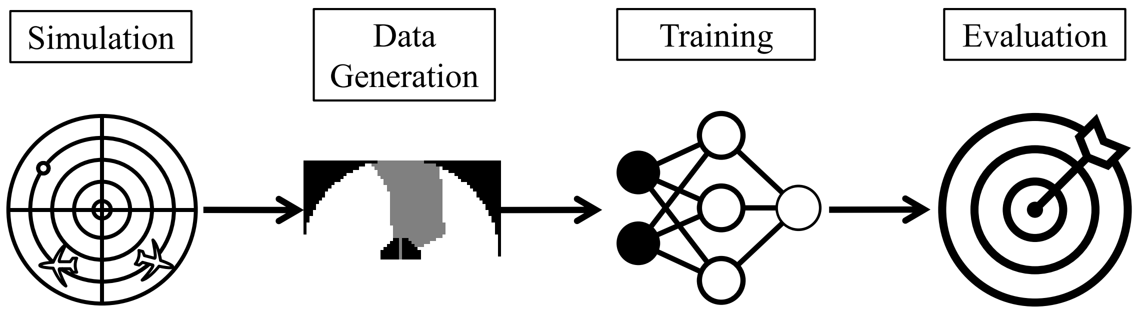


**Figure 1:** The overall research methodology followed during both experiments.

## Report Outline

- **Part I: Scientific Paper**
  The knowledge obtained during the literature review, preliminary analyses and the final experiment are combined in a paper in order to answer the main research question.
- **Part II: Preliminary Thesis**
  This part starts with a literature review which covers the state-of-the-art regarding CD&R and automation, CNN's, the SSD, feature engineering, Explainable Artificial Intelligence and interface design. After this, the methodology and the results of the initial analysis will be discussed. Lastly, this part will give a proposition for the main experiment.
- **Part III: Appendices**
  Appendices which give more detailed information on the execution and intermediate results of the main experiment are presented in this part. The chapter follows the four main phases of the main experiment.
- **Part IV: Conclusions and Recommendations**
  This report is finalised by stating the conclusion found in all previous parts followed by recommendations for future research.

## Part I

# Scientific Paper

# Individual Prediction Modelling for Air Traffic Control Using Architecture Optimization and Feature Engineering

L.J.A. Kloosterman (MSc student)
*Supervisors: C. Borst, E. van Kampen, T. Nunes*
*Section Control & Simulation, Department Control and Operations, Faculty of Aerospace Engineering*
*Delft University of Technology, Delft, The Netherlands*

**In the future, Air Traffic Controllers are expected to work together with more advanced computer-based automation that can automatically take action. The main challenge is then how to design such computer-based tools such that they foster acceptance among air traffic controllers. One possible approach to foster acceptance is by matching the automated decisions and actions to individual human problem-solving styles, the so-called strategic conformance. Another approach is by making the automated tool more transparent and thus interpretable. Previous research aimed to combine these two approaches by making use of the Solution Space Diagram, a decision-support tool for Conflict Detection and Resolution, as a visual feature for a supervised machine learning method that aimed to generate individual human prediction models. Results were promising, but prediction accuracy could be significantly improved. In this study, the impact of feature engineering and a revised machine learning architecture on prediction accuracy will be investigated. This is done by evaluating different feature engineering and architecture options using data generated by a simulation in which Conflict Detection and Resolution is performed. It was found that a Convolutional Neural Network can accurately predict exact resolutions using regression and a more optimized architecture is introduced which significantly increases predictive performance. Furthermore, it is concluded that a larger solution space results in a slight increase in predictive performance while using a color scheme with more colors does not necessarily result in a higher predictive performance.**

## I. Introduction

Advanced computer-based automation systems are expected to work together with Air Traffic Controllers (ATCo) more often in the future [1]. As en-route air space capacity is limited by ATCo workload [2], a demand exists for methods that allows safe control over more en-route aircraft simultaneously. To accommodate this, fully automated ATC solutions are being researched [1]. However, a transition from the current ATC system to such solution is not feasible in the near future as a transition plan often does not exist [3]. Instead of using fully automated ATC solutions, automation systems should support ATCo's in the form of an efficient human-machine collaboration.

The main challenge faced by any implementation of an automation system is controller acceptance [4] which will only be achieved if the controller trusts the system [5]. An automated system which is never used, will not provide benefits in efficiency and therefore not contribute to increasing airspace capacity. Understanding the functioning of the system is the main component for a controller needed in order to trust an automated system[6]. Therefore, to overcome the hurdle of acceptance, controller conformance and transparency should be taken into account when designing an automated system.

Recent work has introduced the theoretical concept of strategic conformance in order to increase controller acceptance [4]. A strategically conformal system is constructed in such a way that the problem solving strategy is equivalent to the strategy of the human controller. Human-in-the-loop experiments have shown that systems which are strategically conformal are accepted more often when compared to systems which are not [4, 7]. A suitable technique to achieve strategic conformance is Supervised Learning (SL) which uses labeled data in order to train algorithms to predict outcomes accurately. It is expected that an accurate supervised learning algorithm, trained on personal controller decisions, will produce solutions which are in conformance with the controllers strategy.

Another approach to increase transparency and acceptance is the introduction of the Solution Space Diagram (SSD) [8]. This is a visual tool which gives the ATCo a clear overview of safe and unsafe heading and speed combinations for a particular aircraft. In a recent study, the SSD was used as an input to an individually trained model in order to predict resolutions [9]. As the figure inputted into the system is identical to the figure shown to the ATCo, human and machine work from a shared mental model which results in higher interpretability. Using data from a human-in-the-loop experiment it was shown that a CNN can successfully predict the bin to which a certain resolution belongs, e.g. a heading change to the left larger than 45 degrees. One could argue that a classification CNN predicts a certain strategy instead of an exact resolution depending on the size of the bins. This means the ATCo would still be required to formulate an exact resolution within the predicted bin without any help.

The main task of Air Traffic Control (ATC) regarding en-route traffic is Conflict Detection & Resolution (CD&R) and is therefore the primary focal point of this research. Under normal operations, the minimum horizontal separation between aircraft is defined to be 5 nautical miles [10]. During CD&R, ATCos predict the trajectory of aircraft, detect conflicts and resolve them in order to maintain sufficient separation. This paper will introduce an alternative CNN architecture performing a regression task in order to predict an exact resolution in the form of a velocity vector. The main ambition will be to create an efficient personalised prediction model in terms of predictive power using two approaches: architecture optimization and feature engineering. For the purposes of this research, predictive power will relate to the ability of the model to predict a certain velocity vector. As a velocity vector is defined by a combination of heading and speed, increasing predictive power is equivalent to decreasing the true heading and true speed error.

During a preliminary test, it is analysed whether a CNN is capable of predicting resolutions using regression given an SSD as input. Additionally, the influence of the shape and size of the SSD's solution space on the predictive power will be evaluated. The main focus of this research will be to find an optimized CNN architecture and hyperparameters in order to obtain satisfactory training behaviour. Additionally, changes to the color coding of the SSD will be done to analyse the effect in terms of predictive power. The final design of the SSD is intended to be optimised for the CNN while still being interpretable for an ATCo to maintain a shared mental model.

The next section gives an introduction into the SSD and CNNs and evaluates how they were used for predicting resolutions in a recent study. The third section will discuss the methods used during this research and the experiment results. After this, a discussion is provided and recommendations for future research will be given. Finally, conclusions to the research will be given in the final section.

## II. Background

The main concepts used to build the resolution prediction model introduced in this paper are the SSD and CNN. In order to illustrate how the model is constructed, background information on these concepts is given in this section. After this, a recently introduced prediction model [9] consisting of CNNs will be discussed as well as opportunities to improve upon it.

### A. Solution Space Diagram

The SSD is a visual support tool which incorporates multiple parameters applicable to the CD&R task. In recent years, it has been used for various application but was first introduced in an aerospace context by van Dam et al.[11] as a self-separation tool. After this, multiple researches have been conducted into the SSD related to workload. Hermes et al.[12], d'Engelbronner et al.[13] and Mercado Velasco et al.[8] all concluded that the use of the SSD has a positive effect on controller situational awareness and workload.

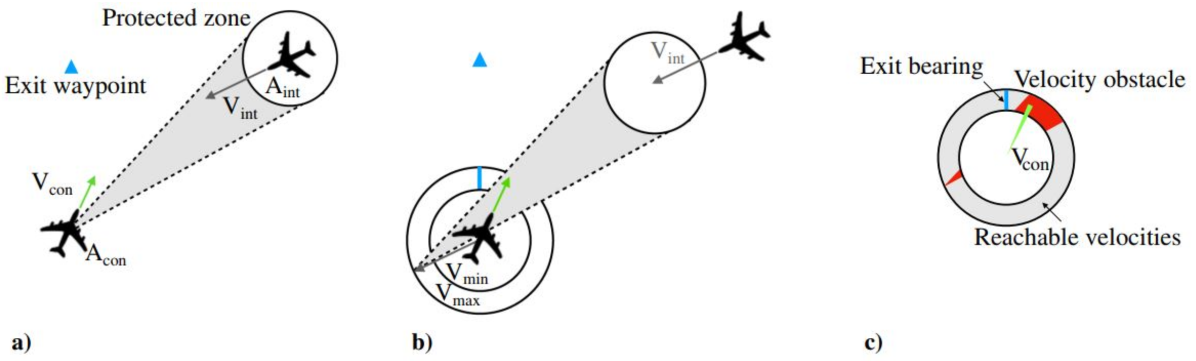


**Figure 1. Construction of the SSD: a) conflicting relative velocities, b) flight envelope and velocity vector displacement, and c) limit to solution space [9].**

Figure 1 gives a representation of how a SSD is constructed. The SSD is constructed by drawing a triangle, known as the Forbidden Beam Zone (FBZ) [14], from the controlled aircraft to the edge of the intruding aircraft's protected zone. Two circles are drawn around the controlled aircraft to indicate the minimum and maximum airspeed. The FBZ is shifted with the velocity vector of the intruding aircraft and the velocity vector of the controlled aircraft is added to obtain the SSD. When the velocity vector ends in the area enclosed by the FBZ the aircraft's trajectory will result in Loss of Separation (LoS). The space between the velocity limitations and outside the FBZ is referred to as the solution space. Multiple FBZ's can be present inside the SSD if there is a situation with multiple intruding aircraft.

## B. Convolutional Neural Networks

From the year 2000 onwards, neural networks became very adequate for detection, segmentation and recognition tasks. A few examples of these tasks are pedestrian detection [15], biological segmentation from microscopical images [16], traffic sign recognition [17]), localization of joints [18] and autonomous off-road driving [19, 20]. In recent years, CNN architectures have become larger, solving more complex problems. Recently, an architecture was introduced using 60 million trainable parameters in order to classify 1.2 million high-resolution images into 1000 different classes for different objects [21]. As the input to the prediction model to be constructed in this research is an image, a CNN is the most obvious model choice.

A neural network is constructed using multiple layers consisting of a set of neurons. Introduced by Rosenblatt [22], a neuron is defined as

$$y = \phi\left(\sum_n w_n x_n + b\right) \tag{1}$$

where inputs $x_n$ are weighted with $w_n$ and summed with bias $b$. This summation is fed to an activation function $\phi$ which calculates the output $y$. A CNN consists of at least one convolutional layer whose filters slide over the image to create a feature map. A filter consists of a set of weights which together are able to detect a certain feature from the image. Equation 2 gives a small example of how the convolutional operation is performed.

$$\overset{\text{Input}}{\begin{bmatrix} 0 & 1 & 3 \\ 4 & 5 & 6 \\ 6 & 7 & 8 \end{bmatrix}} * \overset{\text{Filter}}{\begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix}} = \overset{\text{Output}}{\begin{bmatrix} 24 & 31 \\ 38 & 44 \end{bmatrix}} \tag{2}$$

The training of a CNN is done by optimising the weights and biases using a labelled data set. Each image of the data set is fed to the untrained network during the first iteration. A performance metric is used to compare the network output with the respective label. In order to update the weights of the network, the partial derivatives $\frac{\partial E}{\partial w_n}$ of the error E with respect to all weights $w_n$ are calculated. The weights are updated by

$$w_{n+1} = w_n - lr\frac{\partial E}{\partial w_n} \tag{3}$$

with $lr$ being the learning rate.

Running an image through the network is referred to as the forward pass while updating the weights is referred to as the backward pass. If the forward and backward pass are executed consecutively an epoch is completed. In order for a network to learn how to accurately predict a certain outcome, multiple epochs are needed. After each epoch, progress is monitored using a performance metric until satisfactory performance is achieved. An often used performance metric for numerical data, which will also be used in this research, is the Mean Squared Error (MSE) [23].

### C. Resolution Prediction

In order to create a prediction model using a CNN with the SSD as visual input, four general steps are needed: simulation, data generation, model training and evaluation. The simulation is preferably done using data from ATCo's as they are the intended users of the prediction model. However, due to the amount of data needed and the time constraints involved performing an experiment with real ATCos a computerized simulation using an existing CD&R algorithm can also be used. From the simulation a data set has to be created consisting of SSD images and their true label or value. In order to train a CNN a suitable architecture and hyperparameter tuning is needed in order to find satisfactory training results. Finally, an evaluation metric has to be chosen which fits the output format of the network.

In the recent work of van Rooijen et al. [9] an architecture of three independent CNNs was introduced for resolution prediction. Each of these networks (as shown in figure 2) predict one of the components which together form a resolution, e.g. resolution type, direction and value. The resolution type network predicts whether the situation can best be solved with a heading (HDG), speed (SPD) or direct-to command (DCT). The direction network advises left or right and faster or slower in case of a HDG or SPD command, respectively. The final network, in case of a HDG command, predicts the bin to which the magnitude of the heading change relative to the current heading belongs. These bins are defined as [0, 10], [10, 45] and greater than 45 degrees. In case of a SPD command two bins were predicted: [200, 250] and [250, 290] knots.

Using 12 novice ATCos, data was generated during multiple CD&R simulation runs from which individual data sets were created in order to train the model. The data sets consisted of SSDs (as shown in figure 3) and the commands issued by controllers. As HDG changes larger than 90 degrees are uncommon, the bottom half of the SSD was excluded from the image. The FBZs are displayed using three colors depending on time to closest point of approach ($t_{cpa}$), i.e. red ($t_{cpa} < 60s$), orange ($60 < t_{cpa} < 120s$) and gray ($t_{cpa} > 120s$). Additionally, the velocity vector of the controller aircraft is shown in green and the direction to the exit waypoint is added to the SSD with blue.

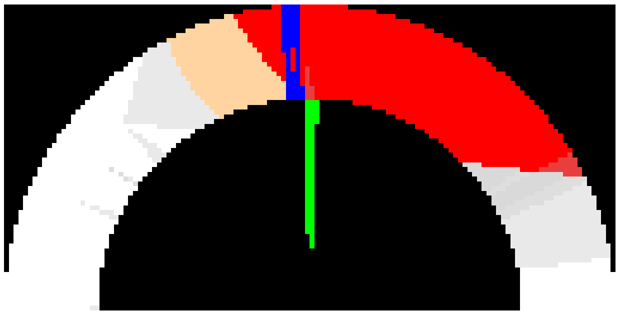


**Figure 3. Example of an SSD used in the research of van Rooijen et al. [9].**

Using this model, Van Rooijen et al. [9] achieved a mean Matthews Correlation Coefficient of 0.52, 0.76 and 0.64 for type, direction and value prediction respectively. It was concluded that more research is needed in terms of model architecture and feature engineering the input in order to improve predictive performance.

#### *1. Model Architecture*

One could argue that this model does not support the ATCo formulating a safe resolution. The main reason being the large bin size, i.e. as the model advises a certain bin the controller still has to find a safe resolution within that bin. After having received an advise, the controller should decide whether to accept it after which he returns to the original task of finding a safe resolution. The ensemble in which a safe resolution has to be found has reduced in size but an extra action is added for the controller. Therefore, it would be worthwhile to reduce the bin size or predict exact velocity vectors.

The airspace is a continuous state space and the

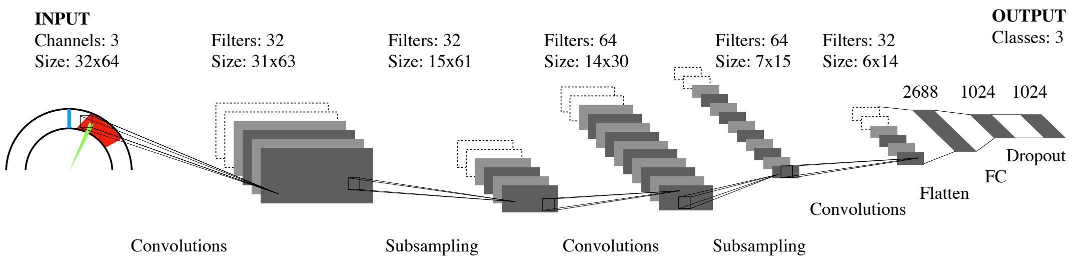


**Figure 2. Overview of one of the CNNs used in the research of van Rooijen et al. [9]. All filters inside the convolutional layers have a size of 2x2 pixels.**

FBZ's presented inside an SSD are not bounded by any bins. In addition to classification, CNNs can also be used to solve regression problems, e.g. image orientation prediction [24] and 3D pose estimation of objects [25]. The problems addressed in these studies show similarities to the prediction of exact velocity vectors. Therefore, this research will explore the options of using regression for exact resolution prediction with CNNs given the SSD as input.

### 2. *Feature Engineering*

The input images used in the current model are clearly interpretable for a human controller, but are not optimised as input to a CNN. Feature engineering provides a framework to increase predictive power by changing the input. Observing figure 3 it can be seen that a significant area of the image does not contain any useful information. These pixels are irrelevant for the calculation of the output, but are fed through the network. This inefficiency could be solved by increasing the size used for solution space.

Furthermore, the SSD is a polar representation of all velocity vectors while the input to the CNN is a Cartesian grid of pixel values. This means a Cartesian grid of equally sized squares will be laid over the solution space. Considering the polar representation, the area used for equal ensembles of velocity vectors is not evenly distributed. Hence, the CNN processes the low speed velocity vectors using less pixels compared to the high speed velocity vectors. This could have the consequences that low and high speed velocity vectors are calculated with different accuracies.

On top of this, only three colors are used to indicate the $t_{cpa}$ of the intruder aircraft. Red, orange and grey, are easily distinguishable for humans but it may be possible that this is not the most optimal way to present it to a CNN as it reads information as a set of 2D matrices. Hence, it could be fruitful to use a color scheme with more colors to increase the resolution of the intruder's $t_{cpa}$. Another option would be to use a continuous color scheme where the pixel value is directly related to a certain $t_{cpa}$.

Lastly, the FBZs of the SSDs used by van Rooijen et al. [9] indicates $t_{cpa}$ with different colors. One could argue that time to Loss of Seperation ($t_{los}$) is more critical information compared to $t_{cpa}$. A primary task of an ATCo during CD&R is detecting conflicts. A conflict occurs when an aircraft is on a trajectory which will result in LoS [10]. Hence, the time in which LoS will happen is therefore more useful for an ATCo. The model introduced in this research will use SSDs form which the colors of the FBZs indicate $t_{los}$.

## III. Methodology

From the previous sections it was concluded that, predictive performance may be improved by introducing a different model architecture and feature engineering the SSD. Regarding the model architecture, it should first be explored whether predicting resolutions in a regressive manner is possible at all before finding a more optimized model architecture. Additionally, it will be evaluated how a SSD with a larger or Cartesian solution space and a FBZ color scheme with more colors influences the predictive power of the overall model. In an initial test, the possibility of using regression and using a SSD with a larger and Cartesian solution space are analysed. The best performing prediction method and SSD format will be used as a starting point for the main experiment. In this experiment, an effort is done to find a more optimized architecture and multiple color coding schemes are evaluated.

## A. Initial Testing

Initial tests were performed to analyse the possibility of predicting exact resolutions using regression and to evaluate the impact of using SSDs with different configurations. Namely, a SSD in which the solution space is maximized and a Cartesian version of the SSD. Due to the preliminary nature of these tests, only pairwise conflicts are included during this phase. In each scenario the controlled aircraft ($A_{con}$) is heading North while an intruding aircraft ($A_{int}$) has a conflicting trajectory as can be seen in figure 4. In total, 8605 conflict pairs are generated by varying $A_{con}$'s conflict angle, CPA and look-ahead-time as presented in table 1.

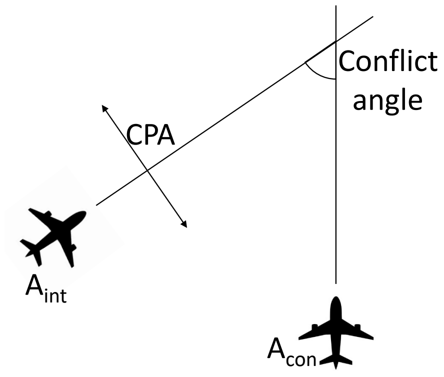


**Figure 4. Simulation geometry.**

**Table 1. Parameters for data generation.**

| Parameter | Value | Unit |
|---|---|---|
| Conflict angle | [20, 21 .. 349, 350] | deg |
| CPA | [-3, -2.5..2.5, 3] | nm |
| Look-ahead time | 150, 300 | s |

Three different SSD formats are introduced, the Standard, Zoomed and Cartesian SSD, from which individual data sets are created. Figure 5 shows an example of these SSD formats given the same scenario.

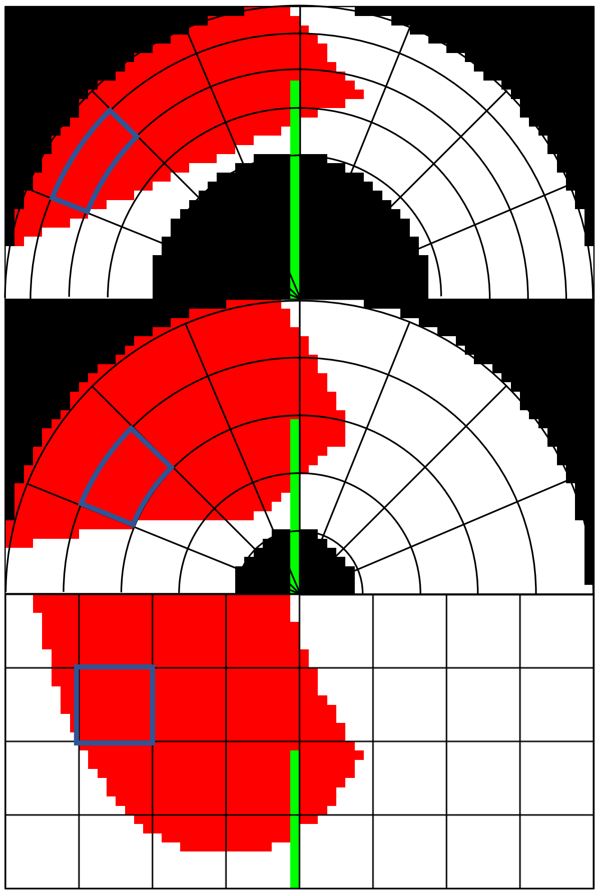


**Figure 5. From top to bottom an example of the Standard, Zoomed and Cartesian SSD given the same scenario, respectively. The blue cell presents how the same area is mapped.**

The Standard SSD is constructed using the same method as described in subsection II.A and will be used as a baseline.

The Zoomed SSD is designed such that the solution space is maximised by manually defining the radii which indicate minimum and maximum airspeed. For each pixel inside the solution space, the related airspeed is calculated as

$$v = v_{min} + (v_{max} - v_{min})\frac{d_{x,y} - d_{min}}{d_{max} - d_{min}} \tag{4}$$

with $v_{min}$ the minimum true airspeed, $v_{max}$ the maximum true airspeed, $d_{x,y}$ distance between pixel x,y and the center of the SSD, $d_{min}$ lower solution space boundary and $d_{max}$ upper solution space boundary. The airspeed together with the heading, as seen from the center of the SSD, form the velocity vector of a particular pixel. Using the velocity vector of that pixel the flight trajectory is calculated. In case of a trajectory resulting in a conflict with an intruding aircraft the pixel is colored. From figure 4 it can be observed that the Zoomed SSD uses more space, and thus more pixels, for the same solution space area. It is expected that this will result in better predictive performance. A disadvantage of this format is the variable solution space resolution if the flight envelope changes. If the flight envelope becomes larger, the area used for solution space remains the same, meaning more information needs to be present in the same space. Additionally, it can be observed that a single FBZ has lost it triangular shape while it still presents the SSD in a circular shape.

The Cartesian SSD is constructed using the same method as the Zoomed SSD but velocity vectors are calculated using horizontal and vertical pixel positions. Hence, all pixels on the same column or row represent equal heading or speed, respectively. From figure 4 it can be seen that the area used for solution space is further expanded compared to the Zoomed SSD. It can also be observed that an equal amount of area is now used for low and high speed velocity vectors. It is expected that this SSD format further increases predictive performance, especially in terms of speed prediction.

For each data set, a CNN was trained which has an identical architecture as the CNN used by van Rooijen et al. [9] except for the final layer. In order to be able to output a specific resolution, in the form of a velocity vector, the final layer is modified such that it has two normalized output values for heading and speed. Table 2 shows an overview of all the hyperparameters used to train all three networks.

**Table 2. Training hyperparameters.**

| **Hyperparameter** | **Value** |
|---|---|
| Optimizer | Stochastic Gradient Descent |
| Loss function | Mean Squared Error |
| Train/val/test ratio | 0.6/0.15/0.25 |
| Learning rate | 0.01 |
| Dropout rate | 20 |
| Input dimension | 32x64 |

Training over 100 epochs produced converging learning curves for all data sets as shown in figure 6. Hence, it is possible for a CNN to learn exact resolutions in the form of a velocity vector given SSDs as input. The Zoomed SSD and Cartesian SSD have shown slightly better performance compared to the Standard SSD for the training and validation data set while the difference between the Zoomed and Cartesian SSD is negligible.

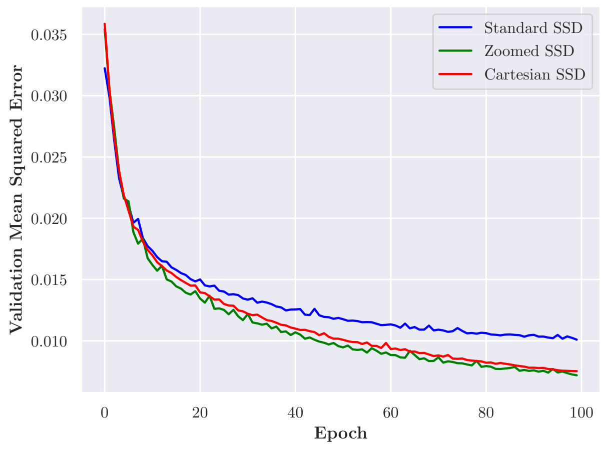


**Figure 6. MSE during training calculated using the normalized output of the validation data set.**

In the final phase of the initial experiment the test data set is used to evaluate the three final models. Table 3 shows the average absolute heading and speed error on the test data set for each SSD format. Similar to the training performance, the Zoomed and Cartesian SSDs outperform the Standard SSD while the difference between them is negligible. However, it can be observed that the Cartesian SSD performs marginally better when it comes to predicting velocities.

**Table 3. Mean true heading and speed errors and standard deviations for different SSD formats.**

| | **Mean errors** | |
|---|---|---|
| | HDG [degrees] | SPD [knots] |
| **Standard SSD** | 0.66 ± 12.64 | 0.98 ± 25.24 |
| **Zoomed SSD** | −0.44 ± 10.73 | −0.82 ± 20.99 |
| **Cartesian SSD** | −0.32 ± 11.12 | −0.61 ± 20.84 |

It can be argued that the Cartesian SSD is not conformal to the mental model of the controller. In contrast to the Zoomed and Standard SSDs, it has lost its circular shape which means it has lost its intuitiveness and interpretability. Therefore, it was concluded that, given the simulation used in this research, the Zoomed SSD is most suitable as input to the network as it gives optimal performance while being conformal to the mental model of the ATCo.

## B. Experiment

Predicting resolutions using regression instead of classification fundamentally changes the task of the CNN. Hence, it is likely that the architecture used in the initial test is non-optimal for this prediction model. Furthermore, the SSD can be further feature engineered by optimizing the color coding of the FBZ to improve model performance. To further investigate how a CD&R prediction model can most optimally be constructed, an experiment is conducted to find a better performing architecture and to analyse the influence of FBZ color coding schemes. The experiment consists of four phases as presented in figure 7.

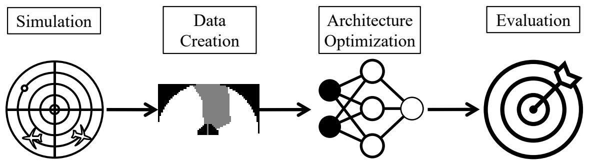


**Figure 7. The four experimental phases.**

all possible conflicts between intruding aircraft are ignored.

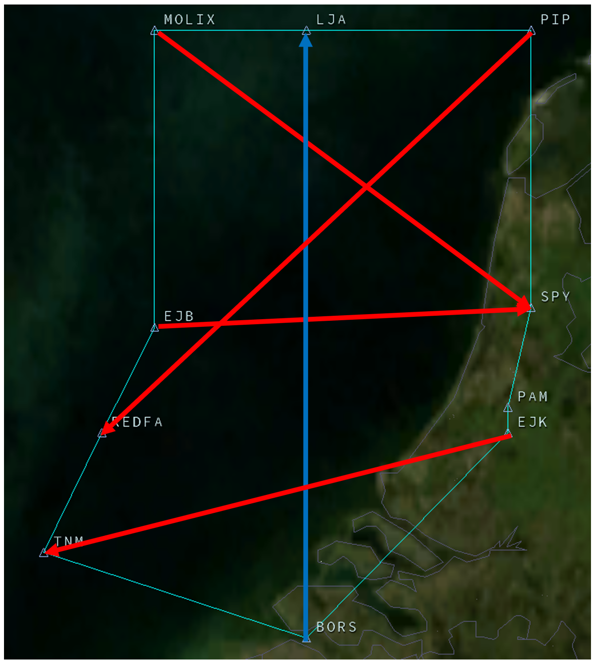


**Figure 8. ATC area used during simulation. The initial trajectories of the controlled and intruding aircraft are presented in blue and red, respectively.**

### *1. Simulation*

In order to generate data, simulations have to be conducted. These simulations only consider horizontal resolutions and all aircraft fly at equal altitude with equal initial ground speed. Due to time constraints and the amount of data needed, the simulation is performed using the Modified Voltage Potential (MVP) algorithm [26] in a fictional ATC area (figure 8) created in BlueSky [27]. The simulation consists of controlled aircraft flying from South to North crossing four airways at which intruding aircraft have to be avoided. Intruding aircraft appear at the beginning of each waypoint at random intervals with sufficient separation to simulate variable traffic flows. All aircraft have equal true entry velocity of 320 knots and the same fixed altitude. The controlled aircraft is the only one allowed to perform resolution maneuvers while

Using this area, the simulation is done twice with different look-ahead-times and separation factors to replicate different controller strategies. The time at which a resolution is initiated before LoS would have happened is defined as the look-ahead-time. The separation factor is a margin placed on top of the separation distance of 5 nm, i.e. a separation factor of 2 means that resolutions are aimed to result in a separation of 10 nm. The Reactive and Proactive strategies have a look-ahead-time of 2.5 and 5 minutes and a separation factor 2 and 1.5, respectively. This results in the Reactive strategy performing more aggressive resolutions with larger heading and speed changes compared to the Proactive strategy as can be seen in figure 9. The simulation has run until approximately 10,000 conflicts have been collected for each strategy.

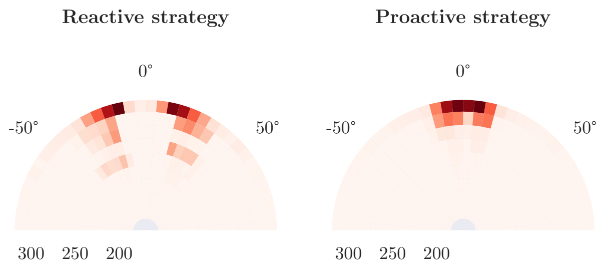


**Figure 9. Heat-Map of the resolution distribution for the Proactive and Reactive strategy, respectively.**

### 2. Data Generation

For every variation to the FBZ color coding two data sets are created, one for each strategy, using the format of the Zoomed SSD. All SSD versions will be cropped to only the top half before being fed into the network, resulting in a fixed image size 32x64 pixels. A color scheme with 3 colors will be used as a baseline for which an example is shown in figure 10. The colors indicate $t_{los}$: red ($t_{los} < 180s$), orange($180s < t_{los} < 300s$) and gray ($300s < t_{los} < 600s$).

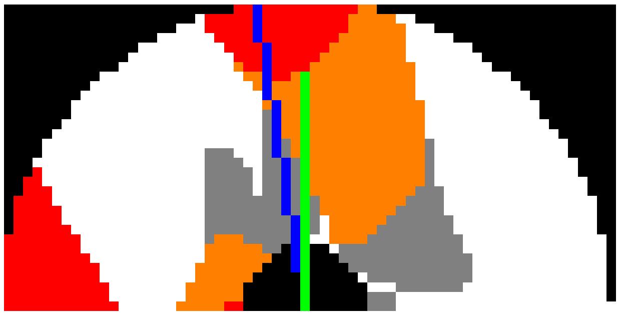


**Figure 10. Example of a SSD used as baseline for this experiment.**

The colors gray, orange and red are commonly used in aviation to indicate different levels of concern as they are easily distinguishable for humans. A CNN however, reads pixel values as numbers and can therefore distinguish a near infinite amount of colors accurately. In addition to the baseline SSD, 5 other SSDs using different color schemes are introduced to analyse the influence on predictive performance. Figure 11 shows these versions given the same situation as the baseline SSD presented by figure 10. The 600 tint version uses a different tint for every second of $t_{los}$ which linearly transforms from dark red to red, to orange, to grey and white at $t_{los}$ of 0, 180, 300, 450 and 600, respectively. It is chosen to also include the colors dark red and white in order to add more difference to the color scheme, i.e. using red and grey at $t_{los}$ of 0 and 600, respectively results in orange being the dominant color in most SSDs. The SSD versions using 5, 10 and 20 tints are constructed using the same color scheme as the 600 tint scheme using steps of equal length. Lastly, a version using only the color red was added to analyse the outcome of using less colors than the baseline SSD.

### 3. Architecture Optimization

As an explicit procedure to find a suitable CNN architecture does not exist, it is often a trial and error process. Due to the large number of parameters that can be changed, an almost infinite amount of combinations can be analysed which can not be done due to time constraints. Instead, the architecture used by van Rooijen et al. [9] (figure 2) will be used as a starting point, since it already has proven descent performance in the initial test. From this network, a more optimal architecture will be sough after in a 7 step process. At each step, multiple variations to the architectures will be trained using the baseline SSD (figure 10) for both strategies. The best performing model will progress to the next step. Aside from the first step, the steps are ordered in descending order of expected influence in predicted performance. In order to evaluate all variations at each step the model was trained with the hyperparameters from table 2 for 500 epochs after which training curves were compared.

1) **Padding:** At each convolutional layer without padding, the image size reduces which limits the number of layers and size of filters which can be applied. Therefore, it is evaluated at this step whether or not padding has a negative influence on the model. It was found, as expected, that padding does not have a negative influence meaning more variations can be analysed in the next steps.
2) **Ouput layer:** A linear activation function is commonly used in the output layer of a regressive network [28]. It was found that a linear output layer results in better performance com-

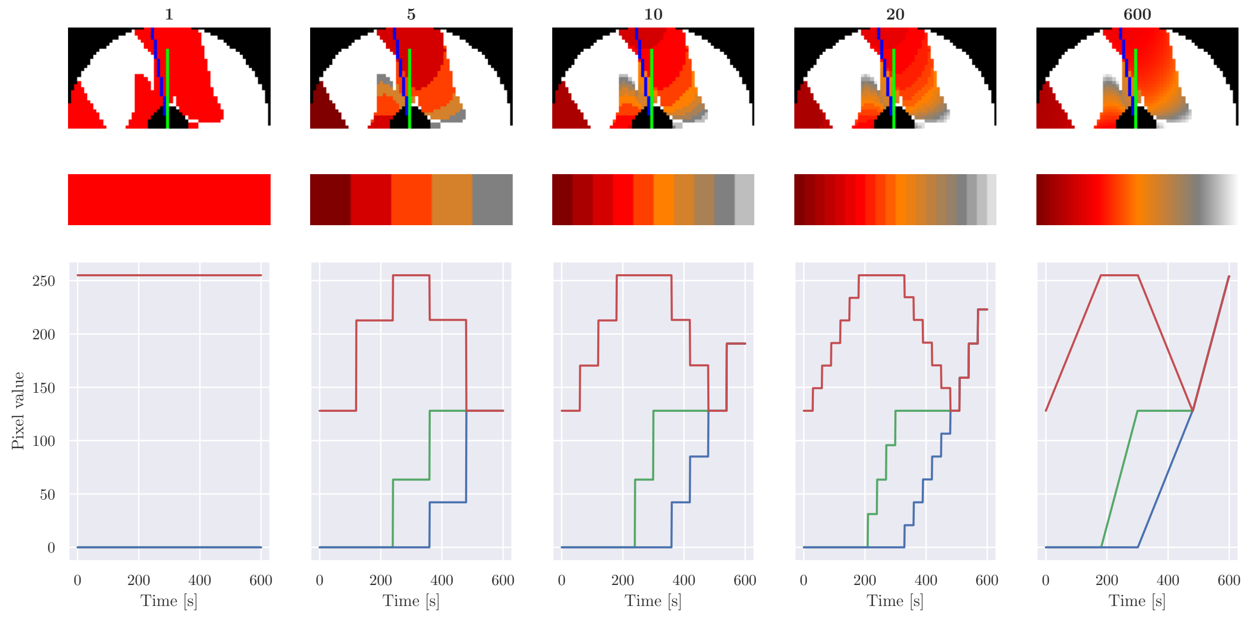


**Figure 11. Example SSDs for different color scheme variations describing the same situation. Below, the color schemes itself and pixel values for individual RGB colors are presented for different $t_{los}$.**

pared to the current ReLU output layer.

3) **Filter size:** A commonly used characteristic of CNNs is an odd filter size which expands towards the end of the network [28], e.g. 3x3, 5x5 and 7x7 for a CNN consisting of three convolutional layers. The baseline architecture however uses only 2x2 filters which is less common. Next to the baseline architecture, a setup is evaluated where the three convolutional layers have a filter size of 3x3, 5x5 and 7x7, respectively. This last setup has shown to significantly increase training performance.
4) **Number of layers:** Another fundamental variable of any CNN is the number of convolutional layers. Testing architectures with 2, 3 and 4 convolutional layers it was found that 3 convolutional layers gives the best trade-off between terms of predictive power and computational power.
5) **Layer depth:** Compared to the baseline architecture a shallower and deeper architecture were evaluated with sizes of 16-32-64 and 64-128-256. It was found that the original depth of 32-64-128 results in the best performance.
6) **Dropout:** Another way to increase predictive performance is placing a dropout before the final layer where some inputs are randomly set to zero. In this experiment dropout rates of 0, 10, 20 and 30% are evaluated. The original dropout rate of 20% is found to achieve the best performance.
7) **Optimization algorithm:** Some more elaborate optimization algorithms are evaluated which automatically tune learning rate, momentum, etc. ADAM and RMSProp are both state of the art optimizers which have shown accurate results in recent years [29]. Evaluating them it was shown that both have better overall training performance compared to Stochastic Gradient Descent while ADAM slightly outperforms RSMProp.

Table 4 gives an overview of the final architecture after optimization. It was found that the ADAM optimization algorithm and an initial learning rate of 0.001 results in the best training performance and the highest predictive performance. Figure 12 presents the validation errors obtained by training the baseline and final architecture. Comparing the baseline with the final architecture it can be observed that the new architecture results in: 1) faster convergence, 2) lower absolute error, 3) more variability and 4) performance decreases over time due to overfitting. As a result of the performance decrease early stopping is applied, i.e the weights of the model are saved after a new minimum is found and training will stop after 50 epochs without any improvement. The training curve variability does not compromise the overall performance of the model and is therefore left as is.

**Table 4. Final network architecture.**

| Layer type | Input size | Filter size | Depth | Activation |
|---|---|---|---|---|
| CONV | 32x64x3 | 3x3 | 32 | ReLU |
| POOL | 32x64x32 | | | |
| CONV | 16x32x32 | 5x5 | 64 | ReLU |
| POOL | 16x32x64 | | | |
| CONV | 8x16x64 | 7x7 | 128 | ReLU |
| Flatten | 8x16x128 | | | |
| FC | 2688 | | | ReLU |
| Dropout | 1024 | | | |
| FC | 1024 | | | Linear |

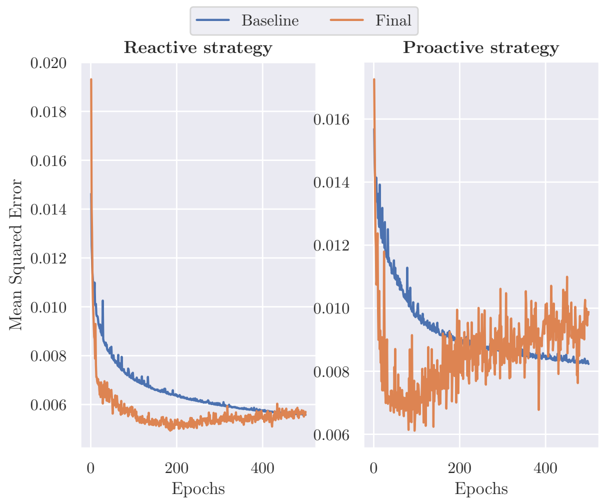


**Figure 12. Mean Squared Error achieved after each epoch by feeding the validation data set into the network.**

Using the newly introduced architecture and training procedure, training is done for all SSD versions defined during the previous step of this experiment. Figure 13 shows the validation error obtained during training for both controller strategies. It can be observed that all SSD versions, except the SSD using only one color, show similar training performance in terms of training curve and the final error. Regarding convergence speed, significant differences are only found for the Reactive strategy. Additionally, it can be seen that the lowest global MSE is found using the 10 and 600 color SSD version for the Reactive and Proactive strategy, respectively.

## *4. Evaluation*

Using the test data set, the final models are evaluated for both strategies. The outputs of the CNN are converted back to true heading and speed values and subtracted from the true values coming from the data set to obtain the true error. Table 5 presents the mean value and interquartile ranges of these errors for all models. It can be observed that the architecture introduced in this paper achieves higher predictive performance for both strategies regarding heading and speed errors. Regarding the different color schemes, it can be observed that differences in predictive performance are minimal excluding the one color version which shows worse performance. Considering interquartile range, it can be concluded that using the the SSD with 20 and 5 different colors results in the most optimal performance for the Reactive and Proactive strategy, respectively.

Finally, it is analysed how the model generalizes given a different ATC area with different conflict dynamics. The controlled aircraft flight trajectory from figure 8 is reversed to create a simulation with different conflict dynamics. For both strategies a simulation is done, creating 1000 conflicts approximately, using the same MVP settings as used during the original simulation. Data sets are created using the baseline color scheme which is evaluated using the weights from the final model. Table 6 shows the mean true heading and speed errors and interquartile ranges given the original and generalization simulation. It can be observed that predictive performance reduces substantially if a model is evaluated using data from a simulation with different conflict geometry. This means that the weights of the final model will only result in accurate performance given the conflict geometries generated by the original simulation. Hence, the model is only applicable to the ATC area used in this research.

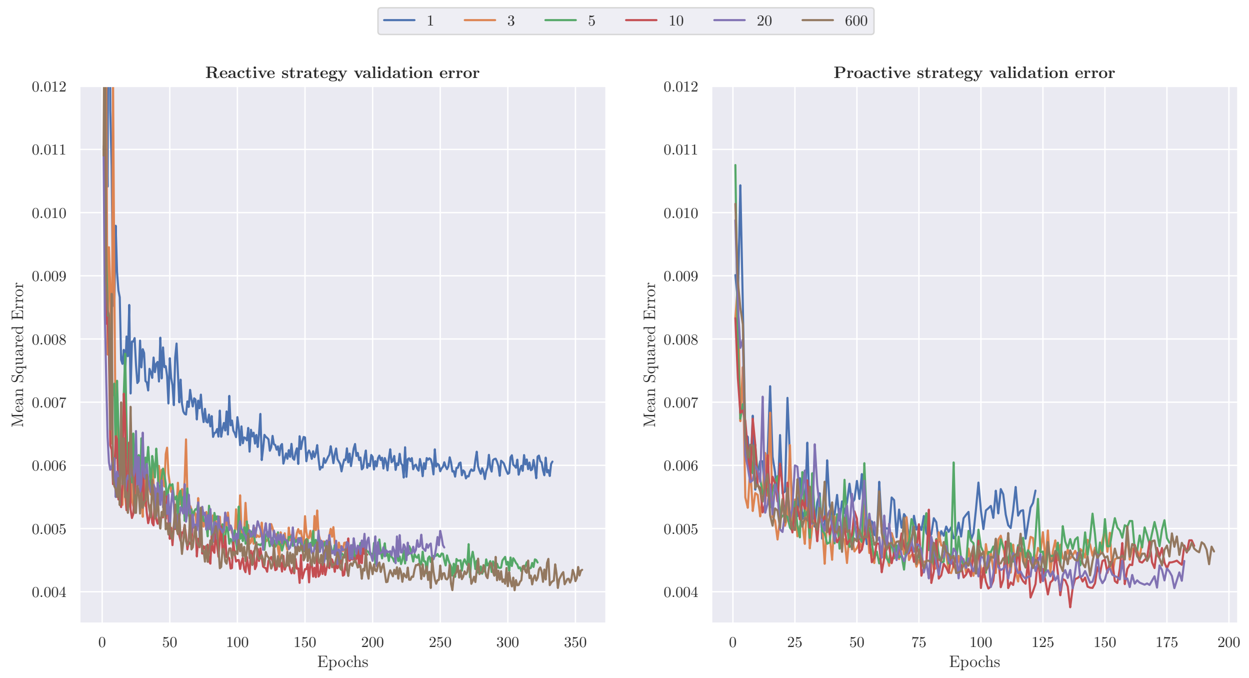


**Figure 13. Validation error during training for the baseline and final architecture.**

**Table 5. Mean true heading and speed errors and interquartile ranges shown between brackets for both architectures given multiple color schemes.**

| | **Number** | **Reactive strategy** | | **Proactive strategy** | |
|---|---|---|---|---|---|
| **Architecture** | **of colors** | HDG [degrees] | SPD [m/s] | HDG [degrees] | SPD [m/s] |
| Baseline | 3 | 0.62 (7.78) | 0.11 (4.18) | 0.27 (5.62) | -0.14 (3.94) |
| Final | 1 | -0.14 (5.30) | 0.35 (2.55) | 0.31 (4.10) | -0.04 (3.18) |
| | 3 | -0.32 (3.90) | 0.07 (2.08) | -0.06 (2.92) | -0.22 (2.42) |
| | 5 | 0.46 (3.44) | 0.50 (1.74) | 0.40 (2.67) | 0.40 (2.19) |
| | 10 | -0.17 (3.65) | 0.43 (1.82) | -0.43 (2.91) | 0.62 (2.10) |
| | 20 | -0.42 (3.34) | 0.34 (1.48) | 0.05 (2.72) | -0.10 (2.42) |
| | 600 | 0.27 (5.62) | 0.57 (1.47) | -0.80 (2.79) | 0.55 (2.18) |

**Table 6. Mean true heading and speed errors and interquartile ranges shown between brackets for the initial and generalization simulation given the baseline SSD with the final architecture (table 4).**

| | **Reactive strategy** | | **Proactive strategy** | |
|---|---|---|---|---|
| **Simulation** | HDG [degrees] | SPD [m/s] | HDG [degrees] | SPD [m/s] |
| Initial | -0.32 (3.90) | 0.07 (2.08) | -0.06 (2.92) | -0.22 (2.42) |
| Generalization | -3.36 (10.66) | 5.21 (8.04) | 2.30 (5.90) | 4.25 (7.59) |

## IV. Discussion

The main objective of this research was to achieve individual-sensitive personalized automation which will be accepted by controllers. By introducing a prediction model using SL which takes the SSD steps have been made towards this main ambition. Using the work of van Rooijen et al. [9] as a starting point, a different model architecture was introduced and the SSD is feature engineered to optimize performance. A model which regresses to a specific velocity vector instead of predicting a certain class bin was proposed.

As the newly introduced model regresses to a single velocity vector, a different performance measure was needed compared to the work of van Rooijen et al. [9]. Hence, a direct measurement in terms of performance between the model proposed in this paper and the model proposed by van Rooien et al. [9] was not possible. However, from a qualitative point of view, one could argue that a model which predicts an exact resolution more accurately advises resolution to its controller compared to a model which predicts a certain bin. Also, it can be observed from table 5 that all interquartile ranges are lower than the bins used in the research of van Rooijen et al.[9]. Therefore, it can be concluded that a model which uses regression outperforms a model which uses classification in terms of predictive performance given the simulations used in this research.

Due to the extensive amount of data needed and the current Covid measurements it was not possible to generate data through human-in-the-loop experiments. All data was collected through simulations in which resolution commands were issued by the MVP algorithm [26]. The settings of the algorithm have been tuned such that it represents a human controller as much as possible and it was assumed that the resolutions generated are representative for a professional human ATCo. There is no guarantee however that the results obtained in this research are also valid for a prediction model trained using human data. Additionally, a large data set was generated to ensure that training was possible. The minimum amount of data needed is still unknown and is probably highly dependent on data set variation, controller strategy and traffic complexity.

Nevertheless, two different MVP [26] settings were used to simulate different controller strategies. The Proactive strategy has a larger look-ahead-time and therefore acts sooner compared to the Reactive strategy resulting in more subtle resolutions. This means the spread of resolutions issued is smaller resulting in a simpler task for the prediction model. This can also be confirmed by observing the validations errors which can be observed in figure 13. For the Reactive strategy, training performance increases as more colors are used while this is not necessarily true for the Proactive Strategy.

Regarding feature engineering, it was proven that a relation exists between the size of the solution space and predictive power. The solution space was maximized while still maintaining the layout of the original SSD. Future research could further analyse the solution space size as the maximum size may not be the most optimal. Additionally, it was shown that adding more colors to the FBZ color scheme does not necessarily increase predictive power. The original SSD using only three colors was designed such that the colors are easily distinguishable by a human operator. However, the influence of adding more or less colors to the FBZ on the controllers judgement is not analysed in this research. Another direction in which further research could be directed is more feature engineering techniques which could potentially increase predictive performance, e.g. a larger image, including the bottom half, adding the exit waypoint of conflicting aircraft, etc.

Regarding architecture optimization, an extensive search for an architecture which results in better performance is done. Nevertheless, many more design options can be explored and different combinations could potentially result in better performance. The steps in which the new architecture is found are ordered in descending expected influence on predictive performance. Due to time constraints not all possible combinations have been explored meaning it can not be stated that the introduced architecture has optimal performance. Additionally, it might be true that a different architecture results in better performance when a different data set is used, i.e. difference in traffic scenarios, controller strategy and SSD format. Future research could potentially further analyse the influence of different data sets on architecture performance.

## V. Conclusion

This research introduced an alternative Convolutional Neural Network architecture performing a regression task in order to predict an exact resolution in the form of a velocity vector given a Solution Space Diagram as input. Using data generated by an initial simulation it was proven that Convolutional Neural Networks are capable of predicting resolutions accurately using regression. Furthermore, a Solution Space Diagram format with a larger solution space was introduced which resulted in higher predictive power. A more elaborate simulation was conducted with different settings to simulate two controllers with different strategies. For each strategy, 6 data sets were created using the newly introduced Solution Space Diagram format. Each data set consisted of Solution Space diagram which uses 1, 3, 5, 10, 20 and 600 colors to indicate different times to loss of separation. Using the 3 color Solution Space Diagram data sets, architecture optimization and hyperparameter tuning has been done which has shown to significantly increase predictive power. No considerable differences are found between the different color schemes using the introduced architecture except for the decreased performance caused by the Solution Space Diagram which uses one color. However, as these conclusions are solely based on the simulations done during this research, a guarantee that such model behaves similarly when used on human data does not exist. Future research therefore could potentially further evaluate the model given data from real Air Traffic Controllers.

## References

[1] Undertaking, S. J., *European ATM Master Plan Executive view*, Publication Office of the European Union, Luxembourg, 2020.

[2] Majumdar, A., Ochieng, W., McAuley, G., Lenzi, J., and Lepadatu, C., "The Factors Affecting Airspace Capacity in Europe: A Cross-Sectional Time-Series Analysis Using Simulated Controller Workload Data," *Journal of Navigation*, Vol. 57, 2004, pp. 385 – 405. doi: 10.1017/S0373463304002863.

[3] Hoekstra, J. M., van Gent, R. N., and Ruigrok, R. C., "Designing for safety: the 'free flight'air traffic management concept," *Reliability Engineering & System Safety*, Vol. 75, No. 2, 2002, pp. 215–232.

[4] Hilburn, B., Westin, C., and Borst, C., "Will controllers accept a machine that thinks like they think? The role of strategic conformance in decision aiding automation," *Air Traffic Control Quarterly*, Vol. 22, No. 2, 2014, pp. 115–136.

[5] Lee, J. D., and See, K. A., "Trust in Automation: Designing for Appropriate Reliance," *Human Factors*, Vol. 46, No. 1, 2004, pp. 50–80. doi: 10.1518/hfes.46.1.50_30392, URL `https://doi.org/10.1518/hfes.46.1.50_30392`, pMID: 15151155.

[6] Hoff, K., and Bashir, M., "Trust in Automation: Integrating Empirical Evidence on Factors That Influence Trust," *Human Factors The Journal of the Human Factors and Ergonomics Society*, Vol. 57, 2015, pp. 407–434. doi: 10.1177/0018720814547570.

[7] Westin, C., Borst, C., and Hilburn, B., "Strategic Conformance: Overcoming Acceptance Issues of Decision Aiding Automation?" *IEEE Transactions on Human-Machine Systems*, Vol. 46, 2015, pp. 41–52. doi: 10.1109/THMS.2015.2482480.

[8] Mercado-Velasco, G., Mulder, M., and Paassen, M. V., "Analysis of Air Traffic Controller Workload Reduction Based on the Solution Space for the Merging Task," *AIAA Guidance, Navigation, and Control Conference*, 2010. doi: 10.2514/6.2010-7541, URL `https://arc.aiaa.org/doi/abs/10.2514/6.2010-7541`.

[9] Rooijen, S., Ellerbroek, J., Borst, C., and Van Kampen, E.-J., "Toward Individual-Sensitive Automation for Air Traffic Control Using Convolutional Neural Networks," *Journal of Air Transportation*, 2020, pp. 1–9. doi: 10.2514/1.D0180.

[10] ICAO, *Doc 4444 - Procedure for Air Navigation Services - Air traffic Management*, ICAO, 999 Robert-Bourassa Boulevard, Montréal, Quebec, Canada, 2016.

[11] Van Dam, S. B., Abeloos, A. L., Mulder, M., and Van Paassen, M., "Functional presentation of travel opportunities in flexible use airspace: An EID of an airborne conflict support tool," *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*, Vol. 1, IEEE, 2004, pp. 802–808.

[12] Hermes, P., Mulder, M., Van Paassen, M., Boering, J., and Huisman, H., "Solution-space-based complexity analysis of the difficulty of aircraft merging tasks," *Journal of Aircraft*, Vol. 46, No. 6, 2009, pp. 1995–2015.

[13] d'Engelbronner, J., Mulder, M., Van Paassen, M., De Stigter, S., and Huisman, H., "The use of the dynamic solution space to assess air traffic controller workload," *AIAA Guidance, Navigation, and Control Conference*, 2010, p. 7542.

[14] Visser, M., van Dam, S., Mulder, M., and van Paassen, M., "Towards an Ecological Design of

a 4-Dimensional Separation Assistance Interface," *Proceedings of the 3rd International Conference on Human Centered Processes (HCP-2008), Workshop " Supervisory Control in Critical Systems Management", Delft, The Netherlands*, edited by C. Brézillon Patrick and Lenca Philipe, Telecom Bretagne, 2008, pp. 1–6. Null ; Conference date: 08-06-2008 Through 12-06-2008.

[15] Sermanet, P., Kavukcuoglu, K., Chintala, S., and Lecun, Y., "Pedestrian Detection with Unsupervised Multi-stage Feature Learning," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013, pp. –.

[16] Turaga, S. C., Murray, J. F., Jain, V., Roth, F., Helmstaedter, M., Briggman, K., Denk, W., and Seung, H. S., "Convolutional networks can learn to generate affinity graphs for image segmentation," *Neural computation*, Vol. 22, No. 2, 2010, pp. 511–538.

[17] CireşAn, D., Meier, U., Masci, J., and Schmidhuber, J., "Multi-column deep neural network for traffic sign classification," *Neural networks*, Vol. 32, 2012, pp. 333–338.

[18] Tompson, J., Goroshin, R., Jain, A., LeCun, Y., and Bregler, C., "Efficient Object Localization Using Convolutional Networks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015, pp. –.

[19] Hadsell, R., Sermanet, P., Ben, J., Erkan, A., Scoffier, M., Kavukcuoglu, K., Muller, U., and LeCun, Y., "Learning long-range vision for autonomous off-road driving," *Journal of Field Robotics*, Vol. 26, No. 2, 2009, pp. 120–144.

[20] Muller, U., Ben, J., Cosatto, E., Flepp, B., and Cun, Y. L., "Off-road obstacle avoidance through end-to-end learning," *Advances in neural information processing systems*, 2006, pp. 739–746.

[21] Krizhevsky, A., Sutskever, I., and Hinton, G. E., "Imagenet classification with deep convolutional neural networks," *Communications of the ACM*, Vol. 60, No. 6, 2017, pp. 84–90.

[22] Rosenblatt, F., "The perceptron: a probabilistic model for information storage and organization in the brain." *Psychological review*, Vol. 65, No. 6, 1958, p. 386.

[23] Rumelhart, D. E., Hinton, G. E., and Williams, R. J., "Learning representations by back-propagating errors," *nature*, Vol. 323, No. 6088, 1986, pp. 533–536.

[24] Fischer, P., Dosovitskiy, A., and Brox, T., "Image orientation estimation with convolutional networks," *German Conference on Pattern Recognition*, Springer, 2015, pp. 368–378.

[25] Mahendran, S., Ali, H., and Vidal, R., "3d pose regression using convolutional neural networks," *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2017, pp. 2174–2182.

[26] Hoekstra, J. M., and Ellerbroek, J., "Bluesky ATC simulator project: an open data and open source approach," *Proceedings of the 7th International Conference on Research in Air Transportation*, Vol. 131, FAA/Eurocontrol USA/Europe, 2016, p. 132.

[27] Hoekstra, J., and Ellerbroek, J., "BlueSky ATC Simulator Project: An Open Data and Open Source Approach," *7th International Conference on Research in Air Transportation*, 2016, pp. –. URL `http://www.icrat.org/icrat/index.cfm,http://www.icrat.org/icrat/7th-international-conference/`, 7th International Conference on Research in Air Transportation, ICRAT'16 ; Conference date: 20-06-2016 Through 24-06-2016.

[28] Brownlee, J., *Deep learning with Python: develop deep learning models on Theano and TensorFlow using Keras*, Machine Learning Mastery, 2016.

[29] Zou, F., Shen, L., Jie, Z., Zhang, W., and Liu, W., "A sufficient condition for convergences of adam and rmsprop," *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 11127–11135.

## Part II

# Preliminary Thesis

**Graded under course code AE4020**

# 1 Introduction

Historically the aviation industry has shown a steady growth which is expected to continue in the long-term according to SESAR JU [76]. In order for this growth to be able to continue, Air Traffic Control (ATC) has to be able to accommodate these extra flights. However, as argued by Majumdar et al. [46] the airspace capacity is already limited by the workload of an Air Traffic Controller (ATCo) can manage. Multiple ATC solutions are being developed making the role of the ATCo redundant in the future but do not include a transition plan from the current system. Hence, as argued by Hoekstra et al. [29], fully automated ATC solutions are not feasible in the near future. On top of this, transferring to an automated system in which the controller is not considered often leads to worse performance as argued by Bainbridge [6], as humans gain more knowledge of the system by interacting with it. In order to also increase airspace capacity in the shorter term, an automated system should support the ATCo with making decision instead of replacing him.

An example of a decision support system is the Solution Space Diagram (SSD) which was introduced in the work of Velasco [47]. It is a visual tool which gives ATCo's a clear overview of all conflict-free velocity vectors that could be issued to a pilot. Multiple studies have shown that, the SSD has a significant effect on the reduction of controller workload [47, 19, 34, 9]. Furthermore, Westin et al.[85] introduced the notion of strategic conformance which aims to make automation individual-sensitive to the controller currently using the system. Enough evidence was provided to reasonably hypothesize that an advised solution, from automation, would be more readily accepted if it matches the individual problem-solving style of the controller. Combining the SSD and strategic conformance, Van Rooijen et al.[59] introduced a Convolutional Neural Network (CCN) which is trained on the actions of an individual controller. Using the SSD as input to the network allows the sytem and the controller to work from a shared mental model. With experiments, Van Rooijen et al.[59] showed that trained models can reasonably predict resolutions using three separate CNNs for command, type, and magnitude. Furthermore, it was found that a model achieves higher predictive performance if the controller has a more consistent problem-solving style.

This research of van Rooijen et al. [59] can be seen as a proof of concept into the capability for an individual-sensitive trained CNN to predict controller resolutions given a SSD as input. This research will expand on this, by evaluating how such model should be constructed in order to establish an optimal human-machine system which will be accepted by the controller. This will be done from three different viewpoints:

- the most efficient use of a CNN in the context of CD&R;
- the optimization of the input to the CNN;
- and the presentation of the advised solution towards the ATCo.

It is hypothesized that, an ATCo is more likely to accept an advisory CNN during CD&R if predictive power is increased and the presentation of the advised solution is done in such way that it provides reasoning behind a certain solution.

As the objective of this research is to increase acceptance of automation during CD&R a more specified research question can be formulated:

***"How can acceptance of Convolutional Neural Networks for Conflict Detection & Resolution be improved for use by Air Traffic Controllers?"***

To be able to answer the research question, four sub-questions are formulated which are summarized below and visually shown in figure 1.1.

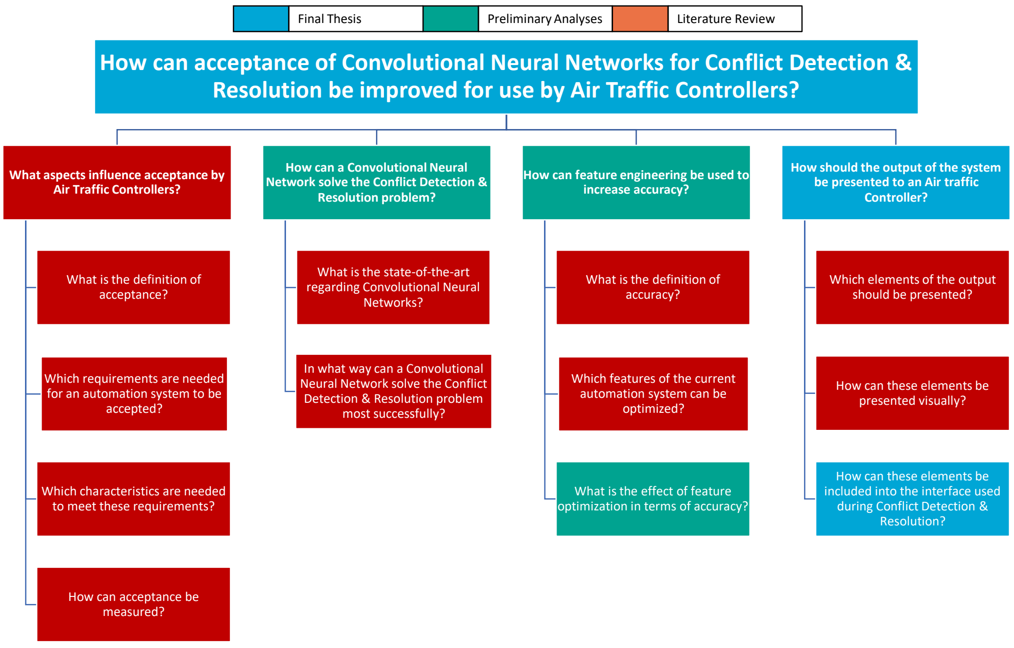

Final Thesis
Preliminary Analyses
Literature Review
How can acceptance of Convolutional Neural Networks for Conflict Detection & Resolution be improved for use by Air Traffic Controllers?
What aspects influence acceptance by Air Traffic Controllers?
How can a Convolutional Neural Network solve the Conflict Detection & Resolution problem?
How can feature engineering be used to increase accuracy?
How should the output of the system be presented to an Air traffic Controller?
What is the definition of acceptance?
Which requirements are needed for an automation system to be accepted?
Which characteristics are needed to meet these requirements?
How can acceptance be measured?
What is the state-of-the-art regarding Convolutional Neural Networks?
In what way can a Convolutional Neural Network solve the Conflict Detection & Resolution problem most successfully?
What is the definition of accuracy?
Which features of the current automation system can be optimized?
What is the effect of feature optimization in terms of accuracy?
Which elements of the output should be presented?
How can these elements be presented visually?
How can these elements be included into the interface used during Conflict Detection & Resolution?

**Figure 1.1:** Overview of research questions with colors that indicate in which phase it is expected that the question is answered.

1. **What aspects influence acceptance by Air Traffic Controllers?**
   (a) What is the definition of acceptance?
   (b) Which requirements are needed for an automation system to be accepted?
   (c) Which characteristics are needed to meet these requirements?
   (d) How can acceptance be measured?
2. **How can a Convolutional Neural Network solve the Conflict detection & Resolution problem?**
   (a) What is the state-of-the-art regarding Convolutional Neural Networks?
   (b) In what way can a Convolutional Neural Network solve the Conflict Detection & Resolution problem most successfully?
3. **How can feature engineering be used to increase accuracy?**
   (a) What is the definition of accuracy?
   (b) Which features of the current automation system can be optimized?
   (c) What is the effect of feature optimization in terms of accuracy?
4. **How should the output of the system be presented to an Air traffic Controller?**
   (a) Which elements of the output should be presented?
   (b) How can these elements be presented visually?
   (c) How can these elements be included into the interface used during Conflict Detection & Resolution?

This research attempts to answer the main question in three different phases: being the literature review, the preliminary analysis and the final thesis. During the literature review SQ1 will be answered and possible methods to answer SQ2, SQ3 and SQ4 will be found and analysed. During the preliminary analyses, basic experiments will be executed in order to find the best method to answer the SQ2, SQ3 most accurately. In the last phase, research regarding SQ4 will be executed after which the research question will be answered by a discussion.

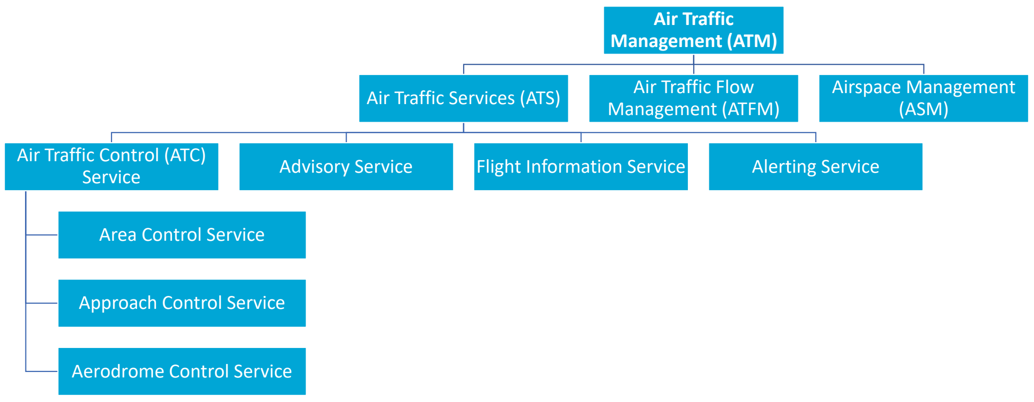
This preliminary thesis contains the findings of the literature review, the findings of the preliminary analyses and a proposal for the final experiment. In chapter 2, the state-of-the-art regarding CD&R and automation will be discussed and an attempt will be made to answer SQ1. After this, a greater understanding of CNN's and feature engineering will be established. With this knowledge the algorithm used in the work of van Rooijen[59] will be analysed in order to find opportunities to increase its performance in terms of predictive power. The third chapter will give an overview of promising XAI techniques applicable to the problem addressed in this research. At the end of the literature review several different interface design frameworks will be discussed. Chapter 6 will discuss the execution and findings of the preliminary analyses. After this, the results of the literature review and preliminary analysis will be concluded and a direction for further research will be chosen. Finally, a proposal for the main experiment will be given in the final chapter.

# 2 Air Traffic Control and Automation

In this chapter the current state-of-the-art regarding to automation in Air Traffic Control (ATC) will be discussed. Firstly, an introduction to Conflict Detection and resolution (CD&R) will be given as this research will only focus on this task. Secondly, the automation efforts which have been investigated prior to this research and the future expected developments regarding ATC and automation are reviewed. After this, a discussion will be given about the definition of acceptance, system requirements and their needed characteristics. Lastly, some methods in order to measure acceptance are analysed.

## 2.1. Introduction to Conflict Detection and Resolution

Air Traffic Management (ATM) is defined by ICAO[32] as: ”ATM is the dynamic, integrated management of air traffic and airspace including air traffic services (ATS), airspace management (ASM) and air traffic flow management (ATFM) — safely, economically and efficiently — through the provision of facilities and seamless services in collaboration with all parties and involving airborne and ground-based functions.” One of the services provided by ATS is ATC, which has the purpose of expediting and maintaining an orderly flow of air traffic and preventing collisions. These tasks are carried out on three different levels by the Area Control Centre (ACC), the Approach Control (APP) and the Aerodrome Control (TWR). An overview of this structure is given in figure 2.1.



Air Traffic Management (ATM)
Air Traffic Services (ATS)
Air Traffic Flow Management (ATFM)
Airspace Management (ASM)
Air Traffic Control (ATC) Service
Advisory Service
Flight Information Service
Alerting Service
Area Control Service
Approach Control Service
Aerodrome Control Service

**Figure 2.1:** Air Traffic Management overview

As automation in ATC is still in an initial phase[76], this research will focus on the service provided by the ACC. This unit controls en-route traffic within its own control area and has the least external variables compared to the other units. Hence, introducing automation to this control unit is the most straight forward. Within the ACC, ATCo's execute multiple tasks simultaneously. However, CD&R is the most important task within ACC which involves active decision making and is therefore the primary and only concern of this research.

As described by ICAO[32] aircraft have a minimum separation of 1000 ft vertically and 5 nm horizontally in designated airspace under FL 410. A conflict occurs when aircraft are on a trajectory which will violate the minimum separation. When the minimum separation is actually violated by the aircraft one speaks of Loss of Separation (LoS). During CD&R an ATCo predicts the trajectory of an aircraft, identifies potential conflicts and resolves them. As described by Seamster et al.[63] the key issues an ATCo must avoid are, in descending order, violation of minimum separation, deviations from standard operating procedures, disorder that may result in cognitive work overload and making unnecessary requests to pilots.

## 2.2. Air Traffic Control and Automation

In the past, multiple automation systems for CD&R are investigated. In a paper submitted in the year 2000 by Kuchar et al.[37] 68 CD&R modelling methods were already reviewed. In all of these modelling methods a given solution approach to the problem is proposed and exercised, typically through a set of constrained and simplified examples. Kuchar et al. concluded that the majority of the models have multiple concerns which should be addressed in future research. Some of these issues are the effects of uncertainty, inability to handle multiple conflicts, controller acceptance and robustness to degradation or failure. Some more recent efforts are proposed based on multi-agent self-separation (Agogino et al.[3]; Hoekstra et al.[27]; Nguyen-Duc et al.[51]), fuzzy logic (Pineau[33])and Reinforcement Learning (Cruciol et al[13]; Weigang et al.[84]). In the work of Regtuit et al.[56] a strategic conformal automation system using machine learning was introduced. In this study, only conflicts with two aircraft in a two-dimensional horizontal plane were taken into account. Data was also created manually by actively using certain strategies extracted from the dataset afterwards. A first proof-of-concept in the usage of machine learning for achieving strategic conformal automation for CD&R was given in this study. However, the authors stated that further research was needed involving more complex scenarios, data generated by professional ATCo's and more states.

The work of van Rooijen et al.[59] is a continuation on the work of Regtuit et al.[56] of achieving strategic conformance using machine learning. In order for the automation and the controller to work from a shared mental model the algorithm incorporated features which are also used by human controllers. The used parameters for the model were extended to the Closest Point of Approach (CPA), time to CPA, conflict angle, relative velocity, traffic density, traffic complexity and exit waypoint. These parameters were incorporated in a Solution Space Diagram (SSD) as shown in figure 2.2. It is a visualisation of the problem based on the solution space, introduced by Velasco et al.[47]. The SSD will be elaborated on extensively in the next chapter. Van Rooijen et al. used a control sector inspired by Amsterdam Sector South 1 and did a human-in-the-loop experiment using ATCo's. They had to perform the CD&R task during a simulation where the SSD was shown to the controllers while solving conflicts horizontally. All the actions of the controllers were logged and used to train a Convolutional Neural Network (CNN). As input of the CNN the SSD is used, as this is conformal to the tool used by the controller. The output of the CNN is a resolution type (heading, speed or direct to waypoint), direction (left or right) and directional value. An extended elaboration on CNN's, in general, and the CNN used for this algorithm will be given in the next chapter.

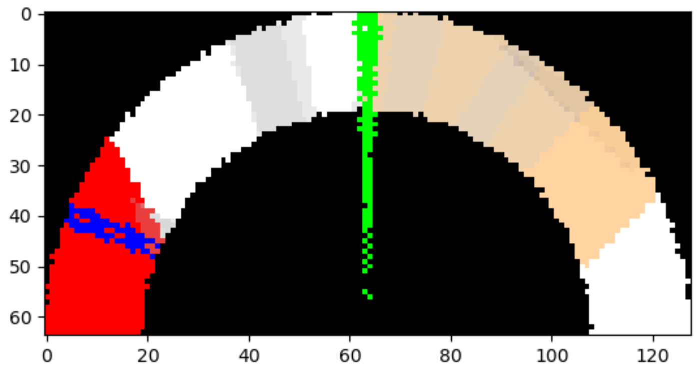


**Figure 2.2:** Example of the Solution Space Diagram. All safe velocity vectors within the flight envelope of the aircraft are shown using white space. If a certain velocity vector results in a LoS, this area will be grey, orange or red depending on $t_{LoS}$. Green is used to visualize the current velocity vector and blue gives the heading towards the goal waypoint.

Currently the world is in the middle of the Covid pandemic meaning the level of air traffic is lower compared to the pre-pandemic situation. Historically, the number of flights has always shown a steady growth with some temporary reduction due to unforeseen situations like 9/11 and the financial crisis of 2008 [52]. After each dip, the number of flights stabilised to its initial steady growth. SESAR JU [76] expects this increase to continue in the long-term from 11 million flight per year in 2018 up to 15 million flight per year in European airspace by the year of 2035. The consequences of this growth are the increase in delays and future air space capacity problems. In the vision described in SESAR's ATM Masterplan, a scalable air traffic managements system will be delivered in the future for both manned and unmanned aircraft which is capable of handling the expected growth. With the main goal of the system being to enable airspace users to fly preferred flight trajectories as cost-efficiently as possible. An increase in automation level and connectivity will digitally transform the current ATM structure. This will result in a more modular and agile system which enables user to use the services regardless of national borders.

As Artificial Intelligence (AI) is progressively transforming the world, it is expected to be a fundamental building block in the transformation of the current ATM system. As stated by SESAR and EUROCONTROL[77] AI has a huge potential in areas where it can reduce human workload or increase human capabilities in complex scenarios. Six accelerators were introduced in the report in order to resolve the remaining key challenges:

1. **Foster data sharing for AI:**
   Access to large shared data sets will give stakeholders the opportunity to maximise the benefits of AI.
2. **Federate an aviation/ATM AI-specific infrastructure:**
   In order to enable data storage, data preparation and facilitate access to computing power an infrastructure should be established.
3. **Develop a new joint human machine system, skills and training:**
   A joint human-AI system combines human intelligence with computational methods will outperform either controller operating in isolation.
4. **Guarantee the safe use of AI:**
   Maintaining or further improving safety in the aviation industry is a notion which always should be considered, also with the introduction of AI.

5. **Master AI to remain cyber resilient:**
   The introduction of AI new risks when it comes to cybersecurity, this risks should be investigated in order to be resilient to cyber-attacks.
6. **Build up an inclusive AI aviation/ATM partnership:**
   In order to share learned lessons and guidelines based on best practices an AI community should be established for the exchange of data.

This research contributes to the third accelerator, the development of a new joint human machine system by giving ATCo's a greater understanding of the actual system.

## 2.3. The Acceptance of Automation

As argued by Majumdar et al.[46] the airspace capacity is limited by the workload of its controllers. Higher level of automation in order to reduce the workload of ATCo's will be inevitable in the near future, if the level of air traffic is to keep . However, as concluded in the work of Bainbridge[6] automation can also have negative effects on the performance of the controller. These effects could be over-reliance, reduced alertness, degradation of human motor skills and short-term workload peaks. These considerations should be taken into account when designing a new automation system in order to prevent misuse, disuse or total rejection of the system. Nonetheless, automation only reduces workload if it is used in the correct way and accepted by its user. In a study conducted by Bekier et al.[7] it was shown that ATCo's accept automation less if it changes their role fundamentally, e.g. preventing them from formulating decisions regarding CD&R. It was also concluded that the levels of automation that will be accepted are at a fairly low level. Therefore, it will be desirable, especially due to the high impact of failures, for the automation to be an advisory decision support tool.

According to Hillburn et al.[26], the acceptance of a newly implemented automation system by an ATCo is one of the greatest hurdles which has to be overcome. As stated by Lee et al.[42] people tend to rely on automation they trust and tend to reject automation they do not. Hence, an automation system will only be successful if the user has enough trust in the system. In a literature review conducted by Westin et al.[85] the following was concluded: ”1) trust in automation develops over time as a result of prolonged experience, 2) acceptance and operator performance decrease when the authority and autonomy of automation increases and 3) acceptance and operator performance benefit from automation actively involving the operator in the control and decision making loop.” However, a paradox can be found when taking into account initial acceptance. As stated in the same work: ”an operator might only develop trust after using a system, but might also be unwilling to trust a system he/she has not used.” Furthermore, Hoff et al.[30] concluded that trust, and thus also acceptance, fluctuates based on the operator's understanding of the methods an automated system uses to perform tasks. In order for a controller to understand the functioning of a system it should function as expected and be interpretable. As a strategic conformal CNN is considered in this research, the expectation is directly related to predictive performance, also known as accuracy, of the algorithm. Unfortunately, a CNN can be considered a black-box system and is therefore not interpretable by itself. Therefore, the output of the system should be presented in such way that the reasons behind a certain decision becomes understandable. Figure 2.3 shows the relations between all definitions related to acceptance.

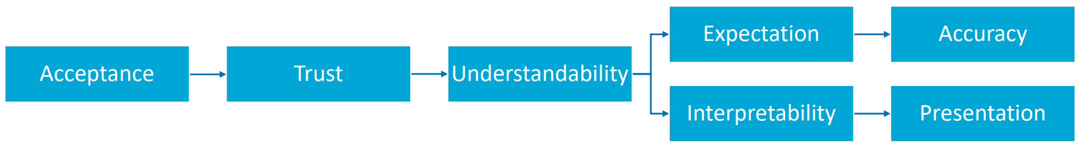


**Figure 2.3:** In order for a controller to accept automation he has to trust it which will only be achieved if he understands the functioning of the system. In order for automation to be understandable it should behave as expected present a solution which will be understood. In the context of this research, this means that expectation is directly related to predictive accuracy and interpretability to the presentation of the solution.

The primary goal of this research is to maximise the acceptance of a CNN used for CD&R by increasing the understandability. This is done by maximising the accuracy and presenting the output in such way that the system is easily interpretable.

## 2.4. Measuring Acceptance

Acceptance within this research should be defined before it can be evaluated. The automation system will give a certain resolution action advisory which the controller can accept or reject. A distinction has to be made on whether the system advises a resolution which successfully solves the conflict or not. Therefore, acceptance will be defined throughout this research as follows: *'An advisory decision support tool is accepted when the controller adheres to the advise given by the tool when a successful resolution advisory is presented'.*

Murdoch et al.[50] introduced a framework which helps with selecting and evaluating interpretability methods. They introduced the PDR framework consisting of three desiderata which should be considered when selecting a interpretation method for a particular model: predictive accuracy, descriptive accuracy and relevancy. Predictive accuracy is a notion which is highly researched in the field of machine learning and can easily be measured using the test set accuracy for which a performance measure will be defined in the next chapter. Descriptive accuracy and relevancy are concepts which are not directly measurable and are highly influenced by the user of a system. In the context of interpretation, Murdoch et al[50] defines descriptive accuracy as: "*the degree to which an interpretation method objectively captures the relationships learned by machine learning models.*" The concept of relevancy is described as the ability to provide insight for a particular audience into a chosen domain problem. The main difference between these two concepts is to what they relate, i.e. the data or the person controlling the system. Descriptive accuracy and relevance should be considered using domain knowledge when designing a system and can only be evaluated using a human-in-the-loop experiment as they are subjective concepts. When evaluating these concepts a distinction between them can not clearly be identified as both concepts influence each other. Throughout this research, interpretability evaluation will refer to evaluations of a combination of predictive accuracy and relevance. It is common practice to evaluate interpretability in a subjective way using questionnaires during or after a such an experiment. Nevertheless, a method to indirectly quantify interpretability is introduced in the work of Poursabzi et al.[54]. An experiment was conducted where participants had to estimate house prices after details about a particular house were shown. After this, the prediction from a model was shown and participants were given the chance to update their initial prediction. From this the Weight of Advice (WoA) was calculated which effectively is the willingness of an operator to change his answer after an advise is given. The WoA is defined as:

$$WoA = \frac{|u_2 - u_1|}{|m - u_1|} \tag{2.1}$$

where m, $u_1$ and $u_2$ represent the model prediction, the initial prediction of the human and the prediction of the human after the advice is given respectively. When the human is not willing to change his belief after been given an advice the WoA will be equal to zero. In the case where the operator chooses to completely grant the advice given the AoW will be equal to one.

Measuring the level of acceptance using the WoA is only possible when a human-in-the-loop experiment will be conducted with the final model. Given the current Covid situation and the limit time frame in which this research has to be completed, it is unlikely that such an experiment will be conducted. Therefore, the WoA will only be proposed as a method to measure acceptance for further research. During this research, it will be assumed that acceptance will increase if predictive accuracy increases and the solution of the model will be presented in such way that it supports interpretability.

# 3 Convolutional Neural Networks for Conflict Detection & Resolution

This chapter will elaborate on CNN's and analyse how to use them for CD&R. First, an introduction to CNN's is given to present a basic understanding and to inform about the state-of-the-art. After this, an explanation about the SSD will be given and how it has been used as input to a CNN for CD&R. Finally, some approaches in order to increase the accuracy related to CNN use and feature engineering will be discussed.

## 3.1. Introduction to Convolutional Neural Networks

The concept of giving machines the ability to carry out tasks which humans would consider 'smart' is defined as AI. Within AI multiple applications exist, however machine learning has experienced the greatest technological advancement in recent years. Due to these advancements, both definitions are often used interchangeably which can cause confusion. Machine learning is defined in the book of Mitchell [48] as: "The study of computer algorithms that allow computers to automatically improve through experience." Within machine learning three principal types are identified by Mitchell [48] as visualized in figure 3.1. In Supervised learning, data is fed to the system to predict a label which will be compared to the given label. The system learns to apply the correct label to a piece of data also when it has not seen the particular data before. With unsupervised learning, labels are not given to the system but the tools for understanding the properties of the data. It is able to cluster data into groups if a great amount of data is fed to the system. Lastly, reinforcement learning is a technique where a system learns a sequence of decisions based on a reward function. The algorithm explores different decisions in order to maximise the result.

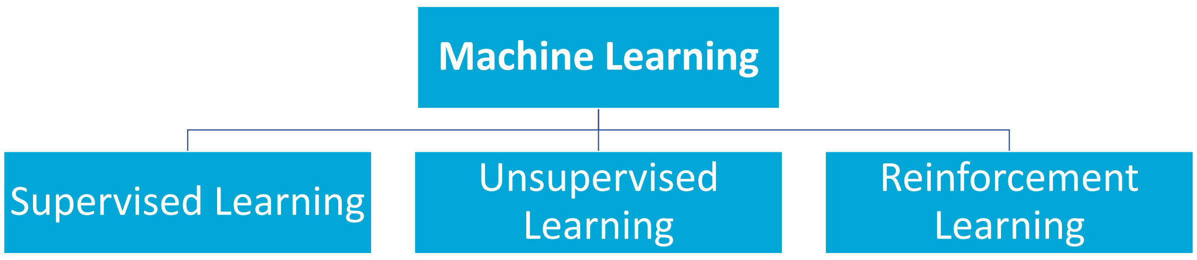


**Figure 3.1:** Categorisation of the three machine learning techniques and their subcategories. Note that multiple taxonomies for machine learning algorithms exist.

The algorithm used in this research intends to reproduce the strategy of an individual ATCo which is done by training, using data and labels coming from actual ATCo decisions. Therefore, the system is a supervised learning algorithm using a CNN. Neural networks (NN) are the most often used tool within supervised learning especially useful for image classification [78, 38]. This section will give an introduction into CNNs by first explaining the basics of NNs. From these basics, an explanation will be given on CNs and the architecture of such networks. Larger neural networks are defined as Deep Neural Networks (DNN), they have shown great advancements in recent years which are discussed at last [36].

### 3.1.1. Neural Networks

The foundation of all neural networks is the neuron which was originally introduced as a perceptron by Rosenblatt [61] in 1958. A schematic representation of perceptron is given by figure 3.2. As can be observed in formula 3.1 the perceptron consists of a weighted sum of inputs $x_n$ with weights $w_n$ to which a bias b is added. This summation is fed to an activation function $\phi$ which calculates the output $y$.

$$y = \phi\left(\sum_n w_n x_n + b\right) \tag{3.1}$$

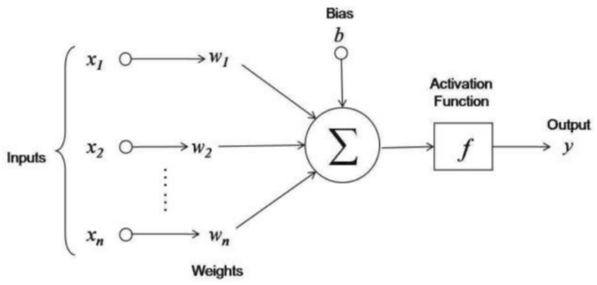


**Figure 3.2:** Schematic representation of a perceptron [61].

When a neural network is ’learning’, it is actually updating the weights and bias on every iteration. As an activation function enables the neural network to have a non-linear relation between input and output it has to be selected with great care. Many different activation function exist but only a subset of them are widely used as argued by Sharma [66]. These functions are the Binary Step Function, the Linear function, the Sigmoid Function, the Hyperbolic Function, the Rectified Linear Unit (ReLU), the Swish function and the SoftMax function for which the mathematical representations are given by equations 3.2 through 3.8 respectively.

$$\phi(x) = \theta(x) \tag{3.2}$$

$$\phi(x) = x \tag{3.3}$$

$$\phi(x) = \frac{1}{e^{-x}+1} \tag{3.4}$$

$$\phi(x) = tanh(x) = \frac{2}{e^{-2x}+1} - 1 \tag{3.5}$$

$$\phi(x) = max(0, x) \tag{3.6}$$

$$\phi(x) = \frac{x}{1-e^{x}} \tag{3.7}$$

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e_{z_k}} \tag{3.8}$$

By staking multiple neurons on top of each other a layer is created. A neural network always exists of at least three layers which are the input layer, the hidden layer and the output layer. A network with more then three layers has multiple hidden layers. A schematic representation of a neural network with three layers is given by figure 3.3.

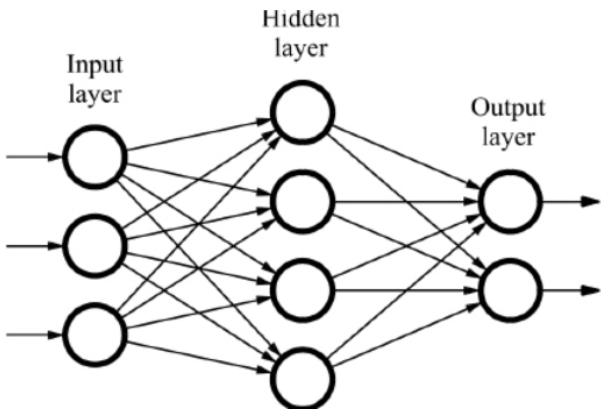


**Figure 3.3:** Schematic representation of a basic neural network.

In order for a NN to train, the output of the network, with the current weights, is calculated first. This output is compared to the corresponding label and an error is calculated. Multiple metrics are defined for this matter, one of which introduced in the work of Rumelhart et al. [62] and shown in equation 3.9. In this equation, c and j represent an index over cases and output units respectively and y and d represent the output state and the desired state respectively.

$$MSE = \frac{1}{2}\sum_{c}\sum_{j}(y_{j,c} - d_{j,c})^2 \tag{3.9}$$

When the error is calculated, the weights are updated using the back propagation. The error is propagated backwards through the network up to the input layer in order to calculate each parameter's influence on the error. Multiple back propagation algorithms exist, the simplest form of which is standard gradient descent introduced by Rumelhart et al.[62]. The partial derivative of the error to each weight $\frac{\partial E}{\partial w_n}$ is calculated for each individual output. In order to update the weights the partial derivative is multiplied with the learning rate which is then subtracted from the current states. The learning rate is a hyperparameter used to regulate the speed of convergence. A so called epoch is completed when all the data is evaluated and all the weights are updated. Usually a network takes many epochs to fully converge.

Many variations of the optimisation algorithm described above exist. These variations differ in how the backpropagtion is carried out and ow the learning rate is upated dynamically. A popular variation to back propagation is stochastic gradient descent (SGD) introduced in the work of Bottou [10]. Instead of finding the partial derivatives to all outputs only a stochastic subset is selected. This has proven to decrease computation power needed and with it the time for convergence. Other more complex optimisation algorithms are momentum (Sutton[70]), Adagrad (Duchi et al.[18]), RMSprop (Tieleman et al.[72]) and Adam (Kingma etal.[35]).

### 3.1.2. Convolutional Neural Networks

As stated above, neural networks are particularly useful when it comes to image classification. This is done using a CNN which is a neural network with at least one convolutional layer. It was firstly introduced in the work of LeCun et al.[41] where a CNN was used to recognise handwritten zip codes. The input to a CNN is most often one in case of gray scale images. For colorized images three matrices are needed one for each RGB channel of the images. Each pixel value is connected to the first layer of neurons. At a convolutional layer a filter shifts over the image in order to obtain a new feature map. The weights inside the filter will be optimised in order to extract certain features from the image. A basic example of the convolution operation is shown by equation 3.10.

$$\overset{\text{Input}}{\begin{bmatrix} 0 & 1 & 3 \\ 4 & 5 & 6 \\ 6 & 7 & 8 \end{bmatrix}} * \overset{\text{Filter}}{\begin{bmatrix} 0 & 1 \\ 2 & 3 \end{bmatrix}} = \overset{\text{Output}}{\begin{bmatrix} 24 & 31 \\ 38 & 44 \end{bmatrix}} \tag{3.10}$$

Multiple parameters have to be set in a convolutional layer:

- **Filter size:**
  When using a larger filter more pixels of the original image are taken into account for the calculation of each convolution. The filter size also influences the output size as can be seen in the example of equation 3.10. It is common practise to use a square filter of odd-number dimension
- **Depth**:
  The depth of a convolutional layer correspond to the number of filters used in parallel in one layer. With a higher depth more features are extracted in that layer.
- **Stride:**
  The stride corresponds to the step size as the filter slides over the input matrix image, e.g. with a stride of two, the filter moves two pixels at each step skipping one pixel. This will influence the size of the output quit drastically.
- **Padding:**
  As said above a convolutional layer reduces the size of the input image. As this can be undesirable in some situations padding can be added to the borders of a matrix. In most cases these pixels will have a value of zero. Padding is also used when the pixels at the edges of the images are important.

Besides convolutional layers, subsampling layers and fully connected layers can be found in a CNN. A subsampling layer, which is often also referred to as a pooling layer, is used to reduce the size of an image. This layer can also be described as a convolutional layer where the filter size and stride are set in such a way that every pixel is only used once. A basic example of such operation is given in equation 3.11 where an 8x8 sized matrix is reduced to a 2x2 sized matrix using a 2x2 filter size and a stride of 2. However, the difference between a subsampling layer and a convolutional layer is that no weights are optimised. Multiple variations to a subsampling layer exist, the max-pooling and the average-pooling are used most often. Both variations can be observed in figure 3.11. The fully connected layer is often found close to the end of a neural network. As multiple convolution and subsampling layers are used, the output will have the form of a long array of 'matrices' with size 1x1 each. Each of these 'matrices' are essentially a neuron containing the probability of a certain feature. In a fully connected layer every output is connected to every neuron. The weights of this determine whether the probability of a certain feature contribute to the probability of a certain output. For example, the output which represents the probability of an image containing a cat has high probabilities on features like fur, whiskers, a tail and possibly more features which define a cat.

$$\overset{\text{Max-pooling}}{\begin{bmatrix}9 & 7\\ 8 & 8\end{bmatrix}} \Longleftarrow \overset{\text{Input}}{\begin{bmatrix}0 & 4 & 7 & 4\\ 9 & 3 & 3 & 2\\ 5 & 4 & 6 & 1\\ 3 & 8 & 9 & 8\end{bmatrix}} \Longrightarrow \overset{\text{Average-pooling}}{\begin{bmatrix}4 & 4\\ 5 & 6\end{bmatrix}} \tag{3.11}$$

By adding different layers of all types together a CNN architecture is created. In figure 3.4 the architecture of the LeNet-5 network, which was introduced in the work of LeCun et al.[39], is presented as an example. As can be seen in the figure it uses six layers in total with two of each type. The architecture of the CNN used in this research is a variation to the network shown in figure 3.4. In this research a variation on LeNet-5 is used which will be discussed more extensively in the next section.

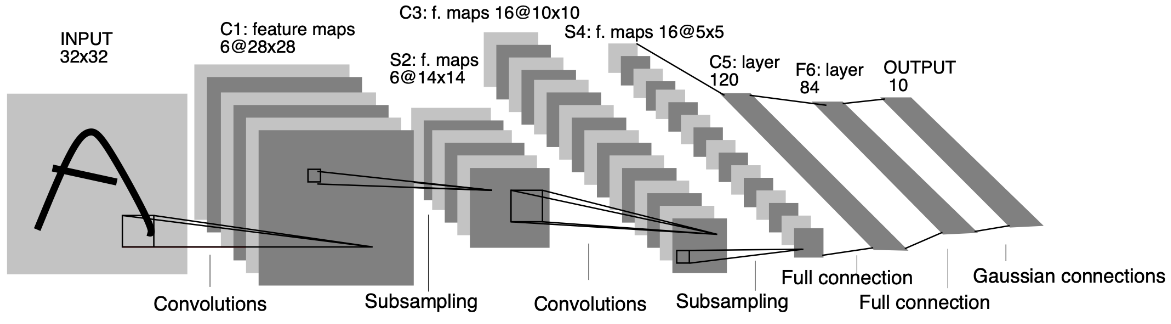


**Figure 3.4:** Architecture of LeNet-5 [39], a CNN used for digit recognition.

### 3.1.3. Deep Neural Networks

Larger neural networks which contains a large number of layers and a high layer depth are often referred to as Deep Neural Networks. larger CNN's, especially have proven to be an excellent tool for almost all recognition and detection tasks as argued by LeCun et al.[40]. At the base of this revolution stands the LeNet-5 architecture of Lecun et al.[39] which was used for document reading. Around the 1990s experiments where done using neural networks for speech recognition (Waibel et al.[83], localising faces (Vailant et al.[78]) and actual face recognition (Lawrence et al.[38]). From the year 2000 onwards neural networks became more successful at detection, segmentation and recognition tasks. A few examples of these tasks are pedestrian detection (Sermanet et al.[65]), biological segmentation from microscopical images (Ning et al.[20] and Turaga et al.[74]), traffic sign recognition (CireşAn et al.[12]), localization of joints (Tompson et al.[73]) and autonomous off-road driving (Hadsell et al.[24] and Muller et al.[49]). Even though, these networks have proven to be very accurate they still are not considered DNN's. Only in recent years CNN architecture have become very large in order to increase performance on object recognition. In the work of Kriszhevsky et al.[36] 1.2 million high-resolution images where classified into 1000 different classes. The training of the network took 5 to 6 days using 2 high end GPUs optimising the 60 million trainable parameters distributed over 650,000 neurons.

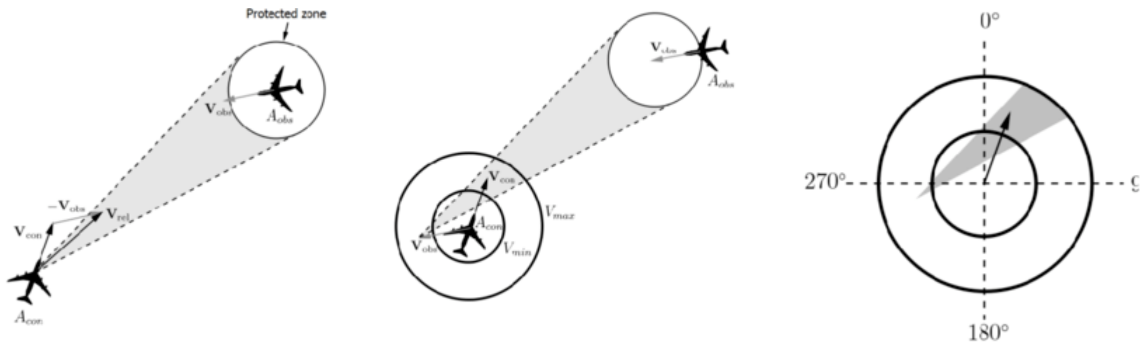
## 3.2. Conflict Detection & Resolution with Convolutional Neural Networks

As stated in the introduction, van Rooijen et al.[59] implemented a CNN which predicts a resolution given a SSD as input. In order to understand how this is done, the characteristics of the SSD will be explained first. After this, the algorithm used by van Rooijen et al[59] will be elaborated on.

### 3.2.1. The Solution Space Diagram

The SSD is a visual support tool which incorporates multiple parameters applicable to the CD&R task. In recent years, it has been used for various applications but was first introduced in an aerospace context by van Dam et al.[80] as an self-separation tool. After this, researche have been conducted into the effect of the SSD on workload. Hermes et al.[25], d'Engelbronner et al.[14] and Mercado Velasco et al.[47] all investigated the correlation between controller workload and several solution-space properties or using the SSD. It was introduced for the first time as an input to an automated system by van Rooijen et al.[59]. This allows the automation and the controller to work from a shared mental model.

Figure 3.5 gives a representation of how the SSD is constructed. The construction starts by calculating the Forbidden Beam Zone (FBZ) which was introduced in the work of Visser et al.[82]. Considering aircraft A and B, the FBZ of aircraft A can be constructed by drawing two lines originating at its own position to the edges of the protected zone of aircraft B. As can be seen in figure 3.5.a this creates a zone in which the relative velocity of aircraft A to B results in LoS in the near future. Shifting the FBZ with the velocity vector of aircraft B defines a zone which results in LoS if the absolute velocity vector of aircraft A ends within this zone as can be seen in figure 3.5.b. After this, two circles are drawn around the aircraft representing the minimum and maximum velocity. Adding the velocity vector of aircraft A the SSD as presented by figure 3.5.c is constructed. When the velocity vector ends in the area enclosed by the FBZ it is on a trajectory which will result in LoS. The space between the velocity limitations and outside the FBZ is referred to as the solution space. Multiple FBZ's can be present inside the SSD if there is a situation with multiple intruding aircraft. Note that the SSD is limited to horizontal resolutions currently. This is not a concern for this research however as also only horizontal resolutions are considered.



(a) Conflict geometry in relative space

(b) Conflict geometry in absolute space

(c) Solution space diagram for the controlled aircraft $A_{con}$

**Figure 3.5:** The construction of the Solutions Space Diagram [59].

The SSD gives the controller a clear overview of the situation and helps in finding an appropriate resolution solution. The following parameters are presented visually by the SSD:

- **Closest Point of Approach (CPA):**
  Using the translation of the FBZ the CPA can be visualised.
- **Time to CPA ($t_{CPA}$):**
  The $t_{CPA}$ is represented by the width of the FBZ, a large $t_{CPA}$ results in a narrow FBZ. Optionally, color coding can be used for visualising $t_{CPA}$. Within a FBZ different $t_{cpa}$s can be presented as it is collection of velocity vectors.
- **Relative velocity:**
  This can be visualised using the tip of the FBZ and the velocity vector. The vector between these points is the relative velocity. Alternatively, insight in this can be obtained by the rate of increasing with of the FBZ.
- **Conflict angle:**
  The conflict angle is visualised by the heading of the center line of the FBZ.
- **Traffic density:**
  All other aircraft appear inside the SSD with a separate FBZ.
- **Traffic complexity:**
  The amount of space which is not covered by a FBZ is considered to be the solution space. The traffic complexity is represented by the free solution space.
- **Exit waypoint:**
  Optionally, the heading towards the exit waypoint can be visualised using a symbol on the SSD.

### 3.2.2. Past Effort of Convolutional Neural Network for Conflict Detection & Resolution

The algorithm van Rooijen et al[59] used is trained on a dataset consisting of RGB images of SSDs with size of 128x128 pixels. The velocity vector is visualised using a green vector and the exit waypoint is indicated with a blue marker. Furthermore, the FBZ's are colour coded as red ($t_{CPA} < 60s$), orange ($60s < t_{CPA} < 120s$) or gray ($T_{CPA} > 120s$). The images are rotated such that the velocity vector is always orientated upwards. After this, the lower half of the images is deleted as turns larger than 90 degrees are highly unlikely and not desirable. This results in a 64x128 pixels image which will be down sampled to 32x64 pixels to decrease computational load. The image is fed to the algorithm in RGB format, each pixel representing the value for red, green or blue. In this way, the solution space, the velocity vector and the exit bearing are incorporated separately. Figure 3.6 shows the SSD as it is fed to the system in its complete form and the three colour channels. The architecture used for this algorithm is a variation on the LeNet-5 architecture (figure 3.4) introduced in the work of LeCun et al.[39]. Table 3.1 includes the total architecture of the algorithm implemented by van Rooijen et al.[59].

**Table 3.1:** Architecture of used network.

| Layer type | Input size | Filter size | Stride | Filters | Activation function | Output size |
|---|---|---|---|---|---|---|
| Convolutional | 32x64x3 | 2x2 | 1 | 32 | ReLU | 31x63x32 |
| Max-Pooling | 31x63x32 | | | | | 15x31x32 |
| Convolutional | 15x31x32 | 2x2 | 1 | 64 | ReLU | 14x30x64 |
| Max-Pooling | 14x30x64 | | | | | 7x15x64 |
| Convolutional | 7x15x64 | 2x2 | 1 | 32 | ReLU | 6x14x32 |
| Flatten | 6x14x32 | | | | | 2688 |
| Fully Connected | 2688 | | | | | 1024 |
| Dropout | 1024 | | | | | 1024 |
| Fully Connected | 1024 | | | | SoftMax | 2 or 3 |

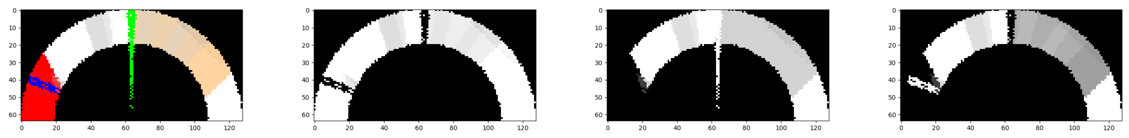


**Figure 3.6:** The SSD as it is fed into the algorithm and the three separate color channels. From left to right the three separate color channels are red, green and blue.

During a human-in-the loop experiment all the actions a controller executed and the related SSD images were logged. This data set was used to train three personalized CNN's for each controller using part of the data set for validation and testing. Each model predicts one of the three decision stages which are resolution type, direction and directional value. The resolution type prediction can be heading (HDG), speed (SPD) or direct to waypoint (DCT). Secondly, the direction prediction can be left or right in the case of a heading command or faster or slower in the case of a speed command. Lastly, directional value is divided into three classes being 0 to 10 degrees, 10 to 45 degrees and larger than 45 degrees for a heading command. For a speed command it gives the command relative to the current state. A complete overview of how the three models together come up with a prediction is presented by figure 3.7. The construction of the CNN's was done using the Keras and TensorFlow libraries in Python. The training for all networks was done with identical hyper parameters which are presented in table 3.2.

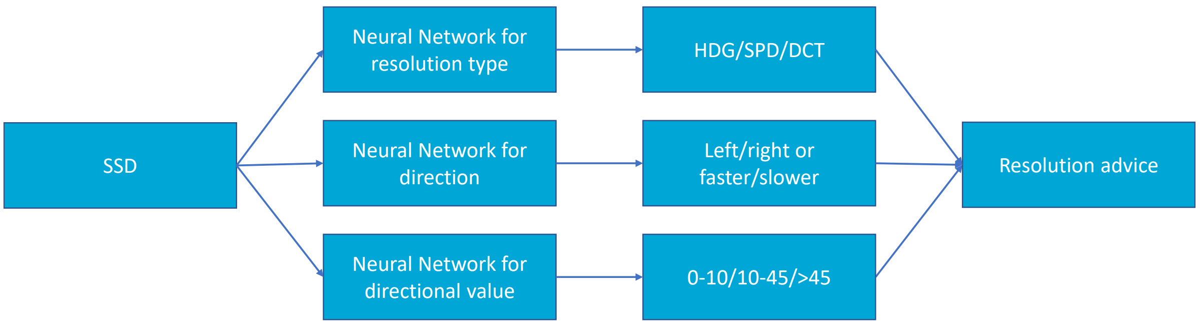

SSD
Neural Network for resolution type
Neural Network for direction
Neural Network for directional value
HDG/SPD/DCT
Left/right or faster/slower
0-10/10-45/>45
Resolution advice

**Figure 3.7:** Fundamental model architecture used by van Rooijen et al. [59]

**Table 3.2:** Hyperparameters used for training by van Rooijen[59].

| Hyperparameter | Value |
|---|---|
| Optimization algorithm | Adam |
| Output activation function | SoftMax classifier |
| Loss function | Categorical entropy |
| Train/validation/test ratio | 60/15/25 |
| K-folds | 5 |
| Mini batch-size | 32 |
| Steps per epoch | 2 x training samples / batch size |
| Epochs | 30 |
| Learning rate | 0.01 |
| Dropout rate | 20 |
| Input image dimensions | 32x64 |

## 3.3. Opportunities for algorithm optimization

Van Rooijen et al.[59] did an initial attempt to use a CNN as an advisory system for CD&R with the SSD as input. It was stated that, further research is needed refining model architecture and optimizing the input to the system. This section will discuss these opportunities for improving the algorithm in terms of accuracy. First, an analyses will be given on the optimal use of the current architecture. After this, an introduction to feature engineering will be given and some possibilities for accuracy optimization related to feature engineering will be discussed. The accuracy of a certain system has to be measured using a suitable metric. Therefore, the final subsection will define accuracy and a suitable metric given the problem and system researched.

### 3.3.1. Use of architecture

In the initial attempt, three networks are used used in order to predict a suitable resolution strategy. In order to do this bins were defined which were used for classification. The output from each network together form the resolution advice given to an ATCo. For example, an heading change to the left of 30 degrees will be shown as: $HDG \rightarrow LEFT \rightarrow [10, 45]$. One could argue that using this structure is sub-optimal due to multiple reasons:

- **Independent networks:**
  The three separate CNN's are used separately for different tasks without any connection between them. Nevertheless, the task for the direction and directional value changes due to the output of the resolution type network. For example, the direction network predicts left or right when the output of the resolution type network is a heading change. When the resolution type is a speed change the direction network uses the same weight to predict faster or slower. As there is no connection between them, the direction network can not make a distinction between what type of output is predicted. This can result in error propagation if the resolution type network gives a false classification
- **Bin size:**
  Another sub-optimal characteristic of this algorithm are the large bin sizes in which an advise is given. The system advises a bin which is an assemble of multiple possible velocity vectors. Even if the systems output is considered correct there can be velocity vectors in the bin which result in a LoS. Therefore, a controller should still derive a solution and search for a save velocity vector. Therefore, one could argue that an advisory system with such large bin size does not result in a reduced workload for ATCo's.
- **Direct to type resolutions are redundant:**
  Often, during CD&R conflicts are solved using two resolutions. The first resolution is a speed or heading command to solve the conflict after which a direct to command will be used when a direct path to the exit waypoint is free. As stated earlier, the heading leading to the exit waypoint is incorporated within the original design of the SSD. It is straightforward for an ATCo to detect whether the exit waypoint is laying inside the free solution space or not. Therefore, the prediction of a direct to resolution is redundant and can possibly lead to a sub optimal performance.

ATCo’s give commands to pilots in specific steps, i.e. heading change commands are done with a minimum of 10 degrees. Despite this, aircraft fly in continuous space and the FBZ’s presented inside an SSD are not bounded by any bins. Therefore, the output of the CNN should ideally also be continuous in the form of a singe velocity vector. Van Rooijen et al.[59] used the CNN architecture for classification with predefined bins. This is a logical choice as CNN’s have shown to be particular successful with image classification. However, CNN’s can also be used for other tasks than classification. In the work of Fischer et al.[21] a regression type of approach was used in order to predict image orientation. Another example of using a CNN for regression is presented in the work of Mahendran et al.[45]. They constructed a CNN which predicts the 3d pose of an object in an image. Both papers conclude that CNN’s can be used for regression and show competitive performance compared to classification given their problem. As a single velocity vector can be constructed using only a number for heading and speed, a CNN should be able to predict a resolution using regression. This can be done by using only one network which has two outputs for heading and speed. Within machine learning the inputs and the labels need to be normalized between 0 and 1 in order to converge. This will mean the output of the network consists of two values between 0 and 1. These values will have to converted to to the heading range of -90 to 90 and the speed range equal to minimum and maximum speed of the aircraft.

### 3.3.2. Feature Engineering

With the development of machine learning algorithms, preparing the data is found to be the most time consuming activity. According to Dasu and Johnson[16], 80% of data analysis is spent on the process of cleaning and preparing the data. This process is often referred to as feature engineering. Despite the time consuming aspect, very little research on the topic is published in a formal way. According to Locklin [44] feature engineering is another topic which does not seem to merit any review papers or books, or even chapters in books. Also, feature engineering is a term used in the domain of software engineering to which most research papers relate. nevertheless, it is a critical aspect for machine learning algorithms. As argued in the work of Domingos [17] the most important factor for the success of a machine learning project are the used features. Raw data is often not optimal for learning algorithms but from this data features can be created that are. Additionally, feature engineering is the only phase in machine learning in which domain knowledge of the problem can be taken into account.

A team of students from the National Taiwan University have shown that using domain knowledge for feature engineering can have exceptional effects on the performance[87]. They participated in the KDD Cup 2010 which is a data mining competition where teams have to predict student algebraic performance. Prediction of the performance was done using two data sets which contained logs for a large number of interaction steps with intelligent tutoring systems. Some of the techniques used were binary data tables for categorical data, scaling for numerical data and identifying feature pairs. The extensive use of feature engineering enabled the team to use only a simple linear regression ensemble. With this method, the team achieved a higher performance compared to other teams, which all used a more elaborated algorithm, and therefore won the competition.

Within the domain of machine learning a feature is defined as an individual measurable variables that is used as an input to a system. Feature engineering aims to optimise these variables in order to make them compatible with the system and to improve the performance. A clear definition is not given in most literature and multiple variations exist. The definition of feature engineering used in this research is the same as used by Brownlee[11]: ”Feature engineering is the process of transforming raw data into features that better represent the underlying problem to the predictive models, resulting in improved model accuracy on unseen data.”

Feature engineering can be seen as a broad discipline in which a diverse range of tasks can be executed. As there is little documentation on the topic, a framework which defines the procedure of feature engineering does not yet exist. Therefore, using feature engineering in a successful way is solely dependent on the domain knowledge of the developer. Manual feature engineering relies on trial and error using multiple techniques which are not necessarily defined by a clear framework. Nevertheless, multiple techniques within feature engineering exist which include mathematical operations or focus more on the selection of features or variables. Some of the techniques which are applicable to CNN’s or this research are listed below:

- **Scaling:**
  One of the most straight forward techniques of feature engineering is scaling. This is done by

normalizing the variables of a data set to a fixed scale. As CNN's use the pixel values of an image as input it is of great importance all images are scaled conformal to each other. Generally, the colour of a pixel is described using values ranging form 0 to 255 but a scale from 0 to 1 is also common in machine learning.

- **Handling outliers:**
  Images often contain noise which is non-optimal for the output of a CNN. These pixels often have a value which differs significantly from its neighboring pixel and can therefore be seen as an outlier. Handling outliers for a CNN is identifying these pixels and assigning a different value.
- **Variable selection:**
  As CNN's are mainly used for image recognition, variable selection often is not applicable. However, the SSD is a figure which is a schematic representation of certain situation. To construct this figure, certain domain knowledge of a particular situation is needed. Using this domain knowledge certain parts of the figure can be identified as more relevant compared to others.

In the experiment done by van Rooijen et al.[59] SSD's were captured as a screenshot from a simulation. These input images are clearly readable for a human controller but are not optimised for using them as input to a CNN. Feature engineering can provide a framework to increase accuracy by changing the input. The main reason to use a CNN for CD&R is that it allows the controller and the automation system to work from a shared mental model. When designing new features, this mental model should be taken into account such that it is not lost. Below three changes to the SSD are proposed in order to increase accuracy:

- **The use of easy distinguishable colors:**
  CNN's identify colours in a different way compared to humans. Humans are able to identify colours directly from the complete image while a CNN breaks it down into three layers, in case of a colorized image, and identifies the values. Table 3.3 provides the pixel value (range) in percentage, over the three colour channels used, to represent the different colours for the SSD shown in figure 3.6. From the table it can be observed that $t_{CPA}$, speed vector and exit bearing are not all effectively incorporated separately in a color channel. Furthermore, it can be observed that the pixel value ranges for white, orange and red are close together. In an ideal situation, the pixel values for a specific colour always have the same value and information regarding $t_{CPA}$, velocity vector and exit bearing are coming from a separate channel.

**Table 3.3:** Pixel values for different colours distributed over the three colour channels.

| Input colour | Colour channels | | | Effective meaning |
|---|---|---|---|---|
| | Red[%] | Green[%] | Blue[%] | |
| Black | 0 | 0 | 0 | Background |
| White | 100 | 100 | 100 | Solution space |
| Gray | 87-91 | 87-91 | 87-91 | $t_{CPA} > 120s$ |
| Orange | 79-97 | 71-83 | 59-73 | $60s > t_{CPA} > 120s$ |
| Red | 100 | 0 | 0 | $t_{CPA} < 60s$ |
| Green | 0 | 100 | 0 | Speed vector |
| Blue | 0 | 0 | 100 | Exit waypoint |

- **Scaling the solution space:**
  The SSD is constructed in such a way that it is conformal to the mental model of a controller. This means the heading is presented in a circular shape and the speed by the distance from the middle. This increases situational awareness and therefore reduces workload. However, a CNN does not have an internal representation of a situation. The algorithm reads a 32x64 pixel image which consists of the capture of a SSD. Each pixel is seen by the algorithm as a variable inside a 32x64 grid which means input layer consists of 2048 pixel variables. In the most optimal situation all these variables should contribute to the calculation of the output. However, approximately 58% of all pixels are part of the background which do not have any relation to the applicable situation and therefore do not affect the solution. This is non-optimal in terms of computational power. A possible solution could be to scale the size of the solution space.

- **Polar coordinates:**
  Another inefficiency of pixel use is the non-linear distribution of velocity vectors over the solution space. The solution space can be converted into polar coordinates which all represent a velocity vector. For example, figure 3.8 shows polar coordinates grouped with a heading separation of 5 degrees and 4 speed groups. It can be observed that the velocity vectors with a lower speed are represented by a smaller area compared to velocity vectors with a larger speed. A CNN reads an image as a cartesian plain of squares of the same area which is inconsistent with the area distribution of the SSD. Hence, the velocity vectors with a smaller speed are represented by less pixels compared to the velocity vectors with a larger speed. This means the variables are non-linear distributed over the velocity vectors and the lower speed solution space is shown in a less precise way when compared to the high speed solution space.

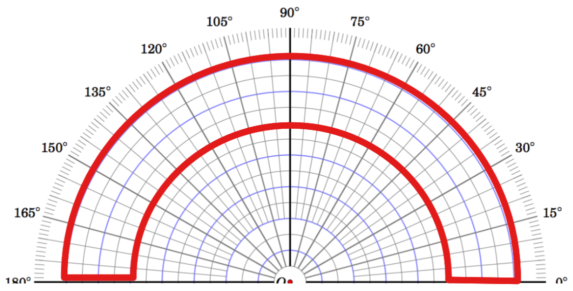


**Figure 3.8:** Polar coordinate representation of a SSD.

## 3.3.3. Defining Accuracy

Before any experiment is done it is important to define what accuracy is in the context of this research. In general, accuracy relates to the condition of quality of being correct. For this machine learning application, accuracy is the number of correctly predicted data points out of an entire data set. In this research accuracy will also relate to the predictive power of the algorithm. Van Rooijen et al.[59] used the Matthews Correlation Coefficient (MCC) as an accuracy measure in order to evaluate the predictive power of the algorithm. It is considered a balanced method without bias to evaluate model performance for classification algorithms. However, such an accuracy metric can not directly be used when a CNN does not solve a classification task. As this research intends to use a regression approach in order to solve the CD&R problem with a CNN, a different accuracy measure has to be used. With the normalized numerical output of the CNN the Mean Squared Error (MSE) which is defined by Fürnkranz et al.[23] is a straight forward accuracy measure which can be used directly. It is defined as

$$MSE = \frac{\sum_{i=1}^{n}(y_i - \lambda(x_i))^2}{n} \tag{3.12}$$

where $y_i$ is the true value for $x_i$, $\lambda(x_i)$ the predicted value for $x_i$ and n the total number of test data points. In contrast to the MCC, a high accuracy is achieved with a low MSE. In order to discriminate between heading and velocity error the MSE can be converted to true heading and velocity error.

# 4 Explainable Artificial Intelligence for Convolutional Neural Networks

AI is becoming more and more a part of the daily lives of humans. The statistics portal Statista forecasts that revenues coming from the AI market will increase exponentially from 4.8 trillion U.S. dollars in 2020 to 31 trillion U.S. dollars in 2025 [68]. Decision algorithms, which use AI, are already a part of our daily life, e.g. personalized advertisement, movie recommendation on streaming platforms, friend suggestions in a social network, etc. These examples have in common that the decisions proposed by them do not affect our lives in a fundamental way. Little harm is done when such an algorithm gives an inaccurate or wrong prediction. However, the use of AI is also increasing in systems where more critical decision are made, e.g. disease diagnosis, military applications, autonomous vehicles and ATC. With all these applications, an inaccurate or wrong decision can seriously harm human beings or even result in loss of life. As argued by Adadi and Berrada [2], when it comes to life-changing decisions it is important to know the reasons behind such a critical decision. Machine learning algorithms in particular are often black-box systems which cannot explain themselves. Therefore, thrusting them with important decisions can result in obvious dangers. Explainable Artificial Intelligence (XAI) intends to mitigate these dangers by improving the interpretability of an AI. Within this research, the XAI definition from Arrieta et al. [5] will be used: ”*Given an audience, an explainable Artificial Intelligence is one that produces details or reasons to make its functioning clear or easy to understand*.” Within this definition the ATCo is referred to as the audience which has to be considered when designing an XAI algorithm.

In the work of Arrieta et al. [5] a distinction has been made between transparent and black-box macine learning techniques. Transparent models are understandable and therefore do not need any post-hoc explaining techniques. As CNN’s are black-box models this research will only focus on post-hoc explaining techniques. Another separation which can be made is between techniques which are specific to a certain model and which are not. Within the framework of XAI a broad variation of techniques exist for different types of models. Figure 4.1 shows all XAI techniques which can be applied to CNN’s. In this research however, only models from which the explanation can be shown in a visual form applicable to CNN’s will be considered. Furthermore, XAI techniques which require to change the original architecture of the CNN will also not be considered. As many XAI methods exist, not all can clearly be subdivided to a particular technique. For example, a particular method can visually explain the relevance of certain features. However, it is possible to make a division on general strategies applicable to CNN’s.

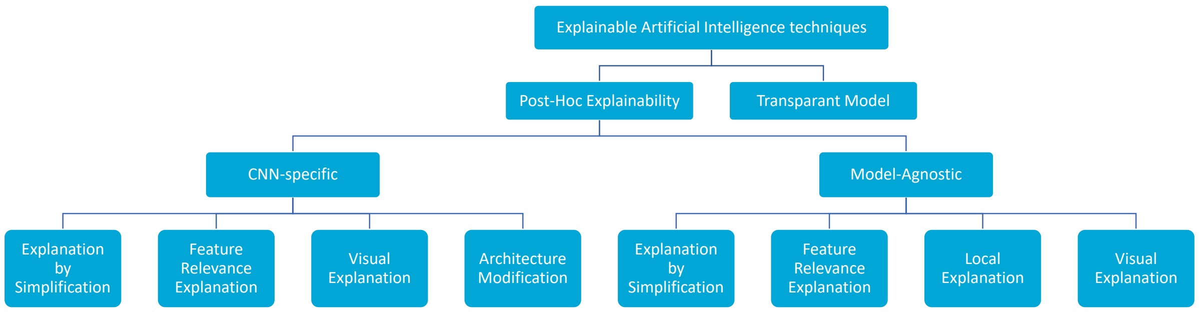

Explainable Artificial Intelligence techniques
Post-Hoc Explainability
Transparant Model
CNN-specific
Model-Agnostic
Explanation by Simplification
Feature Relevance Explanation
Visual Explanation
Architecture Modification
Explanation by Simplification
Feature Relevance Explanation
Local Explanation
Visual Explanation

**Figure 4.1:** XAI techniques for CNN's.

In the next sections, multiple explanation strategies are detailed such as: explanation by example, saliency maps and local models. Each strategy will be explained using a particular method which has shown to have promising performance within its strategy. Next to this, a short discussion will be given on the usefulness of this technique for this research.

## 4.1. Deep k-Nearest Neighbors

One of the strategies to increase interpretability is an explanation by example. In the work of Papernot and McDaniel [53] the Deep k-Nearest Neighbors (DkNN) algorithm is introduced which finds patterns between intermediate results of a particular test item and the training data set. The nearest neighbors inside the training data when a certain test input is fed to the algorithm is found in order to find these patterns. This is done at every layer within the network to confirm that the prediction made is supported by the training data. It enables the user to inspect whether the final prediction is consistent with each intermediate computation. Each hidden layer inside a network calculates a different representation of its input. This representation is fed to the next layer until the final layer outputs a particular class. As the input to the final layer is the output from the second to last layer, the intermediate result of this layer can also be assigned to the same class. This can be back-propagated until the input layer is reached at which point the data can be shown visually as a picture. The nearest neighbors of an input should have the same class throughout every layer of the network. Due to the high-dimensional data of the intermediate results the Locality-Sensitive Hashing function, introduced by Andoni et al. [4], is used for finding nearest neighbors. This function defines a list of candidate nearest neighbors using cosine similarity between vectors.

With DkNN, Papernot and McDaniel [53] aimed to improve the confidence, interpretability and robustness of DNN's. In this research, the main focus will be on the interpretability as experienced by the ATCo. The overall goal is to support the controller in understanding why a certain decision is made by the system. As argued in the work of vailant[79] and Dasgupta et al. [15] locality-sensitive hashing may be a general way of doing computations inside the human brain. In the research done by Stock and Cissé [69] an example of how this works is given using a picture of Barack Obama throwing a football. The picture is wrongly classified as 'Basketball' instead of 'Football' by a residual neural network. Papernot and McDaniel [53] applied the DkNN algorithm to this network in order to explain this classification. Figure 4.2 shows two versions of the input picture and its 10 closest neighbors. Taking a closer look it becomes clear why the algorithm classified the image as 'Basketball'. The color of the football is similar to the basketballs, in 7 of the closest neighbors the player is black and the ball is positioned at the upper part of the image. On the right side the picture is cropped to exclude the football which results in the algorithm predicting 'Racket'. Here it can be seen that the majority of the nearest neighboring pictures include players dressed in white, have a green background and the arm position of the player is above the head. From figure 4.2 it becomes apparent that biases exist within neural networks which can be easily identified by a human using the nearest neighboring training images.

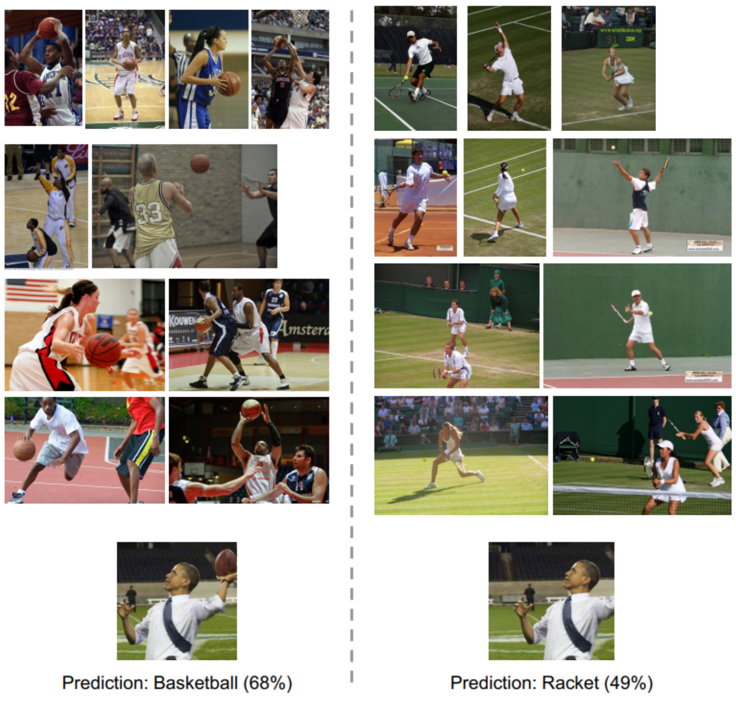


**Figure 4.2:** Incorrect classifications of Barack Obama throwing a football [53].

DkNN is a comprehensible technique which gives the user the tool to clearly discover biases a CNN might have learned. Having knowledge of such bias can help when a user has to decide to accept a certain outcome. However, the parent goal of the system to be designed is to decrease ATCo workload. An explanation will consist of several SSDs which are the most related to the current conflict. This means an ATCo will have to process several SSDs instead of only one. Therefore, explanation by example is not a suitable technique for this research.

## 4.2. Gradient Class Activation Mapping (Grad-CAM)

Another possible technique to give explanations to an ATCo during CD&R could be Gradient Class Activation Mapping (Grad-CAM). This technique, introduced by Selvaraju et al. [64], produces the class-discriminative localization map Grad-CAM $L^c_{Grad-CAM} \in \Re^{u \times v}$ of width u and height v for class c. This map shows the importance of a region of a image for a particular category. With respect to a particular feature map $A^k$ the gradient of the score for a particular class $\frac{\partial y^c}{\partial A^k}$ is computed. These gradients flow back into the network using Global Average Pooling (GAP) which is an operation introduced in the work of Lin et al. [43]. GAP was originally used to replace the fully connected layers of CNN's. With this operation a feature map for each category is generated by the network in the last convolutional layer. The average is taken from these feature maps which is then inputted into the final softmax layer. By applying GAP to the gradients the importance weight $\alpha^c_k$ can be calculated as shown in equation 4.1.

$$\alpha^c_k = \frac{1}{Z}\sum_i \sum_j \frac{\partial y^c}{\partial A^k_{ij}} \tag{4.1}$$

The weight $\alpha^c_k$ indicates the importance of feature map k for a target class c. In order to obtain $L^c_{Grad-CAM}$, a ReLU activation function is used on the weighted combinations of weights for all feature maps as shown in equation 4.2. The Relu is applied as only the features which have an influence on the class of interest are positive. The result of this operation is a heat-map which has the same size as the original convolutional feature maps. A schematic representation of this operation can be found in figure 4.3.

$$L^c_{Grad-CAM} = ReLU\left(\sum_k \alpha^c_k A^k\right) \tag{4.2}$$

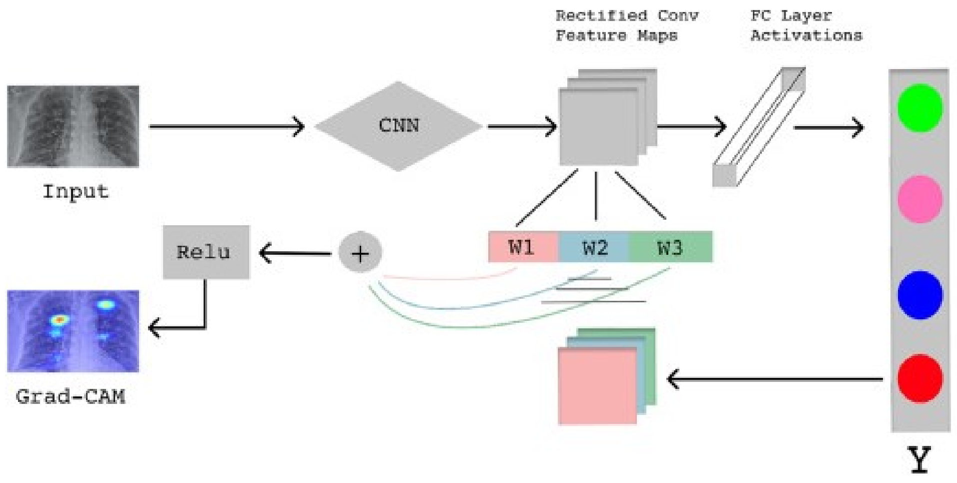


**Figure 4.3:** Schematic representation of the Grad-CAM operation applied to a medical context [64].

Before grad-CAM other techniques which used heat-maps in order to explain the output of a neural network. For example Class Activation Mapping (CAM), introduced by Zhou et al. [88], produces a similar heat map using a different operation. CAM replaces the final fully connected layer with a Gap layer. With this modification the CNN architecture is changed and will learn a feature map in the last convolutional layer. This introduces an additional risk of seriously compromising the performance of the algorithm. Another drawback is the fact that it is only applicable to a particular CNN architecture. CAM can only be applied to an algorithm in which the softmax layer can be directly followed after the last convolutional layer. As the algorithm in this research uses multiple fully connected layers it is not possible to apply GAP without a fundamental change in the architecture. Grad-CAM however, can be implemented with any CNN architecture which makes it a suitable method to give explanation to an ATCo.

## 4.3. Local Interpretable Model-agnostic Explanations (LIME)

A technique based on local explanations was introduced in the work of Ribeiro et al. [57]. They presented the Local Interpretable Model-agnostic Explanations (LIME) technique with the goal of identifying an interpretable model over the interpretable representation that is locally faithful to the classifier. In order to establish an explanation, LIME constructs a local linear model around a prediction of any model. It divides the input into smaller components using contiguous superpixels which are regions of an image which have similar pixel values. More information on this technique can be found in the work of Achanta et al. [1]. These smaller, perturbed instance are fed through the model in order to obtain a data set of probabilities. After this, a linear model is trained on this locally weighted data. The output of LIME is a feature map which presents the superpixels with the largest positive weights. In addition to the visual explanations the probability of a prediction van be given. Figure 4.4 shows an example where the three different predictions for the original picture are explained. The network used to produce this example is Google's Inception Neural network introduced in the work Szegedy et al. [71]. The prediction probabilities for electric guitar, acoustic guitar and labrador are 0.32, 0.24 and 0.21, respectively. For each prediction the explanations shows features which explain the classification in a interpretable way, e.g. the fretboard of the electric guitar prediction.

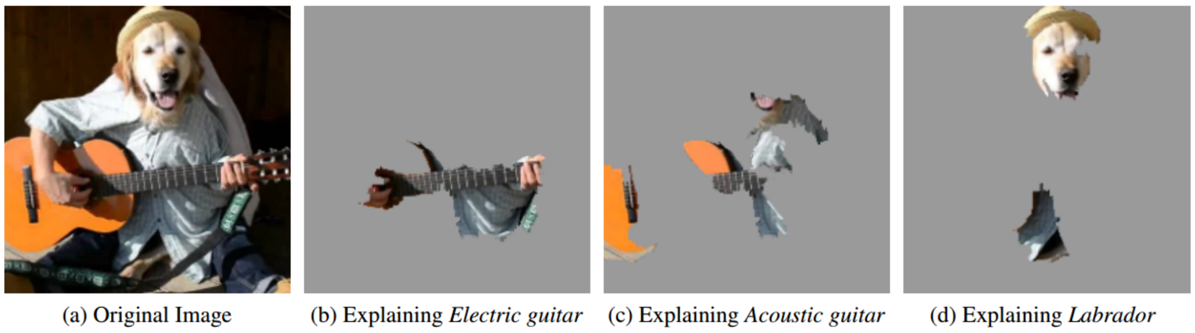

(a) Original Image
(b) Explaining *Electric guitar*
(c) Explaining *Acoustic guitar*
(d) Explaining *Labrador*

**Figure 4.4:** LIME applied to an image of a dog playing a guitar(a) with three different prediction solutions (b), (c) and (d) [57].

One of the advantages of LIME is that it is model-agnostic. The method is able to explain a black-box model without the need for any intermediate results. Only the input and the output are considered for constructing the explanation. Therefore, it is flexible and easy to implement on any model. It can be implemented in Python using the LIME module which consists of multiple functions applicable to any model. This explanation technique can thus be applied without any knowledge of the inner working of it. On the other hand, LIME has to train multiple linear models for the explanation of each prediction. This can be a computationally expensive process when a greater amount of contiguous superpixels is needed. Creating an explanation for one image can take several minutes which makes it inadequate for real-time applications like CD&R.

# 5
# Interface Design

As automation is increasingly applied to more applications, one could argue that the role of human beings in such systems is decreasing to a point that the controller is taken out of the loop. However, as stated by Borst et al. [8], it has become evident that humans continue to be an essential part of technical systems. In the design of complex systems, not all potential problems can be taken into account and therefore they rely on the flexibility and creativity of its controllers. This is one of the ironies of automation described in the work of Bainbridge [6]. Increasing levels of automation tend to distance its controllers further and further from the operations but automation still depends on them in case of an unforeseen situation. Additionally, with the shift to a more supervisory role, the only communication tool between the controller and the system is the interface. This interface should therefore, be constructed in such a way that situational awareness is not lost during operations. As stated by Hollnagel [31]: ”*A good man-machine system should provide the right information at the right time and in the right way.*” However, without a specification of what ’right’ actually means in each specific work domain this definition is useless. In this research, the information from one of the previously discussed XAI techniques has to be presented visually to the ATCo. Therefore, it is of great importance that the ’right definition’ for this research is defined. In the first section, three approaches to human-machine design will be presented and their usefulness for this research will be discussed. In the second section, the framework for designing an interface belonging to the most useful approach will be analysed.

## 5.1. Human-machine Design Approaches

In the work of Flach et al. [22] multiple design approaches for human-machine systems and their implications for what ’right’ means are reviewed. It is argued that the ecological approach is a more comprehensive framework in which the other frameworks can influence it.

### 5.1.1. Technology Centered Approach

A technology centered approach is the most straight forward approach which is often used for less complex systems. It highlights the technological capabilities of the machine and shows the raw sensor values. A basic example is an older car, it only has a few displays like the speedometer, revolutions per minute and the fuel meter. As each displays indicates only one value coming from one sensor also often referred to as the single-sensor-single-indicator approach. In systems with less sensors this is a valid approach however as the systems become more complex the information which has to be processed can be beyond the capabilities of the controller. As argued in the work of Vicente and Rasmussen [81], controllers are only informed about direct sensor data and have to derive higher order domain knowledge themselves. Moreover, the way information is presented in the single-sensor-single-indicator approach is often not conformal to the human perceptual system. In the context of CD&R, multiple states of different aircraft have to be overseen by the ATCo. Perceiving the actual state for a given situation is highly complex when all information has to be derived from numerical values, such as positional coordinates and aircraft attitude. This research intends to design an interface for CD&R with a resolution advice and some explanation. Depending on the explanation, the technology centered approach can be a valid method, e.g. if an explanation only contains the probability of a certain prediction. On the other hand, an explanations which highlights certain pixels from the input image should not be expressed in numerical values.

### 5.1.2. User Centered Approach

In the user centered approach the focus is on the end user, their capabilities and skills, their specific tasks and their preferences. The attitude indicator in a cockpit shows the attitude of the aircraft from the perspective of the pilot and is therefore an example of the user centered design. This reduces the task complexity of the pilot as the individual states do not have to be processed separately in order to obtain situational awareness. In essence, a user centered design addresses the limitations of the human operator in terms of physical, perceptual and cognitive limitations. It reduces the amount of information the controller needs to perceive in order to successfully control a system. However, this approach also introduces some complications. First of all, it gives little to no support in a situation which is unforeseen or unusual. Also, humans interpret systems in different ways and therefore this approach is not always conformal to the mental model of the controller. These complications can be overcome using an additional explanation coming from an XAI technique. However an interface designed according the user centered approach can not give any extra guidance to explain why a certain resolution is chosen. Therefore, the user centered approach is an accurate method for providing guidance in terms of a speed vector within an SSD but it is not suitable for providing explanations.

### 5.1.3. Ecological Approach

As argued by Flach et al. [22] the use centered or ecological approach provides the most comprehensive framework for interface design. It was first introduced in the work of Rasmussen and Vicente[81] and is the only approach which defines the system relative to its work domain. The ecological interface design approach can be described in three steps:

- **Work domain analyses:**
  During the work domain analyses, the functional constraints of a certain system are identified. Examples of functional constraints are the flight envelope of an aircraft, traffic, terrain and weather.
- **Semantic mapping**
  The functional constraints should be visually presented onto the interface in such a way that it is conformal to the mental model of the controller and in the correct coordinate system. For example, in the case of an all engine out emergency an interface could show reachable land using an optimum glide path.
- **Worker competencies analysis**
  In the work of Rasmussen[55] the Skill, Rule and Knowledge (SRK) taxonomy was introduced to define three types of controller behaviour during information processing. The last step in the ecological approach is to identify which behavior is needed for a certain task and design the interface accordingly. For skill based behavior, the interface should show information which triggers fast motor and cognitive responses. For rule based behavior information which triggers familiar actions should be shown. For a knowledge based behavior task, information which supports the controller in solving a new problem for which no rules yet exist.

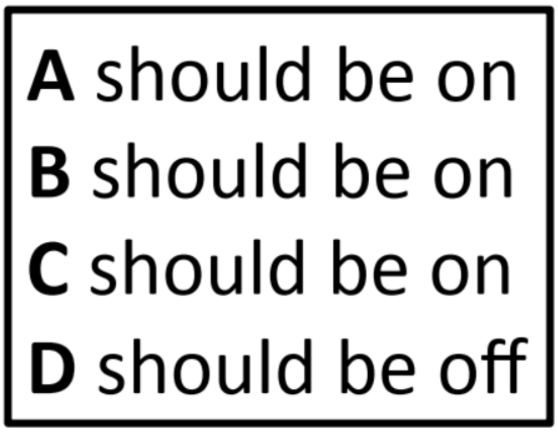
The ecological approach assumes that the actions which should be executed can only be described by a certain state of the work domain. Therefore, it is important to incorporate these states within the interface for the highest situational awareness. Instead of focusing on the technology of the system itself, the technology should be considered as tool to achieve a certain goal relative to the work domain. In this research, an explanation will be given in order for the controller to understand the reasoning of the system. As the SSD gives a representation of the situation and is the input to the system it could be treated as the ’work domain’. An explanation should be given relative to the work domain which consists of a highlighted area in the original SSD image. Moreover, the SSD itself is an interface which is designed according to the ecological approach. As the goal of this research is to add an explanation to this interface, the design approach of the explanation should be conformal to the existing interface.

## 5.2. Basic Principles of Display Design

Usually, an interface consists of one ore more displays which support the controller to perceive the variables of the system. The display is a channel of communication between the system and its controller that should help with the controller's perception and awareness. In the work of wickens et al.[86] 13 human factors principles related to display design are described. These principles can be divided into perceptual principles, mental model principles, principles based on attention and memory principles. In the following subsections, these principles and how they relate to this research will be described.

### 5.2.1. Perceptual Principles

1. **Make displays legible or audible:**
   The first principle relates to the legibility of the display, aspects such as contrast, visual angle and illumination all contribute to this principle. This principle is rather straight forward but of great importance when designing a display. Aspects influencing legibility are mostly a result of the hardware used. In this research the end product will be the design of the interface rather than a physical interface. Therefore, legibility will only be addressed in aspects of the display which can be influenced by the software, e.g. colour use and contrast.
2. **Avoid absolute judgement limits:**
   Humans usually do not perform well when attaching labels to levels, e.g. colors, shapes, sizes, etc. For example, using different tints of one colour for different time to CPAs are hard to be distinguished by humans. Instead, different hues can be used to assign different levels. Within aviation standard colors are assigned for different purposes, a display designed in this research should be conformal to this.
3. **Top-down Processing:**
   Humans are biased to interpret information as they expect it to be. This is particularly important is the case of an anomaly such as a warning signal. An example of this is the example given by figure 5.1, a sequence of on messages can invite the controller to perceive the last line to be 'on'. This problem can easily be solved by highlighting such an anomaly with for example colors. In the case of an explanation which highlights certain pixels in the SSD, it could be true that the algorithm always highlights a certain part of the SSD when an accurate prediction is presented. When another part of the SSD is highlighted this could possibly indicate an inaccurate prediction. In this example, the designer should consider extra cues to indicate the possibility of a false prediction.



**A** should be on
**B** should be on
**C** should be on
**D** should be off

**Figure 5.1:** An example of a checklist where the final line can easily be perceived as 'on'[86].

4. **Redundancy gain:**
   A certain message has a higher change of being perceived correctly if it is given in more than one ways. Presenting the same message using an alternative form also prevents perception in case of degradation of the system. For example, a warning sign presented visually and with audio, it can still be perceived if the visual display fails. Another clear example of a redundancy gain is the traffic light, it uses different locations and colors for the same message. Within CD&R a conflict could be presented with an audio signal in addition to the display.

5. **Discriminability:**
   As similar signals can cause confusion it is important to use discriminable elements. As described in the work of Tversky[75] two signals are received as similar by humans when the ratio is high. A designer of display should by all means prevent confusion by eliminating irrelevant similar features or highlight the differences between similar elements. In order to prevent confusion an explanation should be given in such a way that it does not interfere or can be confused with any other element which is shown on the interface.

### 5.2.2. Mental Model Principles

6. **Principle of pictorial realism:**
   The principle of pictorial realism was introduced in the work of Roscoe [60]. In order for a display to be conformal to the mental model of its controller it should look the variable that its present. For example, temperature is thought of as high or low values and should therefore be visualised on a vertical scale. Moreover, when a display contains multiple elements they should be placed in conformation with the actual environment if possible. As heading is a variable which is perceived by humans in a circular form, the SSD is also presented in a circular way.
7. **Principle of the moving part:**
   Also introduced by Roscoe[60] is the principle of the moving part. A moving elements on a display should move in a direction which is compatible with the mental model of its controller. This can be done using an ego-centric and an exo-centric depending on the controller. Figure 5.2 shows the difference between the two styles for a simple representation of the attitude display. Both displays show an aircraft banking to the right. However, the left and the right figure are conformal to the mental model of the pilot and someone outside the aircraft respectively. In contrary to cockpit displays, ATC uses preferably exo-centric displays as this is conformal to someone outside an aircraft. The interface designed in this research should therefore be exo-centric.

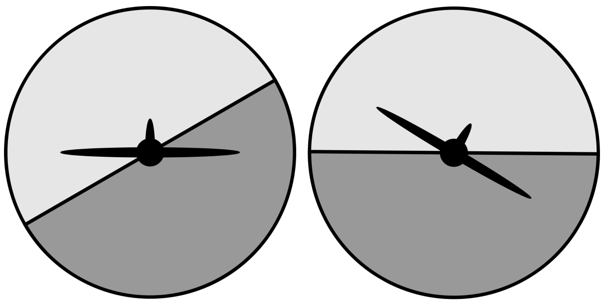


**Figure 5.2:** An ego-centric (left) and exo-centric (right) attitude displays.[86]

### 5.2.3. Principles Based on Attention

8. **Minimizing information access cost:**
   Different displays within an interface should be organised in such a way that frequent used information can easily be found in an easy to find location. When supervising multiple displays, a cost in time or effort usually exists when selective attention shifts between displays. Moreover, presenting displays in an organized way reduces these costs as displays are more easy to find. An example of this principle is the standardised basic T grouping used in all cockpits. This principle should be taken into account when designing an interface in this research. The radar screen which is already presented to an ATCo will be extended with an SSD and a yet to be designed explanation. These three elements should be organised such that the information access cost is minimal.
9. **Proximity compatibility principle:**
   The proximity between different information has an influence on the attention of the controller. When mental integration of multiple information sources is needed for a particular task, it is beneficial to have close proximity between information sources. Close proximity can be obtained by minimising the distance, using matching colors and uniform displays. An example of this principle

is a spider graph which is often used process control domains. On the other hand, close proximity can be harmful if focused attention is required for a certain task. When different information that do not contribute to the same task are presented in close proximity it can lead to confusion. During CD&R multiple tasks have to be conducted. The information sources needed and how they should relate to each other should be analyzed when designing the interface.

10. **Principle of multiple resources:**
    When a great amount of information has to be processed it could be beneficial to divide information over multiple resources. Auditory and visual resources could be addressed at the same time instead of entirely relying on visual information. During CD&R the greatest amount of information which has to be processed comes from the radar image. It contains different information of multiple aircraft within a sector. ATCo's have had extensive training in processing all the information from the radar screen visually. Dividing information over multiple resources is inapplicable and therefore unrelated to this context. However, as stated earlier, in addition to giving a warning visually it can be done using sound.

### 5.2.4. Memory principles

11. **Replace memory with visual information:**
    Humans have a limited memory and can not learn everything by heart. When designing a display the amount of prior knowledge needed has to be taken into account. Different types of controllers have different prior knowledge of the system. Computers programs like Microsoft Excel provide a clear example as they can be operated using prior knowledge or by knowledge shown on the computer screen. A novice user will address certain functions using a menu which gives 'knowledge in the world'. An expert user on the other hand has memorized certain keyboard shortcuts and relies more on 'knowledge in the head'. An ATCo is considered to be an expert user as a great amount of time will be spent using the system. However, designing a system which requires much prior knowledge increases time needed to get familiar with the system and therefore increases the change of being unaccepted. When designing the interface, the amount of prior knowledge needed should be minimised while presenting to much 'knowledge in the world' can overburden the controller.
12. **Principle of predictive aiding:**
    For effective use of a system controllers often require to be proactive. However, humans are generally not accurate when it comes to predicting how a state will behave in the future. Especially when other tasks also have to be executed at the same time a controller is tempted to take a reactive role. A display can help the controller become more proactive by showing the future prediction for a certain state. The SSD already does this by showing which speed vectors result in a LoS in the nearby future. Also, when two aircraft are in conflict it is shown on the radar screen usually with colors. These two features are already accurate ways to make sure the controller maintains a proactive role. As state prediction is inapplicable for the design of the explanation predictive aiding will not be considered when designing the explanation.
13. **Principle of consistency:**
    As humans use a certain display for a longer time they get familiar with it which can trigger inappropriate actions when a new display is introduced. When designing a display these habits should be taken into account as a way to avoid this does not exist. For example, within aviation standardised colors represent different functions throughout all displays. When designing a new interface it should be conformal to the displays already in use. In this research, an additional display will be added to the already existing radar screen. This should be designed in such a way that it is conformal to the existing displays which the ATCo already is using.

# 6 Preliminary Analysis

It was hypothesized that acceptance improves by increasing predictive performance and providing the solution conformal to the controller's mental model while providing insight into the reasoning behind a certain solution. The preliminary analysis will solely focus on optimizing the predictive performance of a CNN used for CD&R. The CNN architecture used by Van Rooijen et al. [59] will be the foundation from which the opportunities described in chapter 3 will be analysed in an experiment. The first goal of this experiment is to determine whether it is possible for a CNN to predict resolutions using a regression approach.

Chapter 3 also describes some suggestions in order to change the SSD input using feature engineering. These suggestions are: using clear colors, scaling the solution space and converting to a Cartesian representation of the SSD. In order to test these features multiple data sets have to be constructed. The second goal of this preliminary analyses is to evaluate the effects in terms of predictive performance of the network when it is trained using an alternative SSD format. These alternative formats include an SSD with a maximised solution space in terms of size and a Cartesian solution space which will use all pixels available.

The main goal of the research is to increase the acceptance by ATCo's of a CNN during CD&R. From literature it can be concluded that a shared mental model is a great tool to achieve this. During this preliminary analysis alterations to the original SSD will be made in order to test the effect on predictive performance. In order not to lose the shared mental model, the ATCo should be shown the same version of the SSD as fed to the CNN. A SSD which is very efficient in terms of predictive performance could have indistinguishable features for a human operator. As the controller and the CNN are using the same figure an trade off has to be made in order to have sufficient predictive performance while an SSD which is still easily readable for an ATCo. The following research question can be formulated for the preliminary analysis:

- Can a Convolutional Neural Network solve the Conflict Detection & Resolution problem?
- How can feature engineering be used to increase accuracy?
  - What is the effect of scaling the solution space in terms of predictive performance?
  - What is the effect of using an Cartesian SSD in terms of predictive performance?
  - Which SSD format is most suitable as the input to an CNN for CD&R while still maintaining a shared mental model with the ATCo.

The preliminary analysis will consist of 4 phases. In the first phase, the ATM simulator BlueSky [28] will accommodate simulations where resolutions are done by the Modified Voltage Potential (MVP) [29] algorithm. In the second phase multiple data sets with different SSD formats are generated from the simulation data. After this, the actual CNN will be designed which will be trained on the different data sets. In the last phase, the predictive performance of the different SSD formats will be evaluated. An overview of the methodology chronicle is given in figure 6.1. After the preliminary analysis, a discussion will be given followed by a conclusion on the preliminary analysis research questions.

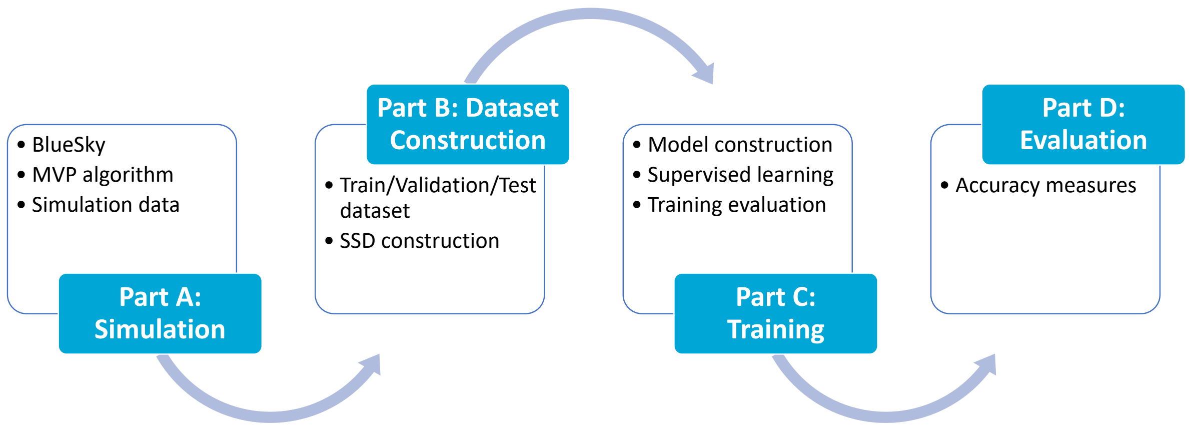


**Figure 6.1:** Methodology of the preliminary analysis.

## 6.1. Part A: Data Generation

In order to test whether a CNN, using regression, can successfully solve the CD&R problem a suitable dataset is needed. Ultimately, the CNN is supposed to approximate individual controller decisions. Van Rooijen et al.[59] did a human-in-the-loop experiment with 12 non-professional participants with varying skill in CD&R. They showed different results in terms of consistency which influenced the predictive power of the algorithm. From this, 12 individual data sets were constructed and used for training. The size of these data sets vary and contain an insufficient amount of data points for supervised learning. This research however, will only address the optimization of the predictive power of a CNN without assessing controller consistency. Therefore, the assumption will be made that professional ATCo's are sufficiently consistent such that a CNN can successfully learn resolution decisions. Furthermore, it is desired to have a data set of sufficient size in order to prevent convergence problems during training. During the preliminary analysis a computer generated data set will be used which will approximate a human controller as close as possible.

The open-source ATM simulator BlueSky developed by Hoekstra and Ellerbroek [27] will be used to generate data. An extensive amount of scenarios are constructed each consisting of two conflicting aircraft. Each scenario consists of a controlled aircraft with heading 0 and an intruding aircraft with varying headings to enforce different conflict angles. The conflict angle is defined clockwise starting at the heading of the controlled aircraft. Both aircraft initially fly at 250 knots and only the controlled aircraft will perform a resolution. The MVP algorithm developed by Hoekstra et al. [28] will be used to commit resolutions commands to the controlled aircraft. The look-ahead time, CPA and conflict angle will be varied to generate 8605 individual scenario's. Two different look-ahead times are used to enforce the MVP to have a long and short term strategy adding more variety to the proposed resolutions. Furthermore, an update interval of 10 seconds is used and a margin of 2 nm is set on top of the separation zone of 5 nm such that the MVP algorithm behaves more 'human like'. All parameters of the simulation can be found in table 6.1.

During each scenario the controlled aircraft performs a resolution after which it returns to its initial heading and speed of 0 degrees and 250 knots respectively. The MVP algorithm is designed for complete autonomous flight and does not considers turn dynamics. The resolutions are performed with incremental steps sending multiple resolutions to an aircraft. As these steps are small and it is undesirable to send multiple resolution commands to a pilot, the commands can not be used directly. In order to simulate a human controller as much as possible, the states of both aircraft at the beginning of a resolution manoeuvre and the maximum heading and speed deviation of the controlled aircraft are captured. An example of such scenario is given in figure 6.2. The states will later be used to construct the actual SSD which, together with the resolutions, form the dataset. From this 15% and 25% of the data set is randomly subsampled for validation and testing respectively.

**Table 6.1:** Parameters for data generation.

| Parameter | Value | Unit |
|---|---|---|
| Resolution algorithm | MVP | - |
| Number of conflicting aircraft | 2 | - |
| Resolution types | Combined heading and speed | - |
| Update interval | 10 | s |
| Protected zone radius | 5 | nm |
| Optimal cruise speed | 250 | kts |
| Min/max speed | 162/354 | kts |
| Conflict angles | [20, 21 .. 349, 350] | deg |
| Closest Point of Approach | [-3, -2.5..2.5, 3] | nm |
| Look-ahead time | 150, 300 | s |

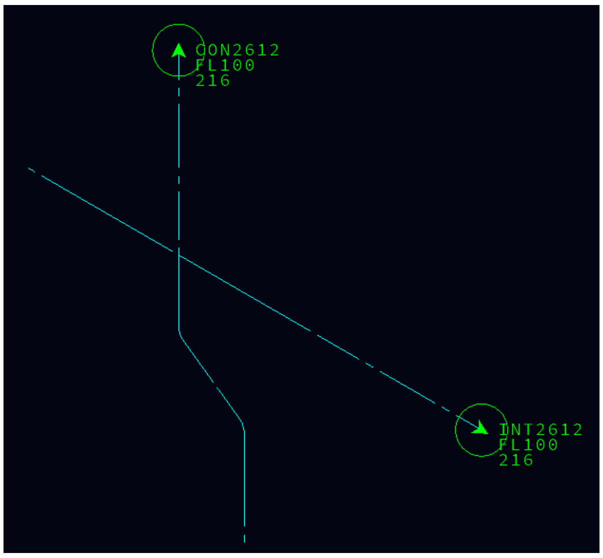


**Figure 6.2:** An example scenario with conflict angle of 120 degrees, look-ahead time of 150 seconds and cpa of 0 nm.

The MVP algorithm was able to successfully solve all conflict scenario with an exception of 18 cases which are deleted from the data set. These 18 cases have in common that the conflict angle was close to 0 or 360 degrees and the LoS occured at the beginning of each scenario. As LoS already occurred during initialisation, it is not an error of the MVP algorithm. This always occurred at the beginning of a simulation with conflict angles close to 0 or 360 degrees. As can be seen in figure 6.3, the CPA of all other scenarios lies between 5 and 7 nm with the majority close to 7 nm. Figure 6.4 shows the spread of all resolutions the MVP algorithm has issued to the controlled aircraft. It can be observed that the algorithm is biased to some specific velocity vectors. One could argue that this is conformal to a strategy which a human controller could use for multiple scenarios.

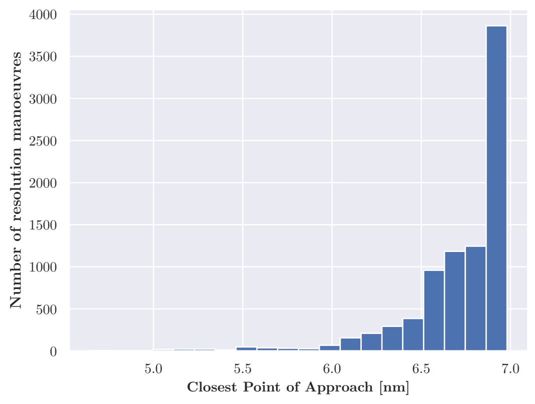


**Figure 6.3:** Closest Point of Approach for both scenarios.

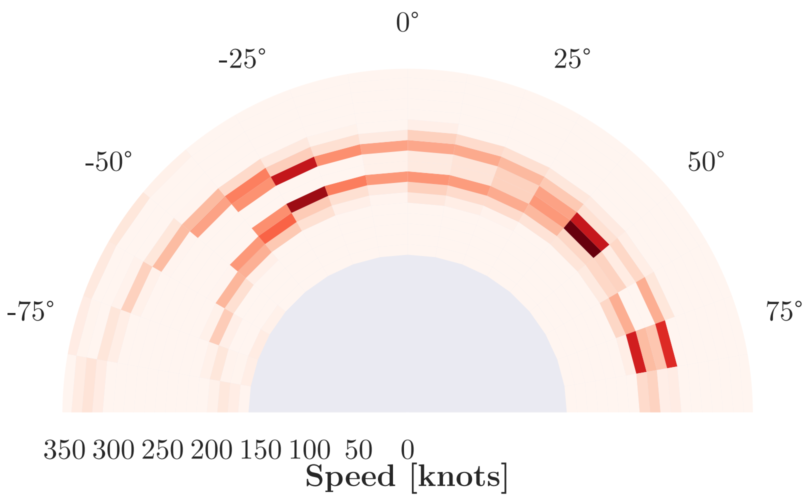


**Figure 6.4:** Spread of different resolutions the MVP algorithm issues. Resolutions are binned and color coded to indicate how often a resolution in a certain bin occurred. A darker color represents a more resolutions issued in the particular bin.

## 6.2. Part B: Solution Space Diagram Construction

Having a data set which only consists of aircraft states and a resolution gives the ability to manually construct the SSD's using multiple methods. This section will describe how three different SSD formats were constructed in order to test the effect of feature engineering on predictive accuracy. All SSD formats use a green line for the velocity vector and a single red color to indicate the FBZ of the intruding aircraft. Furthermore, the SSD formats are defined to investigate fundamental design methods considered in the literature review. The effect of the size of the figure fed to the CNN will not be considered during the preliminary analysis, i.e. All SSD's will have 64 by 32 pixels. The following three versions which are shown visually in figure 6.5 were constructed:

- **Standard SSD:**
  The first SSD format will be used as a baseline in order to compare the other two. The solution space is defined by a half donut where the maximum speed is set equal to the maximum radius

possible in the figure. The minimum speed boundary is defined proportional to this maximum radius. After this, the 'start' of the FBZ triangle is calculated using the velocity vector of the intruding aircraft. The distance and heading towards the intruding aircraft is used to calculate the outer angles of the FBZ. All solution space with a heading between these angles is then colored red.

- **Zoomed SSD:**
  A considerable amount of the pixels of the standard SSD belong to the 'background' and are therefore not used in the calculation of a resolution. In order to resolve this problem a SSD format where the solution space is maximised is proposed. As can be seen in pseudocode 1, the solution space boundaries are manually defined. For each pixel within the solution space the distance to the center of the SSD is calculated using Pythagoras. As the distance is used to indicate the speed of a certain velocity vector, it can be used to calculate the real velocity using

  $$v = v_{min} + (v_{max} - v_{min})\frac{d_{x,y} - d_{min}}{d_{max} - d_{min}} \tag{6.1}$$

  with $v_{min}$ the minimum airspeed, $v_{max}$ the maximum airspeed, $d_{x,y}$ distance between pixel x,y and the center of the SSD, $d_{min}$ lower solution space boundary and $d_{max}$ upper solution space boundary. For the heading of the velocity vector such transformation is not needed as it is equal to the heading of the pixel relative to the center of the SSD. After this, for each velocity vector the flight trajectory will be calculated. If a flight trajectory results in LoS in less tham 10 minutes the related pixel will be colored red. As a consequence of this method, a single FBZ does not necessarily appear as a triangle on the SSD as can be seen in figure 6.5.

**Algorithm 1** Construction of the Zoomed SSD.

```
 1: Define empty array with size equal to SSD image
 2: Define image radii for solution space boundaries
 3: for each column in SSD array do
 4:     for each row in SSD array do
 5:         if pixel belongs to solution space then
 6:             Append pixel coordinate to solution space coordinate list
 7: Calculate pixel distance between solution space coordinates and SSD center using Pythagoras
 8: Calculate real velocity for all solution space coordinates using equation 6.1
 9: Calculate real heading for all solution space coordinates using the tangent function
10: Calculate flight trajectory intruding aircraft
11: for each velocity vector inside solution space do
12:     Calculate related flight trajectory using Trigonometric functions
13:     Calculate minimum distance between controlled and intruding flight trajectories using Pythagoras
14:     if minimum distance between flight trajectories < separation zone then
15:         Append pixel coordinate to FBZ
16: Find pixel coordinates belonging to controlled aircraft velocity vector
17: Assign pixel values to solution space, FBZ and controlled velocity vector
```

- **Cartesian SSD:** The last SSD format is optimized to include all available pixels into the solution space. In order to do this the solution space is converted from the half donut shape to a rectangle equal to the figure size. The construction steps are presented by the pseudo code of algorithm 2. It is done in a similar way as the Zoomed SSD but velocity vectors are assigned to pixels differently. The main difference is that the heading and velocity are indicated by horizontal and vertical pixel position, respectively. This means that the upper and lower velocity boundaries are represented by the upper and lower pixel rows. The first and last pixel column represent a heading of -90 and 90 degrees, respectively. All pixels between the boundaries are given heading and velocity values linearly in relation to the boundaries. As can be seen in figure 6.5, this methods fundamentally changes the appearance of the SSD. As it is optimised to include all pixels into the solution space the half donut shape of the SSD is lost completely. It can also be seen that shape of the FBZ has changed considerably more compared with the Zoomed SSD.

**Algorithm 2** Construction of the Cartesian SSD.

```
1: Define empty array with size equal to SSD image
2: Obtain heading related to each pixel column
3: Obtain velocity related to each pixel row
4: Calculate flight trajectory intruding aircraft
5: for each combination of heading and velocity do
6:     Calculate related flight trajectory using Trigonometric functions
7:     Calculate minimum distance between controlled and intruding flight trajectories using Pythagoras
8:     if minimum distance between flight trajectories < separation zone then
9:         Append pixel coordinate to FBZ
10: Find pixel coordinates belonging to controlled aircraft velocity vector
11: Assign pixel values to solution space, FBZ and controlled velocity vector
```

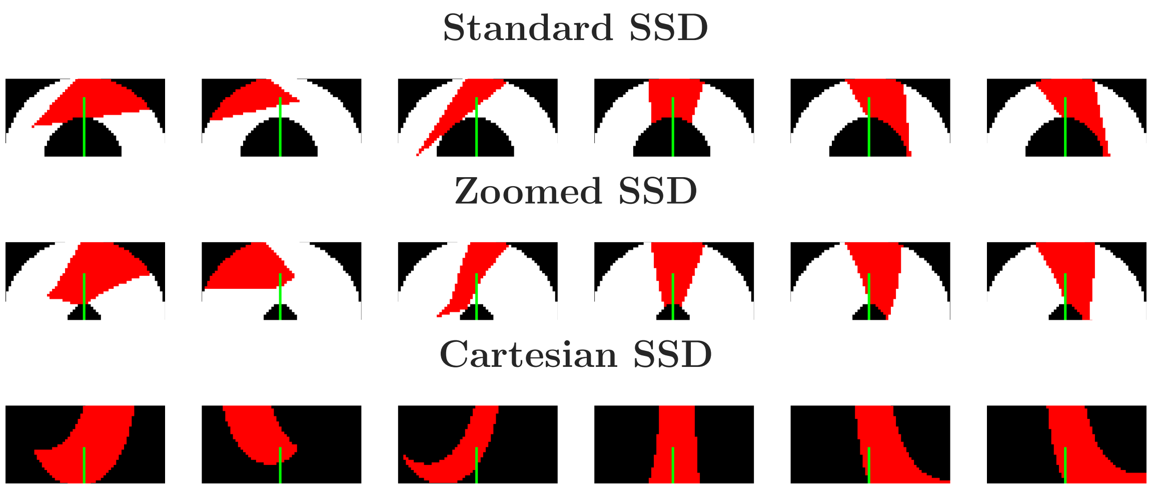


**Figure 6.5:** The three different SSD formats from multiple scenario's.

## 6.3. Part C: Model and Training

The network used for this experiment will be similar to the network used by van Rooijen et al.[59]. However, the last SoftMax layer of the original architecture can not be used for regression and has been replaced by a ReLU activation. Another change in architecture will be the output size which will be 2 in all cases. An overview of the architecture is given in table 6.2.

**Table 6.2:** Network architecture during the preliminary analysis.

| Layer type | Input size | Filter size | Stride | Filters | Activation function | Output size |
|---|---|---|---|---|---|---|
| Convolutional | 32x64x3 | 2x2 | 1 | 32 | ReLU | 31x63x32 |
| Max-Pooling | 31x63x32 | | | | | 15x31x32 |
| Convolutional | 15x31x32 | 2x2 | 1 | 64 | ReLU | 14x30x64 |
| Max-Pooling | 14x30x64 | | | | | 7x15x64 |
| Convolutional | 7x15x64 | 2x2 | 1 | 32 | ReLU | 6x14x32 |
| Flatten | 6x14x32 | | | | | 2688 |
| Fully Connected | 2688 | | | | | 1024 |
| Dropout | 1024 | | | | | 1024 |
| Fully Connected | 1024 | | | | ReLU | 2 |

As the fundamental task of the CNN changes different parameters have to be selected. Even though hyperparameter tuning will not be analysed during this experiment, it is inevitable in order to obtain satisfactory model performance. The hyperparameters used in the experiment of van Rooijen et al.[59] has been the starting point for selecting hyperparameters. However, by using a different data set and doing regression the following hyperparameters became inapplicable:

- Adam optimization algorithm
- Categorical entropy loss function
- K-folds validation
- Mini batch size
- Steps per epoch

A different optimization algorithm and loss function had to be selected for which Stochastic Gradient Descent (SGD) and MSE have been selected respectively. Firstly introduced by Robbins and Monro[58], SGD has been proven to be a straight forward optimization algorithm with suitable performance. Using the MSE is also a straight forward decision as it is also used as accuracy measure during the evaluation phase. Furthermore, it has been observed that 30 epochs is not enough for reaching satisfactory convergence using regression. Therefore, training will be done using 100 epoch which have proven to enable better training performance. All hyperparameters of the network used in this preliminary analysis are shown in table 6.3.

**Table 6.3:** Characteristics used to train the Convolutional Neural Network during the preliminary analysis.

| Hyperparameter | Value |
|---|---|
| Optimization algorithm | Stochastic Gradient Descent |
| Loss function | Mean Squared Error |
| Train/validation/test ratio | 60/15/25 |
| Epochs | 100 |
| Learning rate | 0.01 |
| Dropout rate | 20 |
| Input image dimensions | 32x64 |

Training the network using the above mentioned hyperparameters results in train and validation loss during training as shown in figures 6.6 and 6.7 respectively. First of all, it can be observed convergence is achieved for training with all SSD formats. Hence, it can be concluded that a CNN can solve the CD&R problem using regression successfully. Another observation which can be made, is the slightly improved training performance of the zoomed and Cartesian SSD compared to the standard SSD. Also, the zoomed SSD surprisingly shows a negligible better performance compared to the Cartesian SSD.

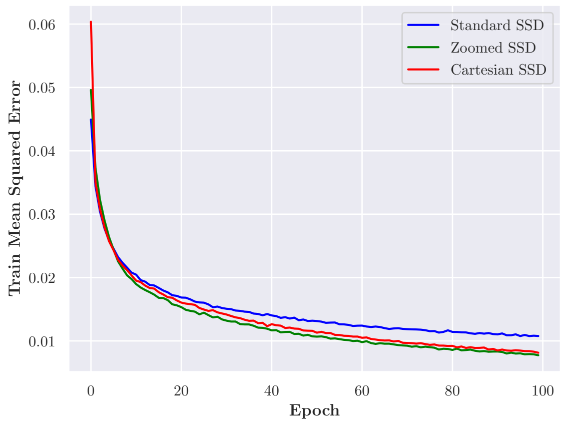


Figure 6.6: Mean Squared Error of the training data set.

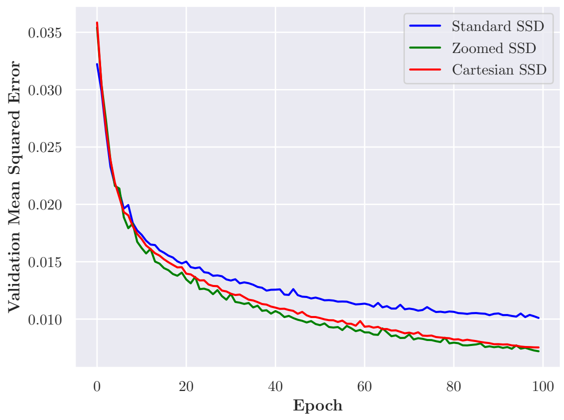


Figure 6.7: Mean Squared Error of the validation data set.

## 6.4. Part D: Test Results

In order to evaluate the SSD formatss the test dataset, consisting of 2151 data instances, is fed to the trained models. Each normalized output consists of a value for heading and speed respectively which has first been converted to real values. As can be seen in table 6.4, the average absolute heading and speed errors of the zoomed and Cartesian SSD has slightly improved compared to the standard SSD. It can be observed that the zoomed and Cartesian SSD are more accurate in heading and speed respectively. Figures 6.8 and 6.9 shows the distribution of the test errors using all SSD formats in a violin plot. In these figures the same results can be observed.

**Table 6.4:** Average absolute heading and speed error for the different SSD formats.

| | Standard SSD | Zoomed SSD | Cartesian SSD |
|---|---|---|---|
| Average absolute heading error [degrees] | 7.64 | 6.81 | 6.89 |
| Average absolute speed error [knots] | 15.11 | 13.14 | 12.83 |

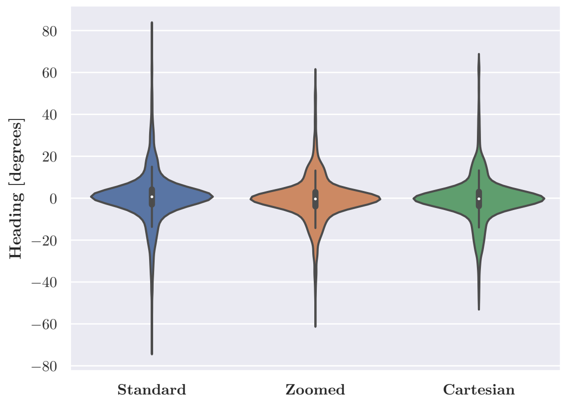


Figure 6.8: Heading error evaluation of test data set.

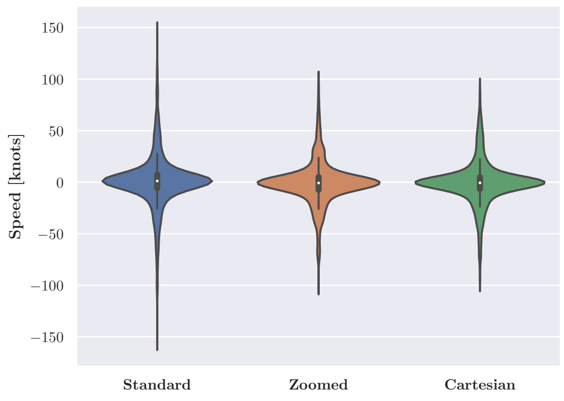


Figure 6.9: Speed error evaluation of test data set.

In order to visually show these errors, the true and predicted velocity vectors are projected on the input SSD in figure 6.10. It can be observed that the blue predicted vector in all cases is directed in the direction of the purple true vector. However not in all cases this is done accurately, in the second scenario from the left all three SSD formats propose a resolution which will result in LoS for example.

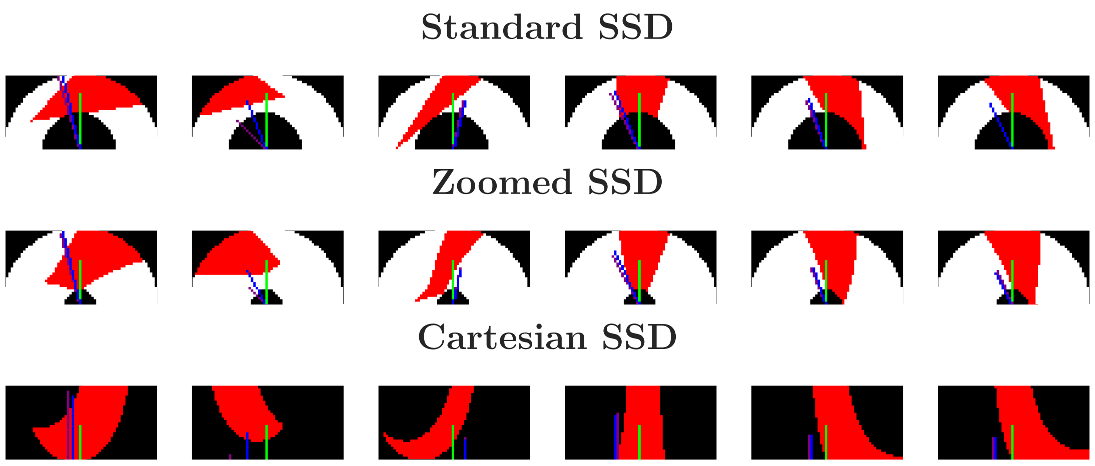


**Figure 6.10:** Visualization of 6 random subsampled predictions projected on the input image. The purple velocity vector represents the true resolution while the blue velocity vector represents the resolution as predicted by the model.

## 6.5. Conclusions and Discussion

Using a regression approach instead of classification not only changes the possible accuracy measure but also the amount of data needed. As the model now predicts exact values instead of probabilities of two or three bins, the task becomes more complex. Therefore, a significantly larger data set was needed in order to train the network successfully compared to the data used in the work of van Rooijen et al. [59]. This data set was generated using decisions of amateur controllers with varying consistencies. In this phase of the research however, consistency is not considered in any way as the main focus will be with improving accuracy. In order to obtain a larger data set, artificial data was generated using a 'free flight' ATC solution algorithm which can be considered to be highly consistent.

Due to the different data set and accuracy measure a direct numerical comparison is impossible. Despite this, many subjective conclusions can be drawn comparing the results. First of all, the bins used by van Rooijen et al. [59] were designed in such way that only a certain resolution strategy was advised. The actual resolution within this strategy is still to be determined by the controller without any assistance of the CNN. With the regression approach proposed in this research, an exact solution will be advised which can be interpreted directly. Hence, one could conclude that a regression approach will result in a lower workload compared to classification with large bins. Furthermore, the average heading errors of all SSD formats are smaller than the smallest heading bin used by van Rooijen et al. [59]. The bin for speed was only defined as faster or slower relative to the current speed so a comparison can not be made. However, given the large flight envelope of the simulation done the error can be considered small. All things considered, it can be concluded that a CNN, given a SSD as input, can predict resolutions more accurately when it uses regression instead of classification. It will provide a more useful solution to an ATCo and can achieve higher predictive performance.

During the preliminary analysis three SSD formats were constructed and used for training. All of them contained the same information and used the same color coding. The standard SSD format will be used as a baseline and is constructed using the same method as done in all previous literature. However, due to the large simulation flight envelope, the solution space is reasonably larger compared to most SSDs used in literature. Nevertheless, the CNN can quite accurately predict resolutions using this SSD format. The zoomed SSD is constructed in an alternative way which allows for more scaling options. It shows a slightly improved performance compared to the standard SSD format. It can also be observed that the shape of the FBZ is not a triangle with straight lines anymore. The Cartesian SSD format uses all pixels available as solution space but only shows similar overall performance to the zoomed SSD. The fact that the Cartesian format is not conformal to the controllers mental model is a critical drawback. A solution will be to show the controller and network a different version of the SSD. However, this is also undesirable as the shared mental model between human and machine will then be lost. Therefore, it can be concluded that the zoomed SSD format is the most suitable as it provides the highest accuracy while still maintaining a shared mental model between controller and automation system.

# 7 Conclusion

To conclude this preliminary thesis an effort is done below to answer the questions introduced in the introduction. After this, a short discussion will follow on the direction of further research.

1. **What aspects influence acceptance by Air Traffic controllers?**
   An automated system will only be accepted if a controller understands its functioning. It should behave as expected by the controller which is directly related to predictive performance. Furthermore, interpretability can be increased by presenting the output of the system in such way that it gives insight why a certain solution is given.
2. **How can a Convolutional Neural Network solve the Conflict detection & Resolution problem?**
   The preliminary analysis has shown that a CNN can more accurately predict resolutions using regression instead of classification. Furthermore, regression can further decrease controller workload as a precise resolution can be advised instead of a strategy.
3. **How can feature engineering be used to increase accuracy?**
   The preliminary analysis has shown that increasing the size of the solution space positively influences the predictive power. A Cartesian SSD on the other hand, does not achieve greater accuracy compared to a SSD with a maximised solution space. The zoomed SSD is most optimal for resolution prediction using a CNN as accuracy is maximised and the SSD is still conformal to the controllers mental model.
4. **How should the output of the system be presented to an Air traffic Controller?**
   In order to improve interpretability the interface of the system should be designed accordingly. It should contain an explanation which provides reasoning to why a solution is advised. The advised resolution and the explanation should be included into the existing interface using an ecological design approach.

Exactly predicting a certain resolutions using regression fundamentally changes the task of the CNN. However, the CNN architecture used in the preliminary analysis was optimized for a much simpler classification task. Further research should be conducted in order to analyse what CNN architecture is most optimal for regression in the context of CD&R. On top of this, more information can be added to the SSD if data is generated using a more realistic multi aircraft simulation. For example, a color scheme which indicates what the $t_{CPA}$ is could possibly further increase predictive power. For the remainder of this research the areas of XAI and interface design will not be analysed as multiple opportunities to further optimise predictive power still remain. In the following chapter further research will be proposed to analyse the CNN architecture and the features of an SSD consisting more information with an experiment.

# 8 Experiment Proposal

This research aims to analyse how a personal-sensitive trained trained CNN with the SSD as input can be used most optimal during CD&R. The ultimate goal is a human-machine automation system which will be accepted by ATCo's and decreases workload. This goal is linked to the following research question:

***"How can the acceptance of Convolutional Neural Networks for Conflict Detection & Resolution be improved among Air Traffic Controllers?"***

From the preliminary analysis it became clear that a CNN can predict exact velocity vectors with reasonable accuracy. Using the architecture from van Rooijen et al. [59] for a regression tasks showed promising but far from perfect performance. Furthermore, it was proven that increasing the size of the SSD's solutions space positively influences the predictive power of the CNN. Both conclusions show promising results but are not yet analysed fully. The architecture used during the preliminary analysis was optimized by van Rooijen et al. [59] to do a classification task and more information can be included into the SSD when data is coming from a more extensive simulation. Therefore, the final experiment will be designed solely with the objective of further increasing predictive power, since it is hypothesized that increasing predictive power will lead to higher acceptance. During the experiment it will be assumed that artificially generated resolution data can be constructed such that the consistency is similar to professional ATCo's. With this assumption the notion of consistency will not be analysed during this experiment.

The experiment will consist of three phases as shown in figure 8.1. In the first phase, two more realistic simulations are done from which the data will be used throughout this experiment. Both simulations are designed to generate data as if it is coming from a human ATCo's with different strategies. Data will be generated during these simulations to a create baseline SSD's. After this, an effort will be done to find a more optimal network architecture inspired by two classic networks being Lenet-5 [41] and VGG[67]. In the final phase, more information will be added to the SSD in order to analyse the effect on predictive performance.

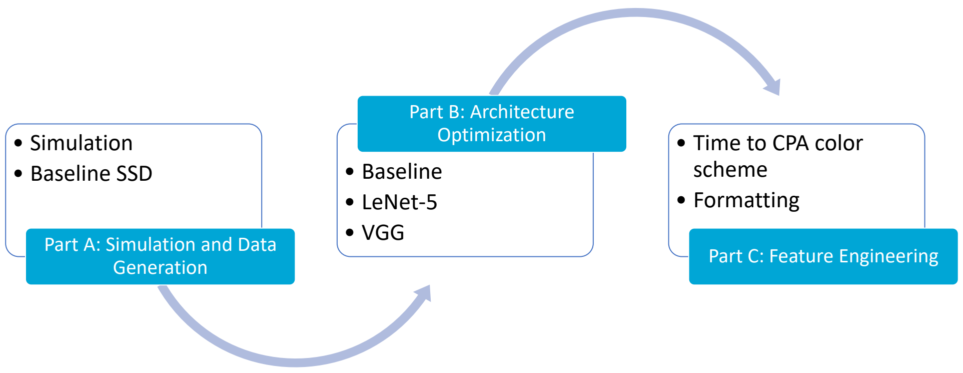

• Simulation
• Baseline SSD
Part A: Simulation and Data Generation
Part B: Architecture Optimization
• Baseline
• LeNet-5
• VGG
• Time to CPA color scheme
• Formatting
Part C: Feature Engineering

**Figure 8.1:** Experiment setup.

## 8.1. Part A: Simulation and Data Generation

In order to test the predictive performance of the to be designed CNN model a data set is needed. The data set should resemble a real life scenario as close as possible and more importantly deliver a data set suitable for training. First of all, the simulation scenario and its constraints will be described. After this, an explanation will be given on the simulation software and the resolution algorithm used during the simulation. Finally, the data extracted from the simulation will be used to construct a baseline SSD.

### 8.1.1. Scenario

The scenario will be constructed to simulate realistic air traffic flows following airways. However, as conflicts are only solved horizontally a real Control Area (CTA) with existing routes can only be used with heavy modification. Instead a theoretic CTA will be defined inspired by Amsterdam CTA West shown in figure 8.2. The overall shape of this CTA will be used with theoretical waypoints and routes. The traffic flow will have a variable traffic density simulating a normal traffic flow during the simulation. For simplicity, the simulation is exclusively done with Boeing 737s flying at a realistic altitude such that there flight envelope is limited. In order to obtain enough data for training , the scenario will at least contain 7000 resolutions.

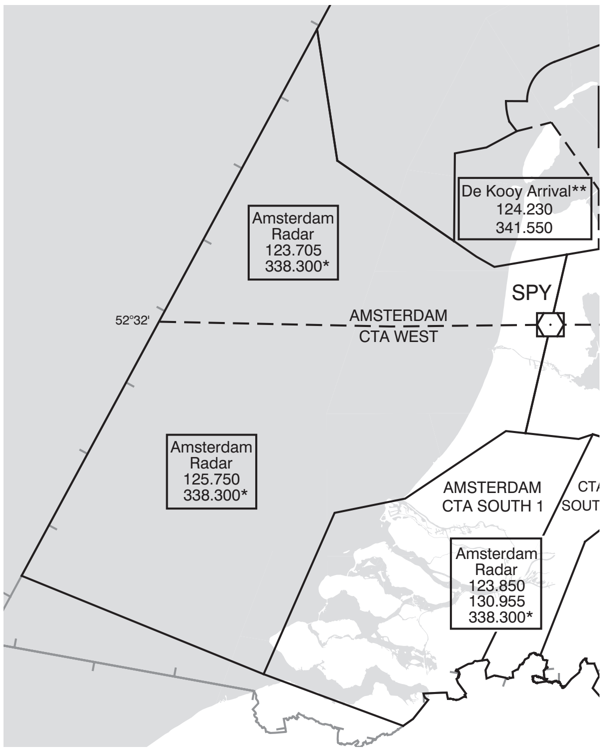


**Figure 8.2:** Chart of Amsterdam CTA West.

A resolution can typically be divided into a manoeuvre to solve the conflict and a direct to waypoint command after the conflict is resolved. The SSD itself presents all safe and unsafe solution space in a clear manner. For an ATCo it is obvious to interpret whether the target waypoint lies on a safe heading. Therefore, only the manoeuvre to resolve the conflict itself will be included into the data set as resolution.

### 8.1.2. Software

Similarly to the preliminary analysis, the above described scenario will be accommodated by the open-source ATM simulator BlueSky developed by Hoekstra and Ellerbroek [27]. In BlueSky it is possible to create a custom ATC area with waypoints in a real world enviroment. Also, aircraft have a real life inspired flight envelope and scenario's are easily applied using python generated text files. Two simulation will be conducted to imitate two different controller with a long and short-term strategy. The MVP algorithm will enforce this by a look-a-head time of 5 and 2.5 minutes respectively. Other MVP settings to imitate a human controller, are a no-look time of 10 seconds and an additional separation zone margin of 5 nm. At every second the identification code, latitude, longitude, heading and speed will be logged for all aircraft. In order to construct the SSDs, the states will be saved at the start of each resolution manoeuvre. The resolution itself will be saved at the end of the manoeuvre to imitate a resolution as given by a controller.

### 8.1.3. SSD Construction

Two data sets will be created consisting of a resolution in the form of a velocity vector and a baseline SSD. The SSD will have equal solution space size as the zoomed SSD from the preliminary analysis but contains the same information as the SSD used in the work of van Rooijen et al. [59]. The FBZ's are indicated using red, orange and grey, representing a $t_{CPA}$ lower than 60 seconds, between 60 and 120 seconds and larger than 120 seconds respectively. Furthermore, the heading towards the exit waypoint will be indicated by a blue line projected on the solution space. This SSD format will be used throughout part B where the model architecture will be analysed.

## 8.2. Part B: Architecture optimization

Finding a suitable network architecture can generally be a tedious task involving a trial and error approach without a clear procedure. However, starting with small filters and layers and increasing this incrementally is a basic principle when constructing a CNN. Adhering to this principle will enforce the network to first capture fundamental local information after which the information captured will become higher level and more global. In order to find a more optimal CNN architecture for predicting resolutions three models will be constructed. First of all, the model from the preliminary analyses will be used as a baseline. After this, two models will be constructed using the above principle applied to the general structure of two classical CNNs. The first CNN structure used was introduced by Lecun et al. [41] in which convolutional layers and pooling layers alternate. The original architecture, Lenet-5, is visually presented in figure 3.4. The other network used as inspiration is the VGG type architecture introduced by Simonyan and Zisserman [67]. In this type of network more convolutional layers are placed in between the pooling layers as can be seen in figure 8.3. For this experiment, the general structure of alternation between two convolutional layers and one pooling layer will be used to construct the model.

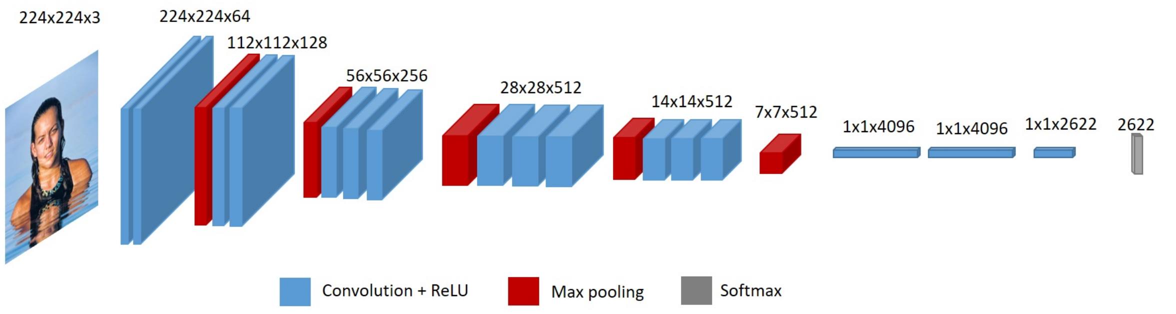


**Figure 8.3:** Architecture overview of VGG-16[67].

The LeNet and VGG inspired models will be optimised using a similar amount of effort in multiple steps. For both models, the starting point will be a network with a minimal number of shallow layers for which optimal performance will be found using hyperparameter tuning. After this, the networks architecture will be increased in a step-wise manner while maintaining compliance with the above stated principle. Each variation to the starting networks will again be optimized using basic hyperparameter tuning with a similar amount of effort. During this phase the hyperparameters shown in table 8.1 will remain constant while the number of epochs, learning rate and dropout rate will be varied. After the most optimal architecture is found in terms of number of layers, layer size and filter size, an effort will be done to further increase predictive performance by experimenting with different activation functions. Finally, a more extensive effort to hyperparameter tuning will be done using more advanced optimization algorithms. As computational power will always be limited, a trade-off might be applicable between predictive power and architecture size. This should be taken into account during the entire process.

**Table 8.1:** Constant hyperparameters during architecture optimization

| **Hyperparameter** | **Value** |
|---|---|
| Optimization algorithm | Stochastic Gradient Descent |
| Loss function | Mean Squared Error |
| Train/validation/test ratio | 60/15/25 |
| Input image dimensions | 32x64 |

## 8.3. Part C: Feature Engineering

Using a more realistic scenario more information can be included into the SSD which gives the opportunity to further analyse how predictive performance can be increased using feature engineering. During the simulation done in the preliminary analysis a conflict was resolved after which the aircraft returned to its original heading and speed. During this experiment however, all aircraft have a target waypoint which might influences the most optimal resolution given a certain conflict. Different formats to include the heading towards this waypoint can be applied concerning shape, color and thickness.

Doing a simulation with more than two aircraft at the same time results in multiple FBZ's in the SSD with different $t_{CPA}$. In the original SSD this is indicated using bins with grey, orange and yellow. Many variations to this colour scheme can have an effect on the accuracy of the model. For example, a color scheme with continuous pixel values using different saturation levels of a single color. Another option is applying different colors based on distance towards intruding aircraft.

To truly analyse each alteration to the SSD, all variations will be used for training with the most optimal model established in part B. The number of variation that will be tested is dependent on available project time and the accuracy achieved during the previous phases. Alterations to the SSD which are most likely to have a larger impact will be done first prior to changes which are believed to have a lower impact. Therefore, this phase will start with analysing how accuracy is influenced using other FBZ coloring for different $t_{CPA}$ indication. It is hypothesized that color coding the FBZ using a continuous color scheme will achieve a higher predictive performance opposed to a binned color scheme. Optionally, an analysis can be done on the general formatting of the SSD, included thickness and shapes, and adding extra information to the SSD. Examples of extra information could be heading to non-target waypoints and non-conflicting aircraft. Throughout this entire phase of the research a the trade-off between controller conformance and accuracy should be taken into account. Alternating the SSD with as goal a higher predictive power could result in undesirable indistinguishable features for a human operator.

# Part III

# Appendices

# A Simulation

This chapter details the simulations conducted and the data sets obtained by it. The first section gives an overview of the distribution of conflict angles during the simulation. After this, the previously shown resolution distribution will be presented in more detail. Finally, a graph will present the distribution of SSD complexity for both strategies.

## A.1. Conflict Angle

Figure A.1 presents the conflicts angles between $A_{con}$ and $A_{int}$ for both simulations. As simulations are done in the same ATC area, bith simulations show similar conflict angles. However, it can be seen that the Reactive strategy has more smaller conflict angles due to sequential conflicts, i.e a conflict caused by the resolution of a previous conflict.

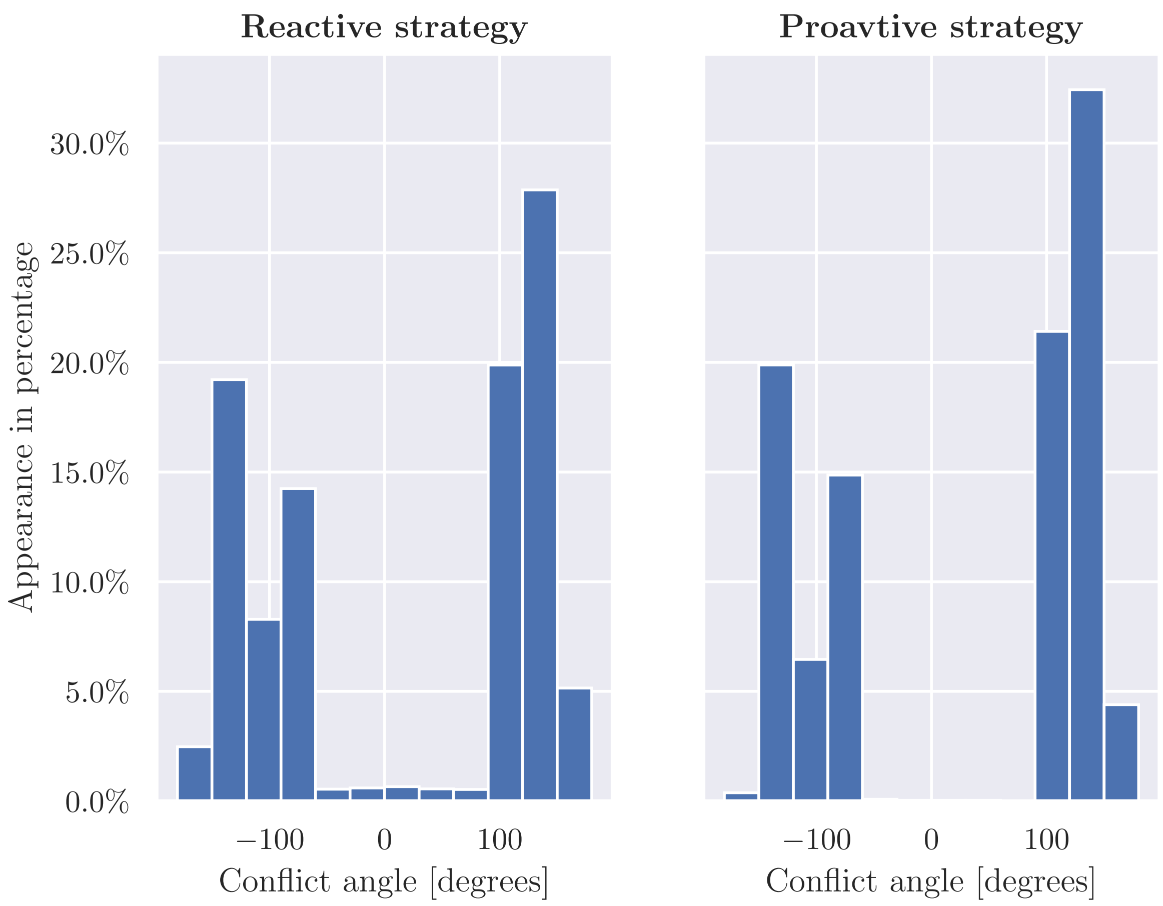


**Figure A.1:** Conflict angle distribution for the Reactive and Proactive strategy simulation.

## A.2. MVP strategy

Figure A.2 and A.3 shows the heat map of resolution issued by the MVP algorithm [28] for the simulations with the Proactive and Reactive strategy, respectively. As the initial speed of $A_{con}$ is always 320 knots it can be observed that for this simulation the algorithm only issues resolutions with equal or lower velocity. Additionally, it can be observed that the Reactive strategy uses larger heading and speed changes more often compared to the Proavtive strategy. This is as expected due to the shorter look-ahead-time and larger separation factor.

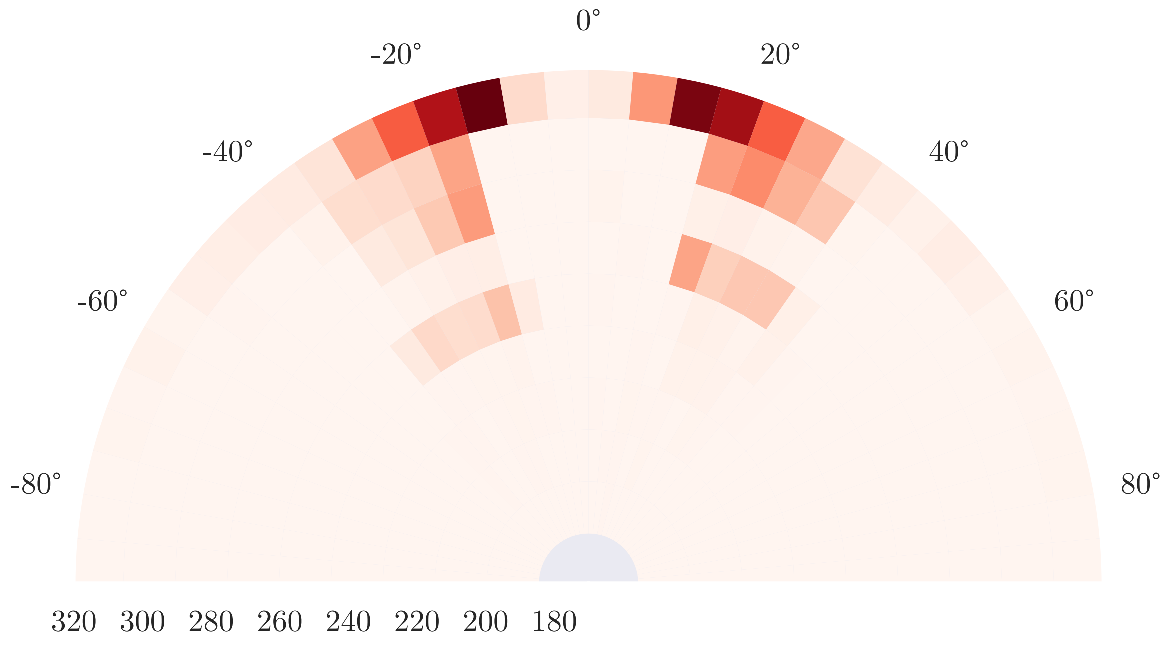


**Figure A.2:** Heat map of the resolution distribution for the Reactive strategy.

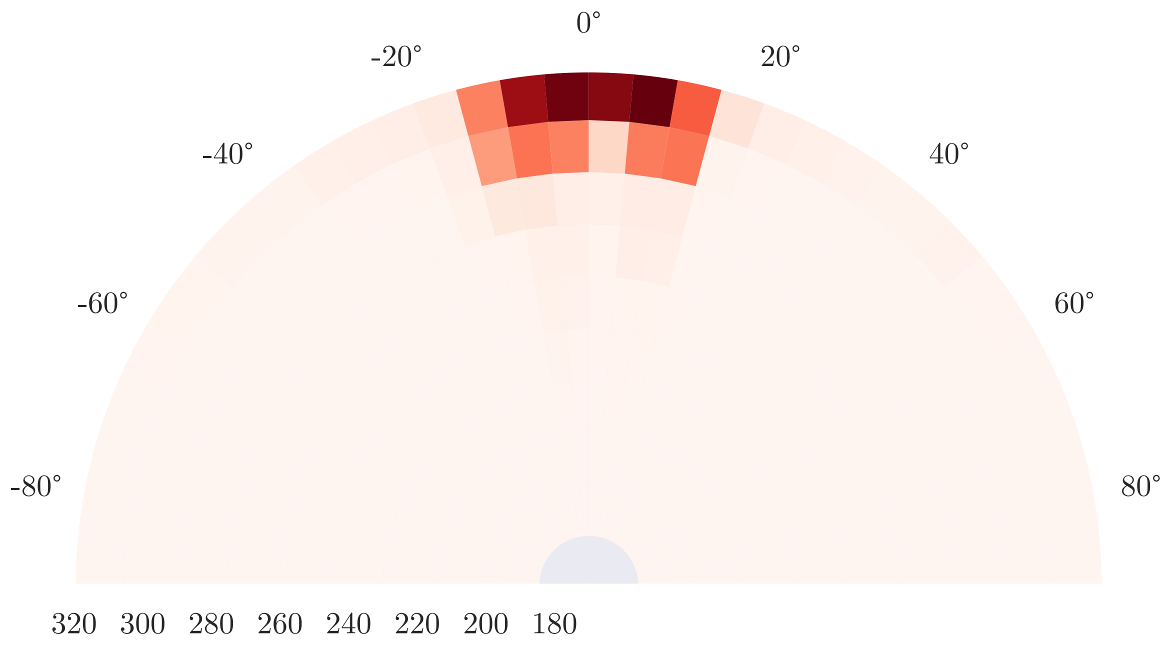


**Figure A.3:** Heat map of the resolution distribution for the Proactive strategy.

## A.3. SSD Complexity

Figure A.4 shows the spread of SSD complexity over both data sets. The percentage of solution space pixels occupied by a FBZ is defined as the SSD complexity, e.g. if al solution space is occupied by FBZ the complexity is 100%. Only trajectories with LoS smaller than 600 seconds are included in the FBZ and no distinction in lower LoS is made with the calculation of these graphs.

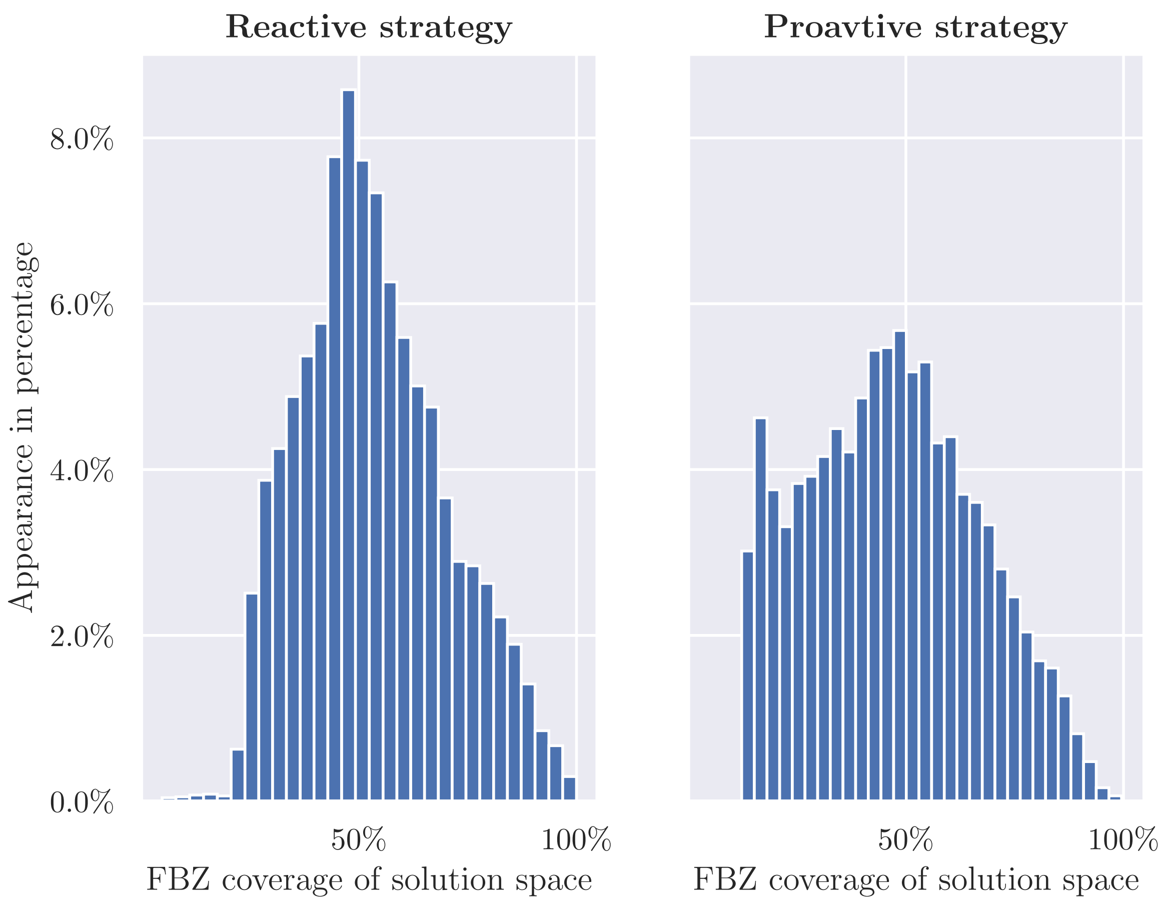


**Figure A.4:** Distribution off SSD complexity in the data sets coming from both simulations.

# B
# Architecture Optimization

This chapter will give a more detailed insight intro the process of finding a suitable CNN architecture. The first 6 sections will present the validations errors during training of all individual steps for both strategies. After this, the final model will be compared with the initial model in terms of training validation error and predictive performance.

## B.1. Padding and Output Layer

Figure B.1 presents the training validation errors for both strategies for the first two optimization steps. First, it was analysed whether padding has a negative influence on the model. Using a network with padding enables more architecture variations to be analysed in further steps. Compared to the baseline strategy, it was found that there is a limited performance reduction for the Reactive strategy while the Proactive strategy shows similar performance. Due to the greater flexibility in further steps it was chosen to continue with padding. After this, the possibility of a linear output layer activation function was evaluated. This is a common output layer function for regressive networks and therefore expected to increase performance. It can be observed in figure B.1 that this is indeed true, especially for the Proactive Strategy.

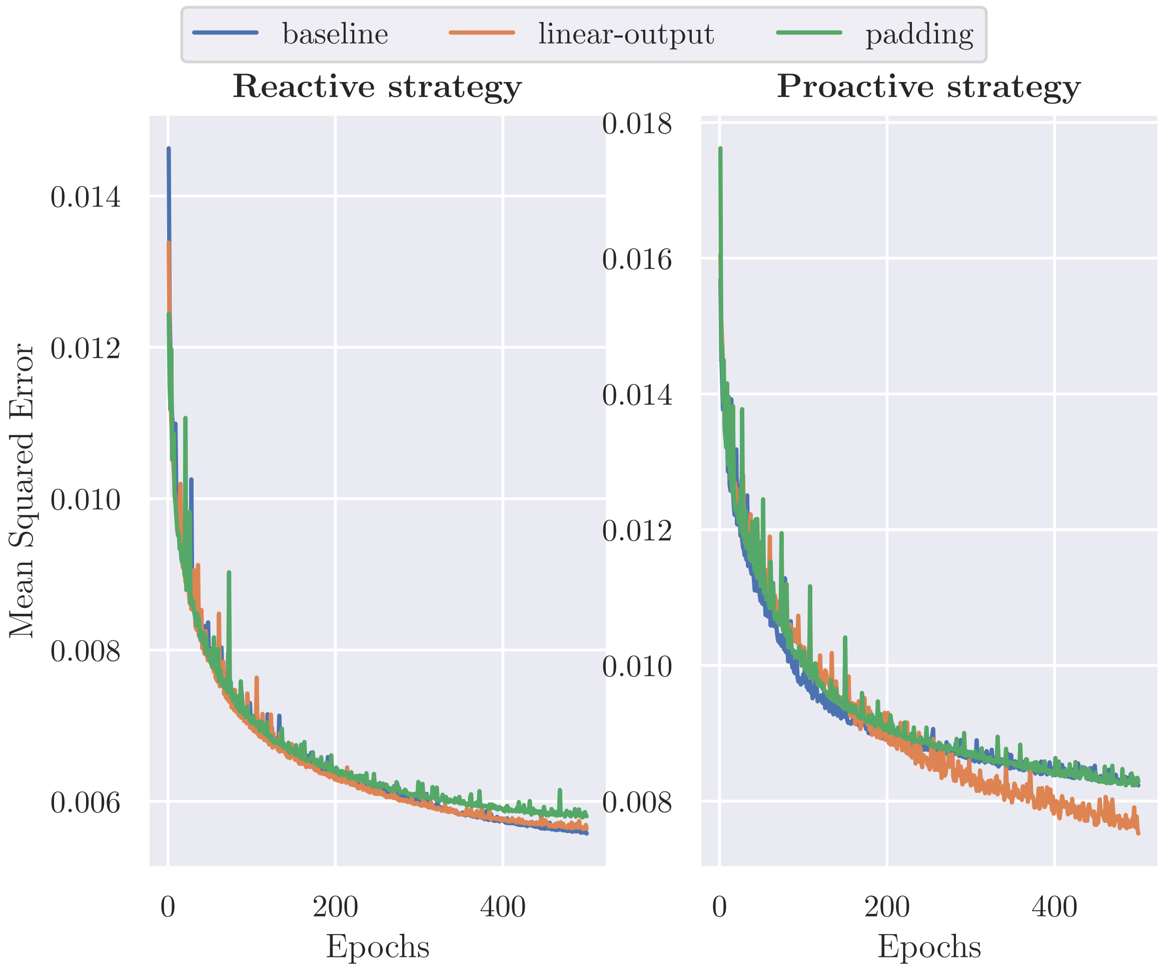


**Figure B.1:** Validation error during training for the baseline architecture, an architecture with padding and one with a linear output activation function.

## B.2. Filter Size

The original architecture uses only 2x2 sized filters in the convolutional layers while odd sized filters which increase in size deeper into the network are more common. Most common is a structure of 3x3, 5x5, 7x7, etc. which enables to capture local features early in the network after which more high level features are observed. At this step, it is evaluated whether this structure also results in better predictive performance given the model analysed in this research. Figure B.2 show that this is indeed true based on training validation error for both strategies using a 3 layer architecture. At this step, only the general filter size structure was evaluated. At later steps this structure will be used when more or less layers are evaluated.

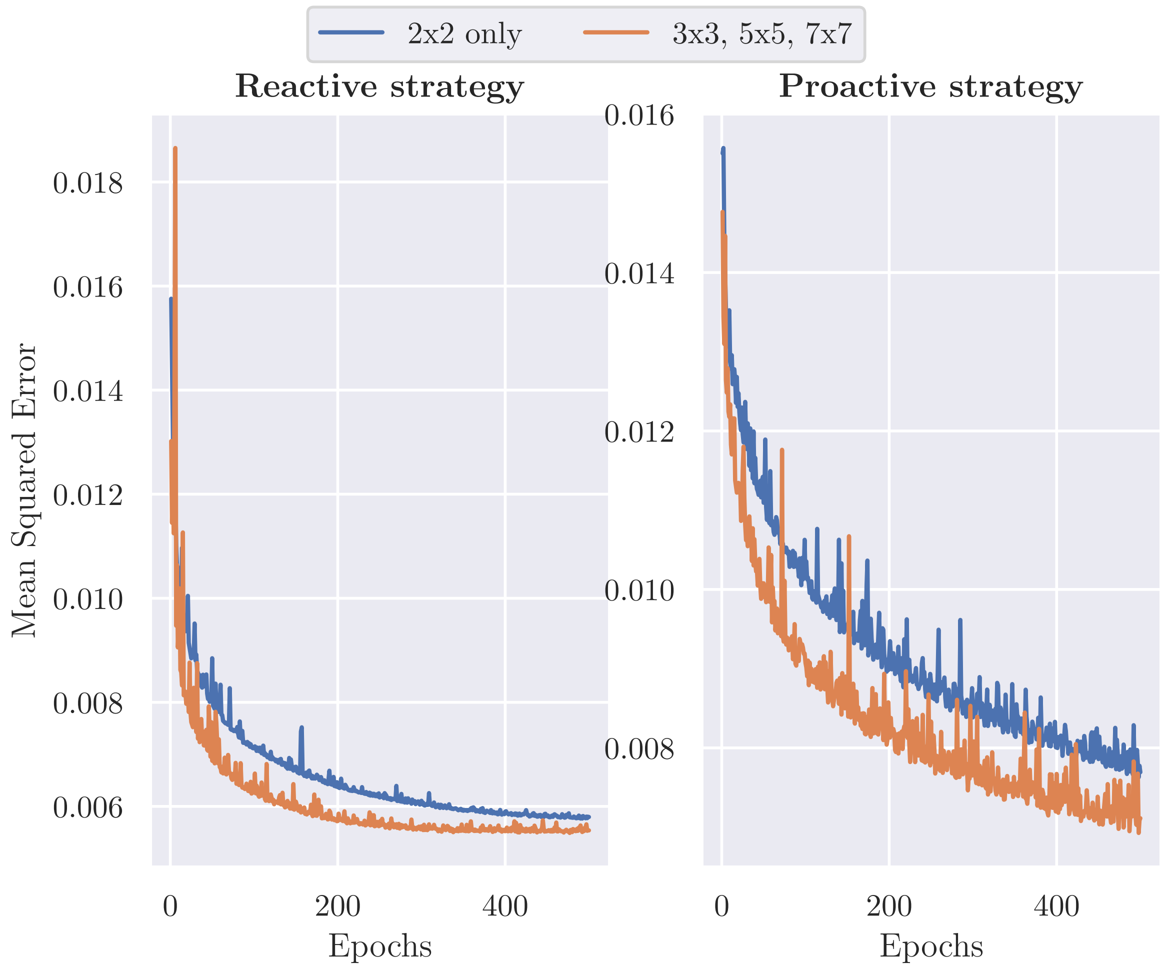


**Figure B.2:** Validation error during training for the network with original filter size structure and a filter size structure of 3x3, 5x5 and 7x7.

## B.3. Number of Layers

Another common used CNN feature is convolutional layers getting deeper with a factor of 2 when progressing deeper into the network. Most commonly a structure in which the convolutional layers have a depth of 32-64-128 and so on. As can be seen in figure B.3 it is found that using this structure with 2 layers does not decrease training performance. After this, a 3 layer architecture was evaluated which was found to increase convergence speed and predictive power. An effort to train a 4 layer architecture is also done. This training run was aborted due to substantial increase in training time.

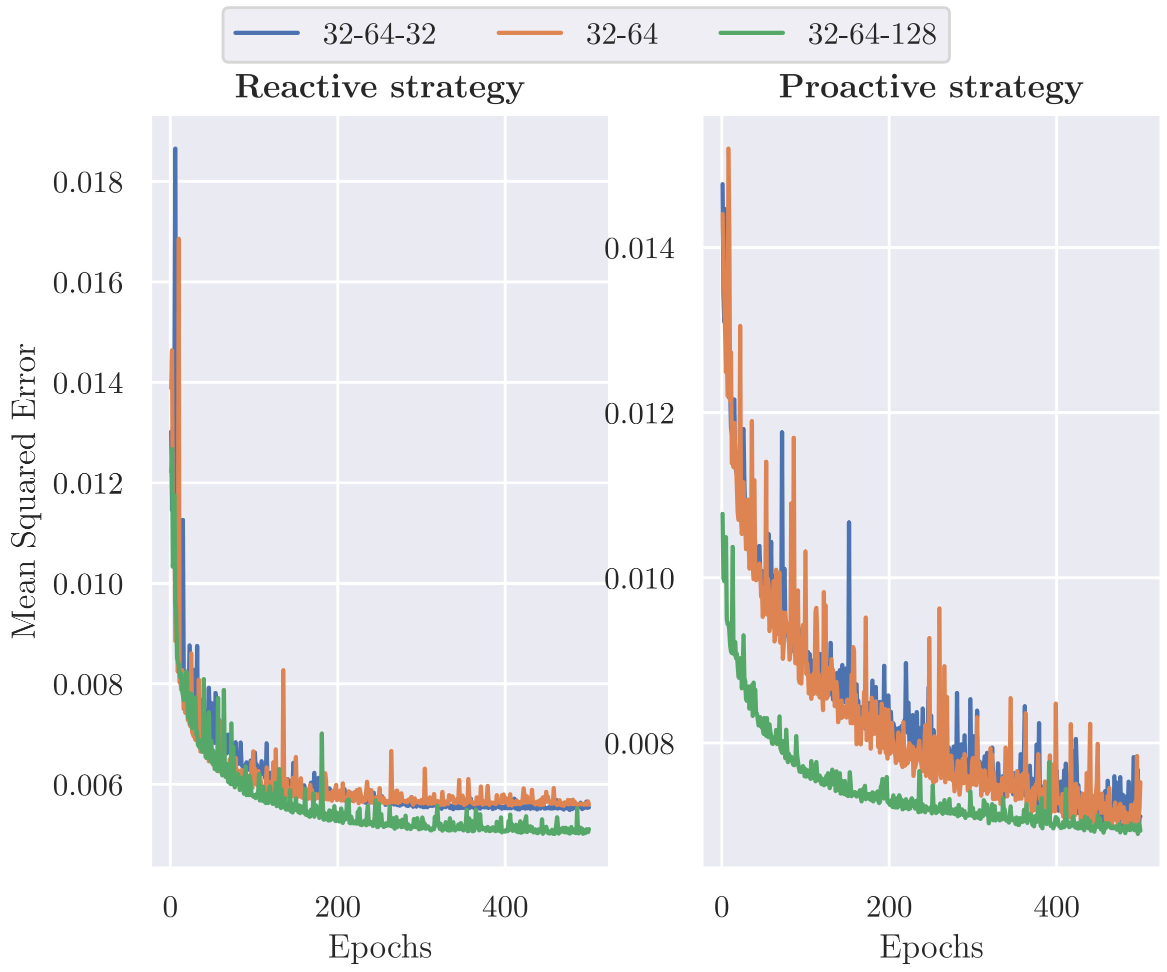


**Figure B.3:** Validation error during training for the architecture with different convolutional layer structures.

## B.4. Layer Depth

At this step, it was analysed whether shallower or deeper convolutional layers result in better training performance. Next to the original network, architecture using 16-32-64 and 64-128-256 structures were evaluated. As can be seen in figure B.4 the shallower networks shows worse performance in terms of convergence speed and minimal error for both strategies. An effort to evaluate the deeper network was done but similar to previous step aborted due to increased training time.

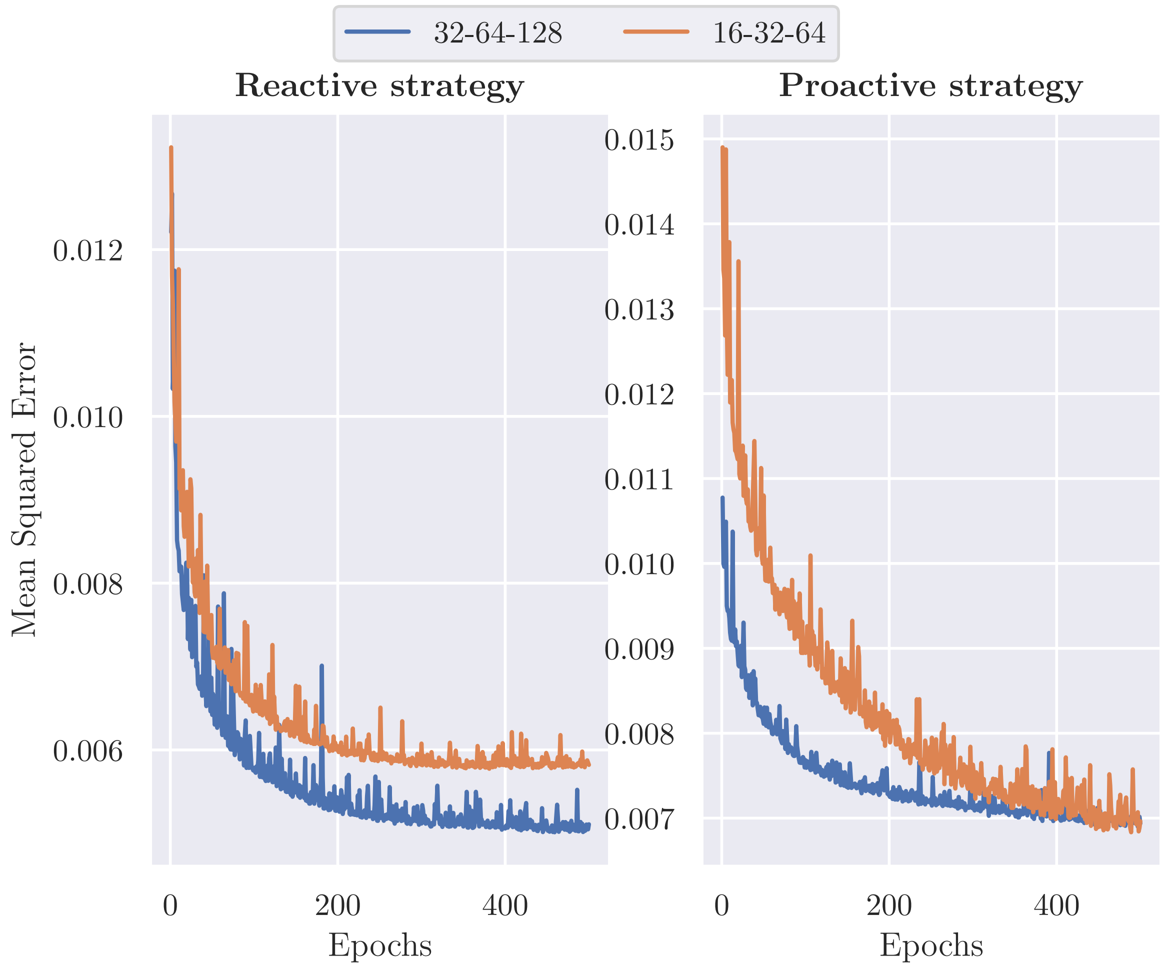


**Figure B.4:** Validation error during training for the original network and one with shallower convolutional layers.

## B.5. Dropout Rate

The dropout layer is placed just before output layer and randomly sets some of inputs to 0 at each epoch. The dropout refers to the percentage of inputs set to 0 related to the total input for that particular layer. Van Rooijen et al. [59] found that a dropout rate of 20% improves accuracy of the model. For completeness it is also evaluated for the model analysed in this research with dropout rates of 0%, 10%, 20% and 30%. As can be seen in figure B.5, a 20% results in faster convergence for the Proactive strategy and lower final error for the Reactive strategy.

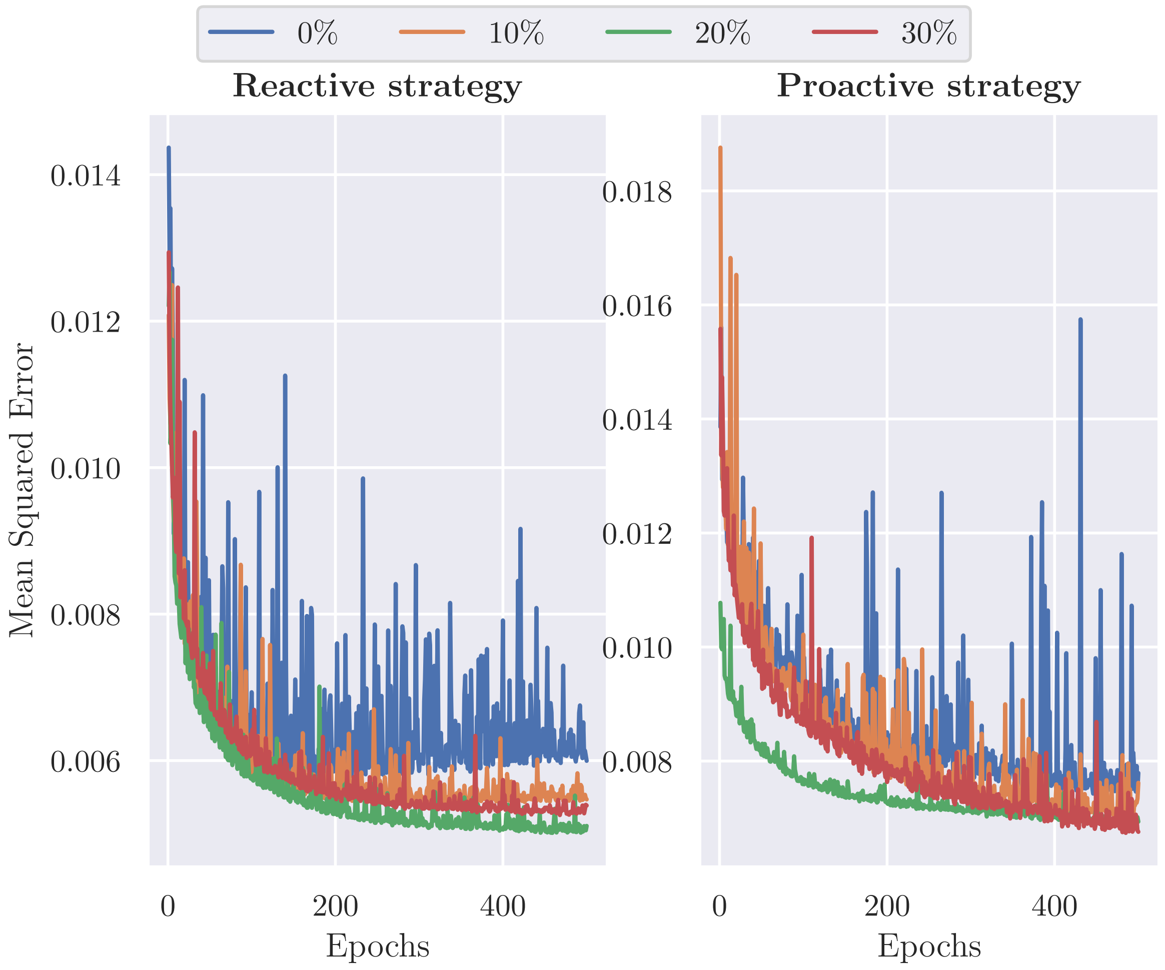


**Figure B.5:** Validation error during training for 0%, 10%, 20% and 30% dropout rate.

## B.6. Optimization Algorithm

At the final step two more elaborate optimization algorithms are evaluated. Figure B.6 shows that for both strategies RMSProp and Adam outperform SGD. The final architecture will be trained using Adam as it results in a slightly lower global MSE. However it can be seen that RMSProp and Adam have more variability and an increase in performance after a certain amount of epochs compared to SGD. Therefore, an early stopping condition is added in which the weight are saved if a global minimum is found and training will stop if no global minimum is found after 50 epochs.

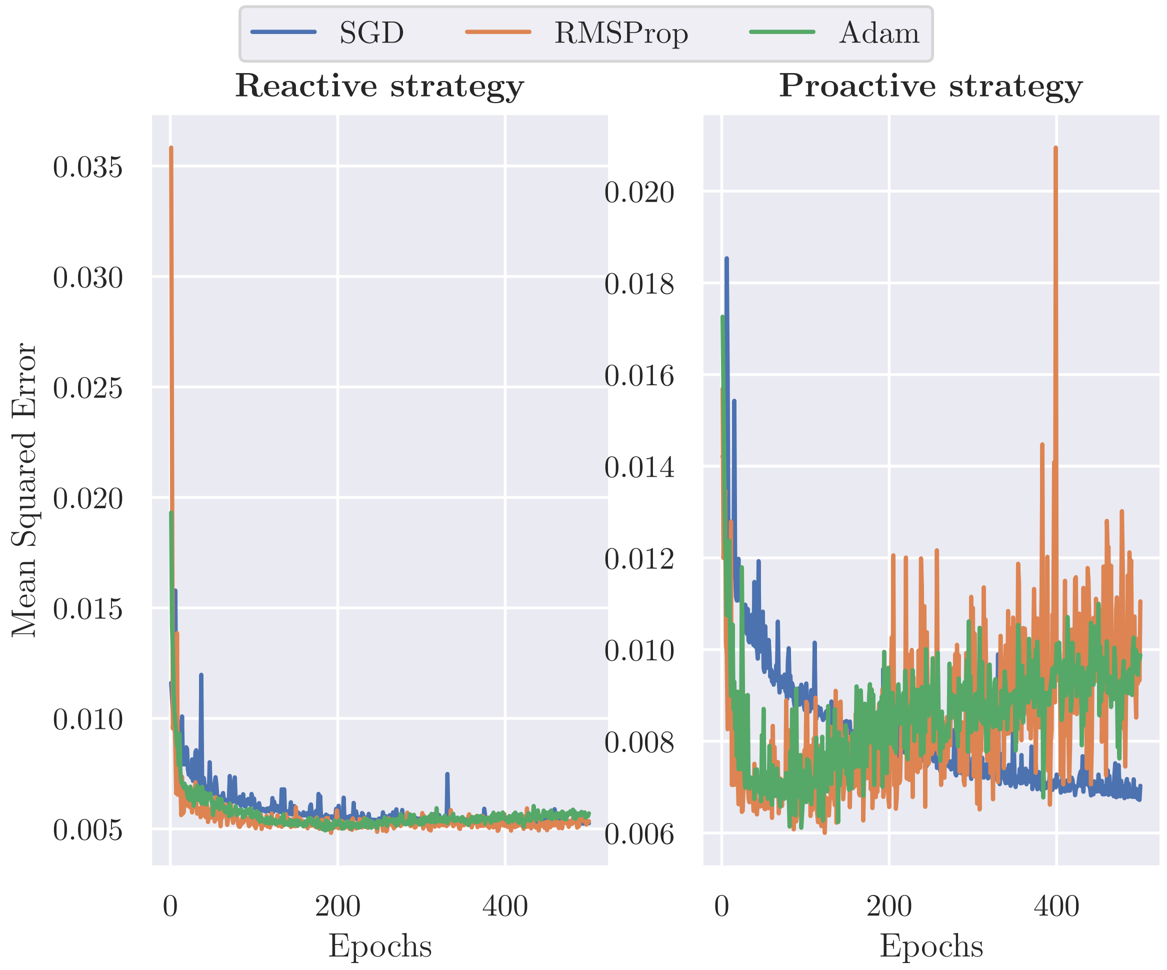


**Figure B.6:** Validation error during training for the final architecture trained with different optimization functions.

## B.7. Final Architecture

All steps described above have led to the CNN architecture presented in table B.1. With this network the training performance has significantly increased in terms of global MSE an convergence time as can be seen in figure B.7. Using the test data set evaluation has been done from which the true heading and speed error distributions are presented in the boxplots shown in figure B.8. It can be observed that the predictive performance is substantially decreased for both strategies in terms of true heading and speed error.

**Table B.1:** Final network architecture.

| Layer type | Input size | Filter size | Depth | Activation |
|---|---|---|---|---|
| Convolutional | 32x64x3 | 3x3 | 32 | ReLU |
| Max pool | 32x64x32 | | | |
| Convolutional | 16x32x32 | 5x5 | 64 | ReLU |
| Max pool | 16x32x64 | | | |
| Convolutional | 8x16x64 | 7x7 | 128 | ReLU |
| Flatten | 8x16x128 | | | |
| Fully connected | 2688 | | | ReLU |
| Dropout | 1024 | | | |
| Fully connected | 1024 | | | Linear |

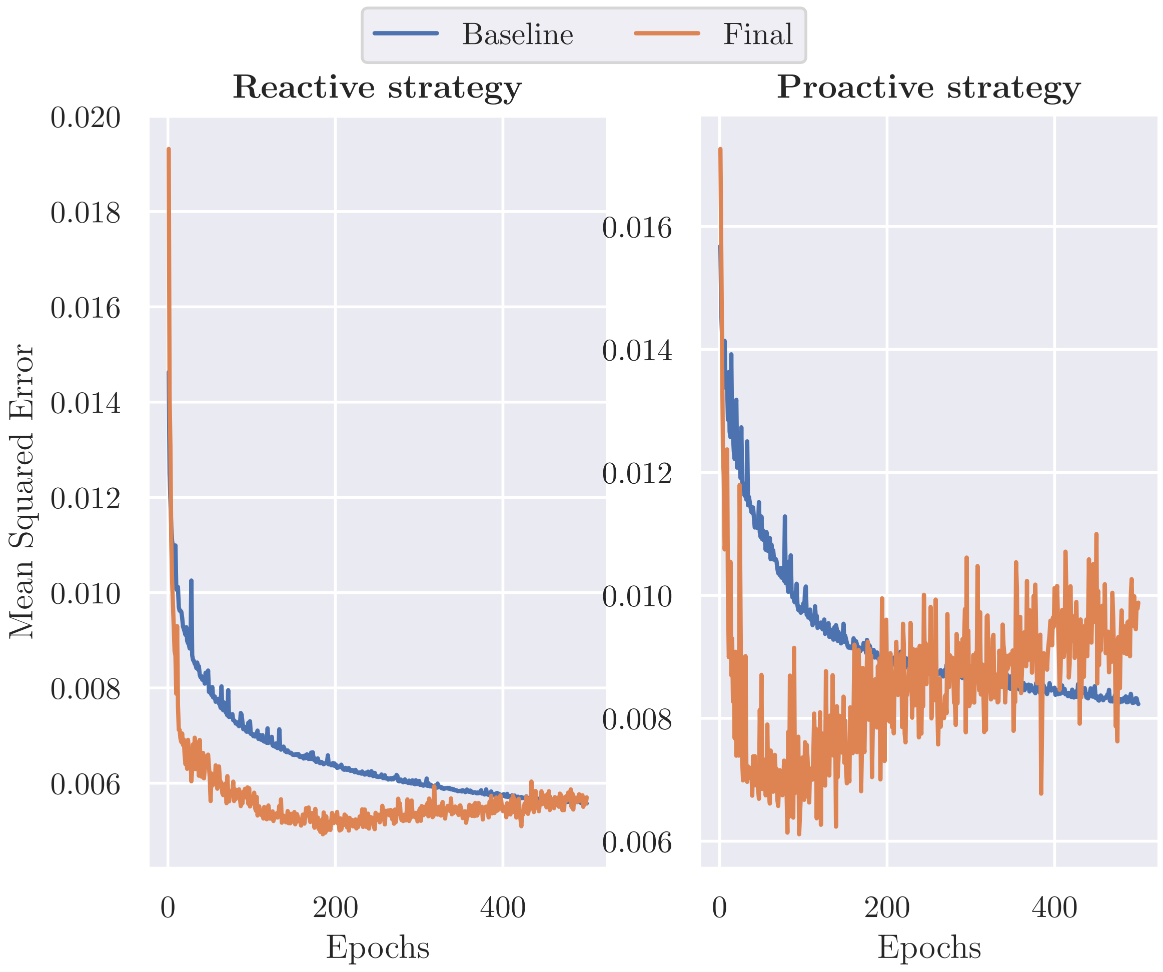


**Figure B.7:** Validation error during training for the final architecture trained compared to the initial architecture.

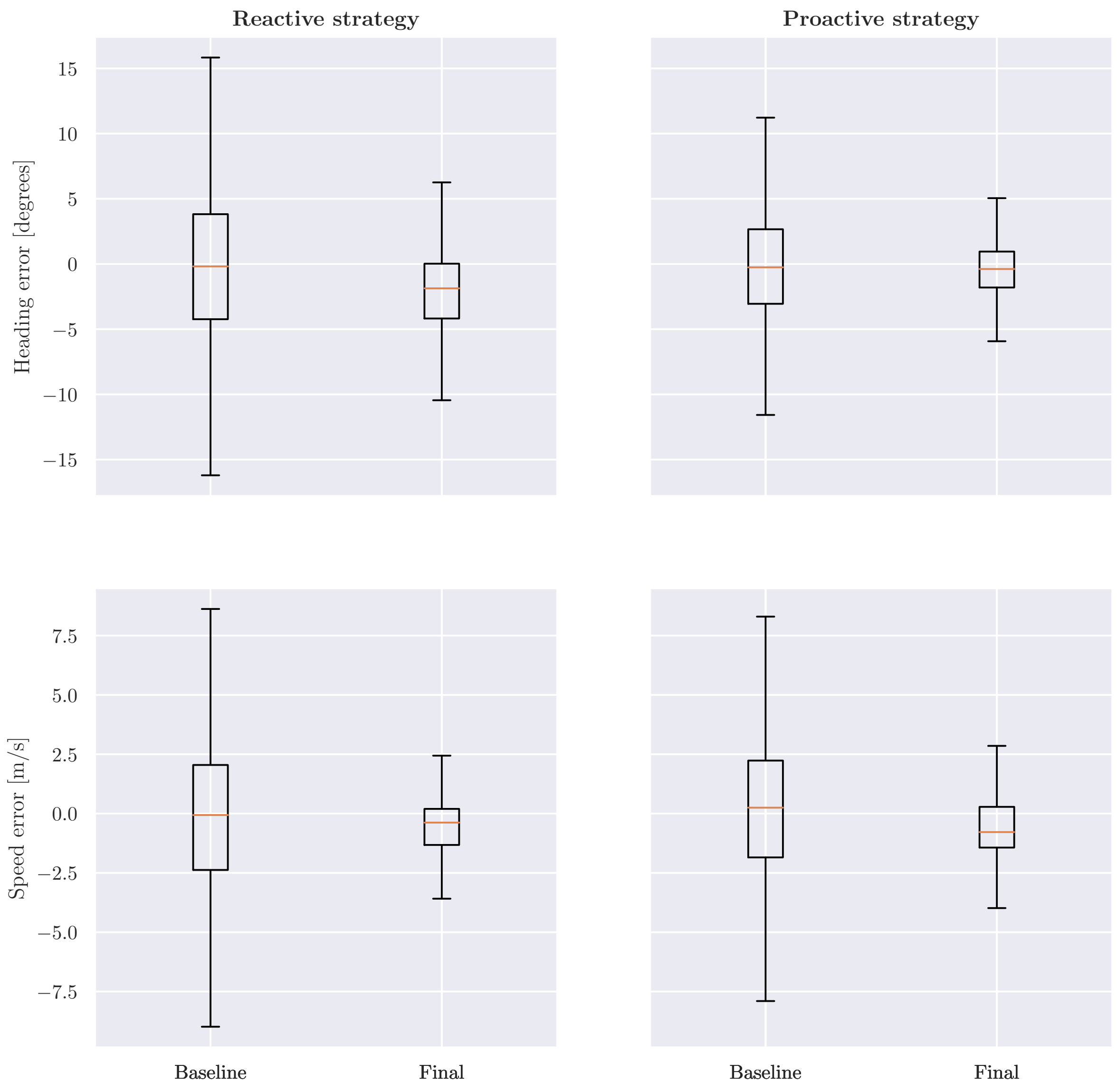


**Figure B.8:** True heading and speed error distribution comparison between the in initial and final architecture.

# C Feature Engineering

This chapter will present the SSD color schemes evaluated in this research with more detail. After this, the training curves corresponding to each dataset will be presented. Finally, evaluation of the SSD versions will be detailed using boxplots.

## C.1. color Schemes

Figures C.1, C.2, C.3, C.4 and C.5 detail the color schemes which use 1, 5, 10, 20 and 600 colors, respectively. From top to bottom, each figure shows an example SSD, the color scheme itself and pixel values for individual RGB colors plotted against $t_{los}$, respectively. The top example SSDs are generated using the same conflict situation.

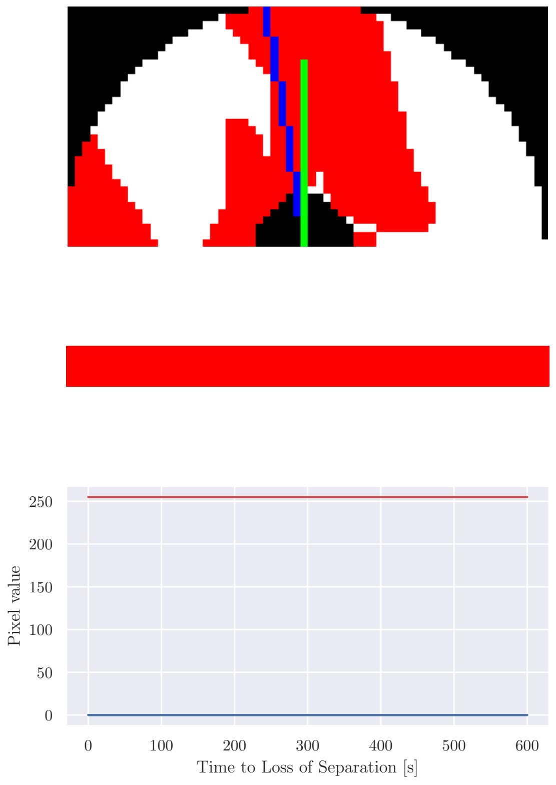


**Figure C.1:** Example SSD for the color scheme using 1 color, the color schemes itself and pixel values for individual RGB colors plotted against $t_{los}$.

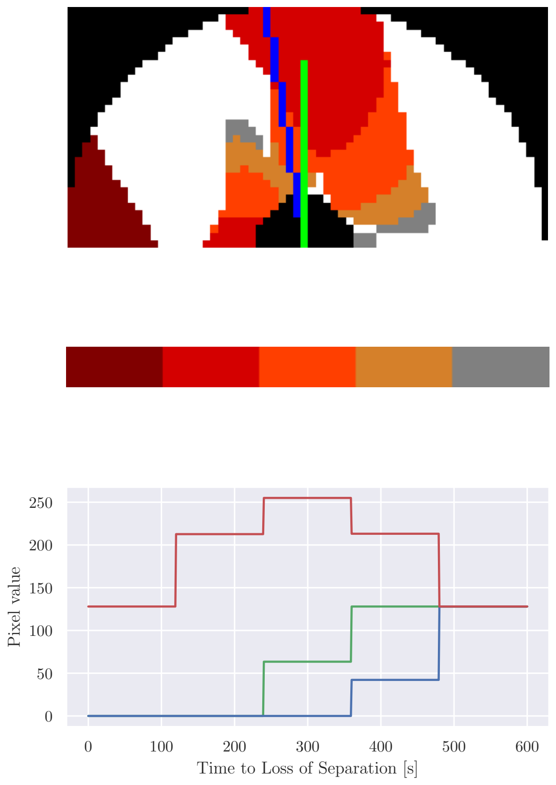


**Figure C.2:** Example SSD for the color scheme using 5 color, the color schemes itself and pixel values for individual RGB colors plotted against $t_{los}$.

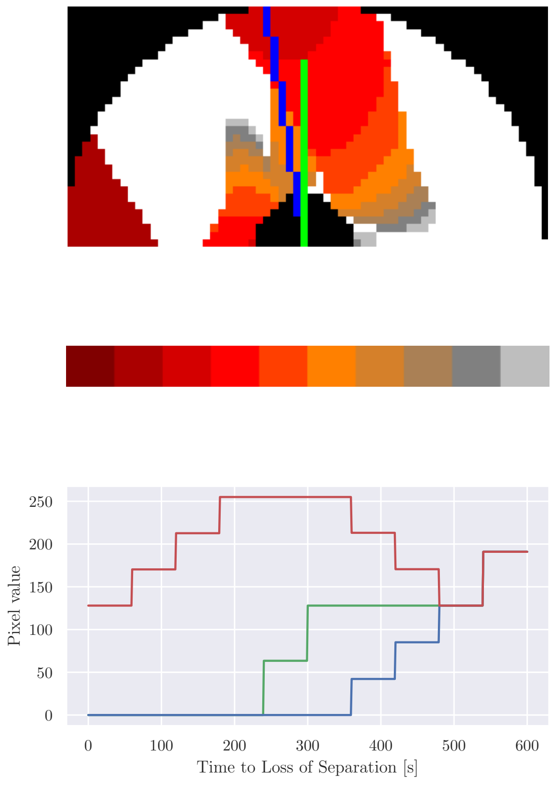


**Figure C.3:** Example SSD for the color scheme using 10 color, the color schemes itself and pixel values for individual RGB colors plotted against $t_{los}$.

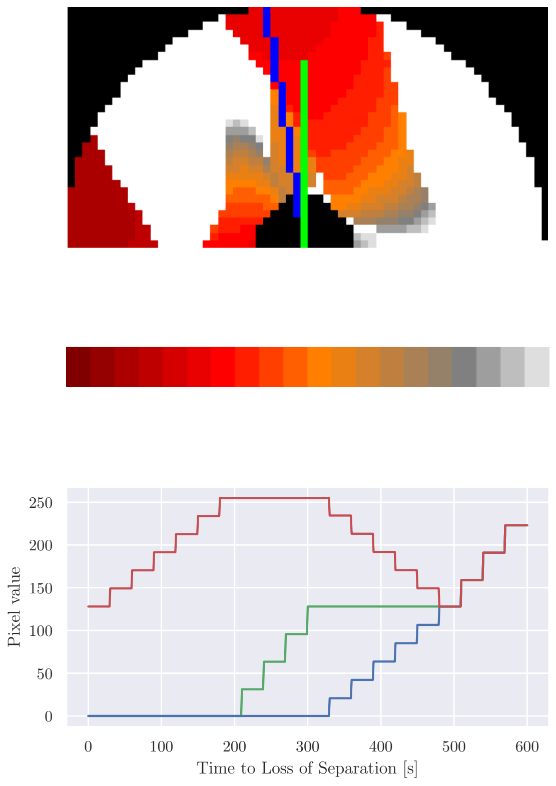


**Figure C.4:** Example SSD for the color scheme using 20 color, the color schemes itself and pixel values for individual RGB colors plotted against $t_{los}$.

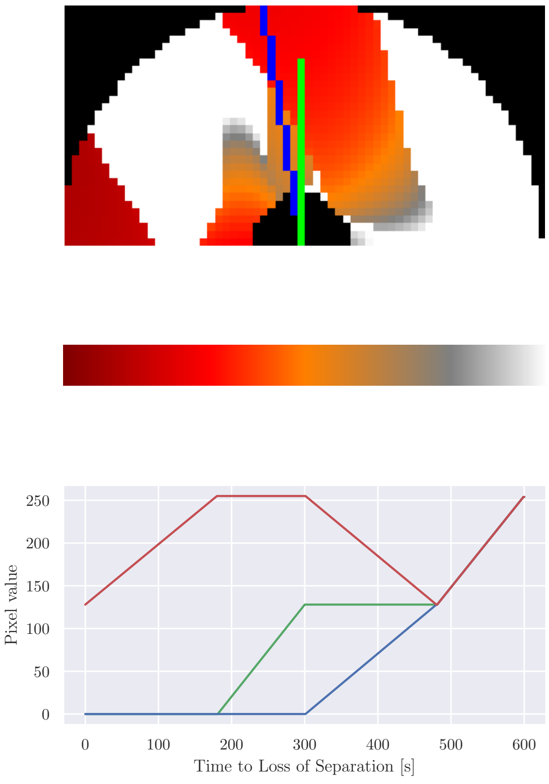


**Figure C.5:** Example SSD for the color scheme using 600 color, the color schemes itself and pixel values for individual RGB colors plotted against $t_{los}$.

## C.2. Evaluation

By feeding the test data set to the trained models for different color schemes variation their predictive power is evaluated. Figure C.6 presents the true heading and speed error distributions for both strategies in boxplots. It can be seen that adding more colors not necessarily results in better predictive performance. All color scheme variations, excluding the scheme with 1 color, show similar performance.

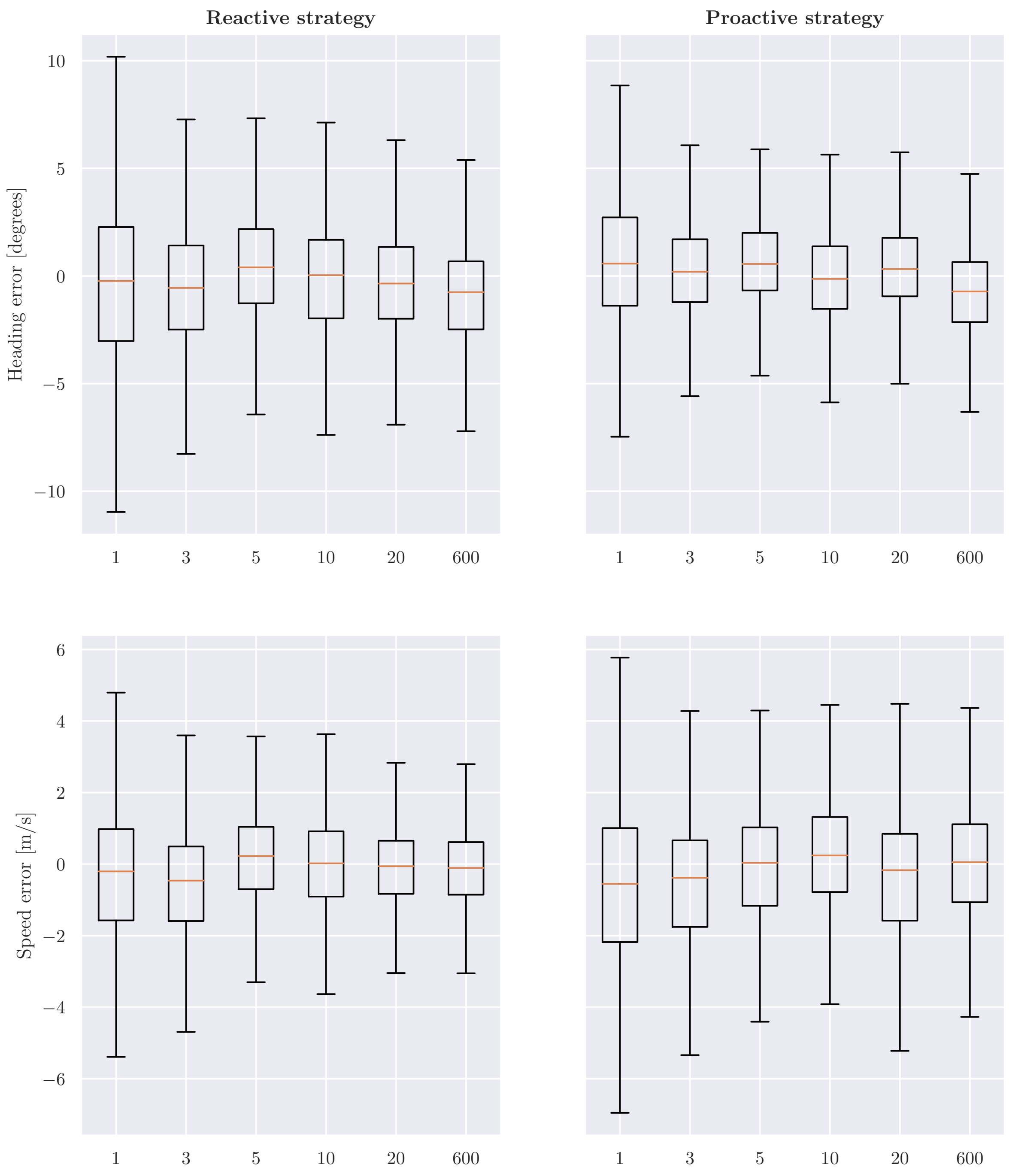


**Figure C.6:** Boxplots presenting true heading and speed error distribution for caused by training with different SSD color schemes for both strategies.

# D Model Generalization

The chapter details the model generalization analyse. It was was evaluated how a model behaves given data generated with a different ATC area with different conflict dynamics. First, a different ATC area was constructed in which the a new simulation can be conducted. After this, it has been evaluated whether the new data indeed has different conflict dynamics. The third section details how the resolutions are distributed by the MVP algorithm [28] after which the SSD complexity graphs will be analysed. Finally, the predictive performance of applying the model to data coming from the original or new simulation will be evaluated.

## D.1. ATC Area

Figure D.1 shows the ATC area during the model generalization analyses. It can be observed that the new area is identical to the original area except the direction of $A_{con}$'s trajectory. By altering the direction, the conflict dynamics will be different while all other variables remain the same.

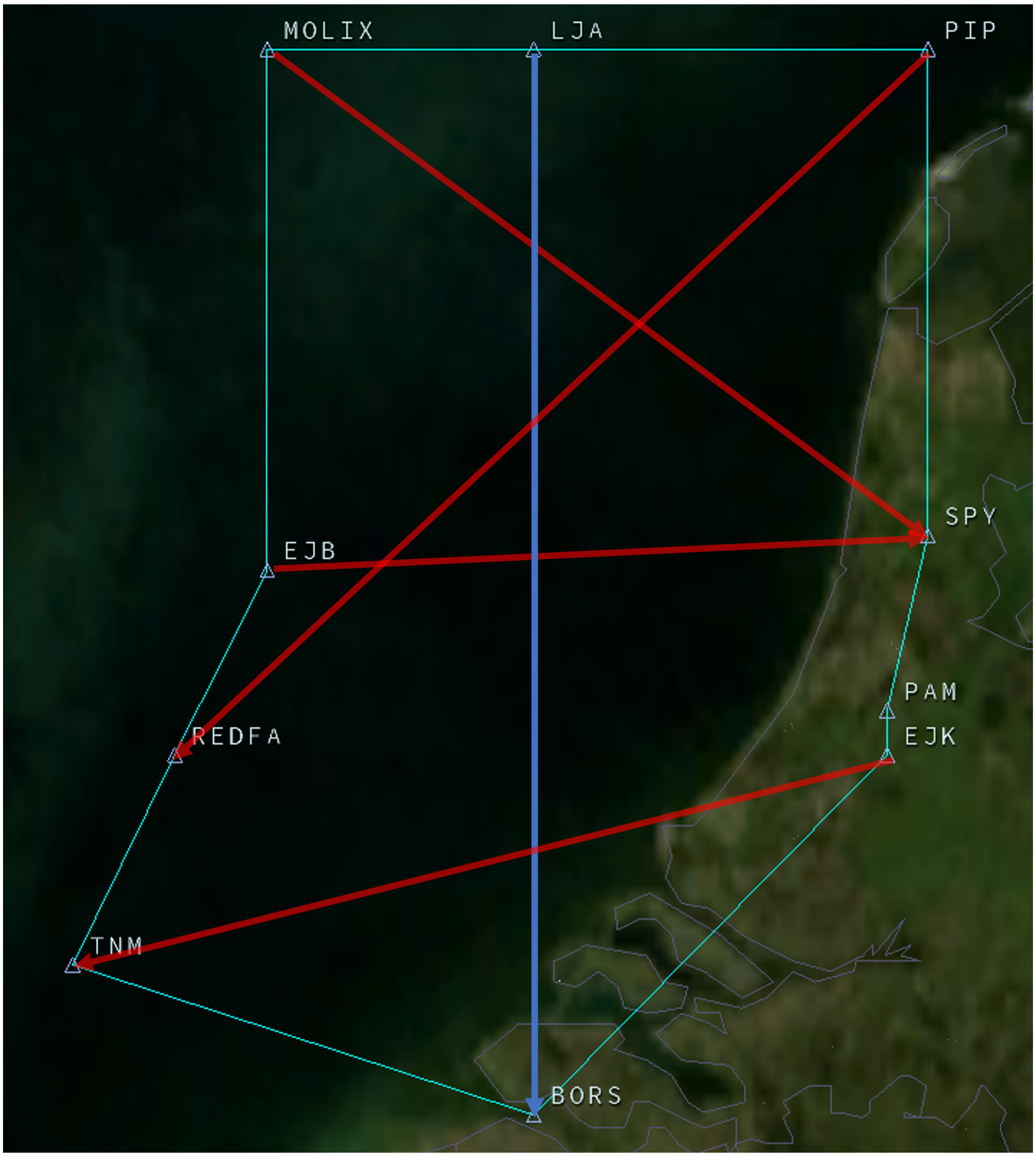


**Figure D.1:** ATC area used for the model generalization analyses.

## D.2. Conflict Angle

The conflict angles obtained during the model generalization simulation are presented by figure D.2. Comparing this figure with the conflict angels obtained by the original simulation (figure A.1), it can be observed from a qualitative view that conflict dynamics are different.

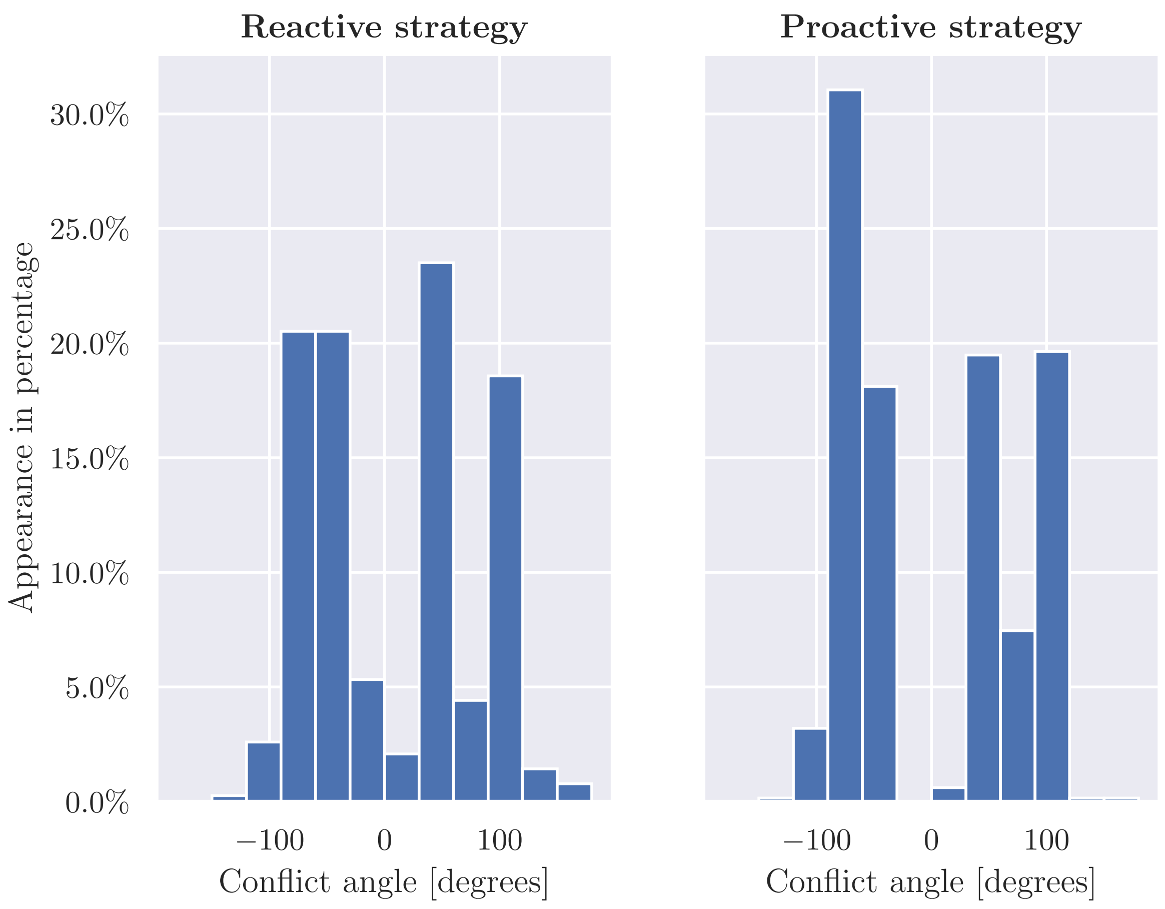


**Figure D.2:** Conflict angle distribution for the model generalization simulation for both strategies.

## D.3. MVP Strategy

Figure D.3 and D.4 present the distribution of resolution issued by the MVP algorithm [28] during the model generalization simulations for Reactive and Proactive strategy, respectively. Comparing these figures with figures A.2 and A.3 similarities can be found. It can be observed that the Proactive strategy uses smaller heading changes and larger changes in speed are less common.

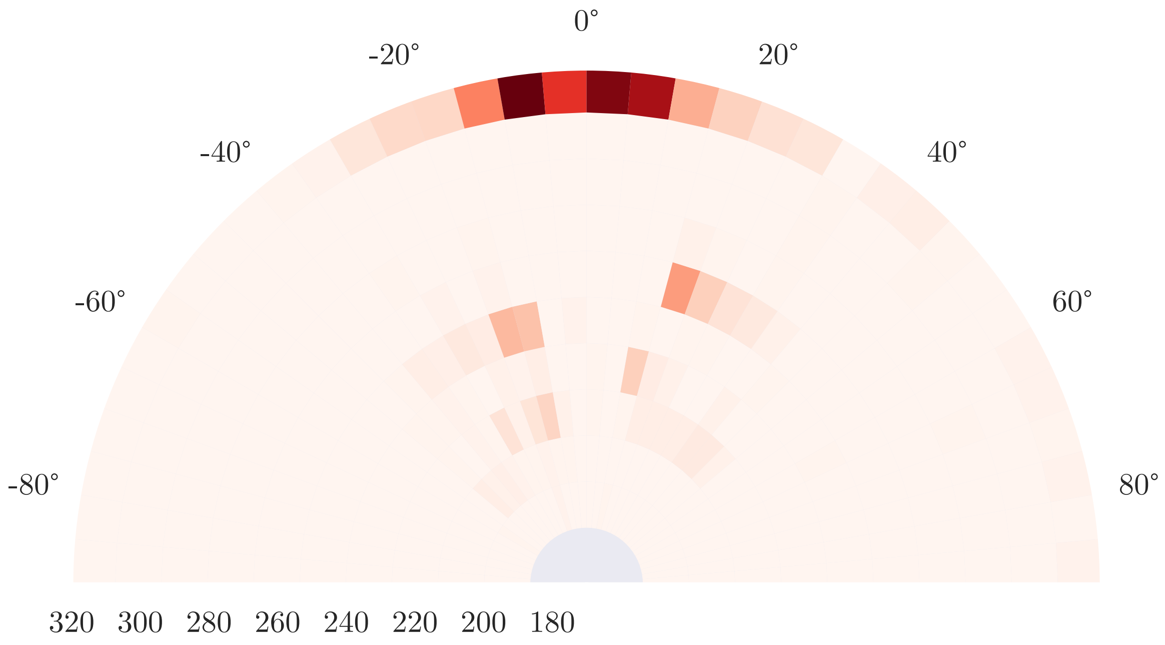


**Figure D.3:** Heat map of the resolution distribution for the Reactive strategy issued during the model generalization simulation.

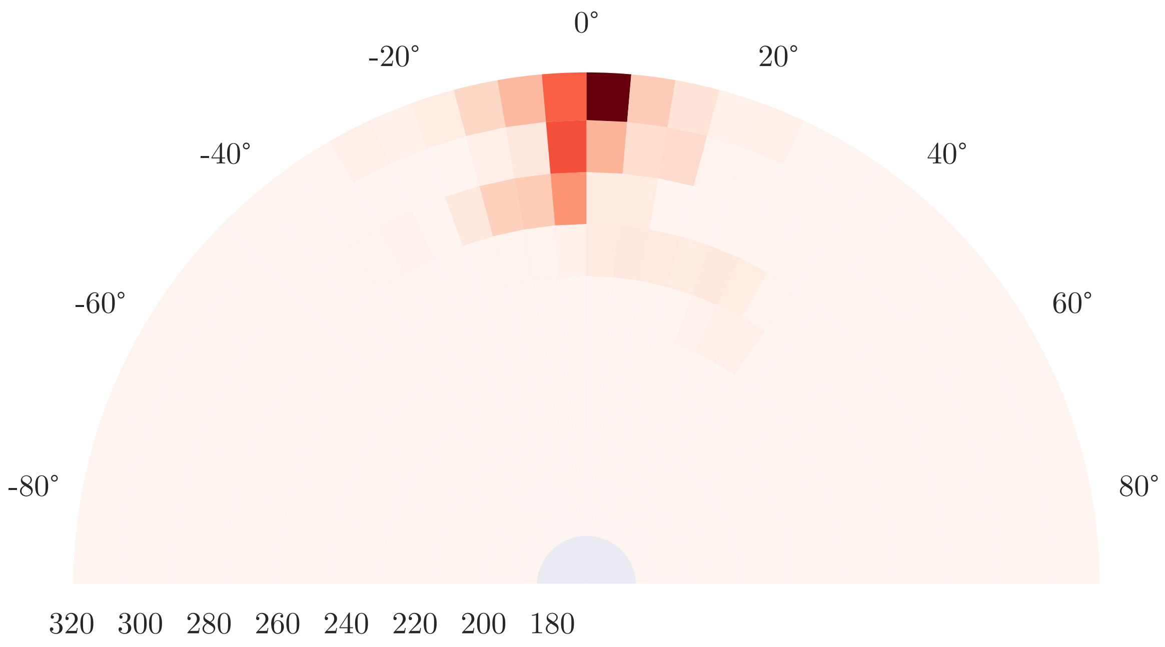


**Figure D.4:** Heat map of the resolution distribution for the Proactive strategy issued during the model generalization simulation

## D.4. SSD Complexity

The spread of SSD complexity over both data sets using the model generalization simulation is shown in figure D.5. The percentage of solution space pixels occupied by a FBZ is defined as the SSD complexity, e.g. if al solution space is occupied by FBZ the complexity is 100%. Only trajectories with LoS smaller than 600 seconds are included in the FBZ and no distinction in lower LoS is made with the calculation of these graphs. Comparing D.5 with figure A.4, it can be observed that dissimilarities exist between the distribution of SSd complexity for both strategies. This is another confirmation that conflict dynamics for the model generalization simulations are indeed different. The model generalization data sets generated for this analyses use the Zoomed SSD format using 3 colors to indicate different $t_{los}$.

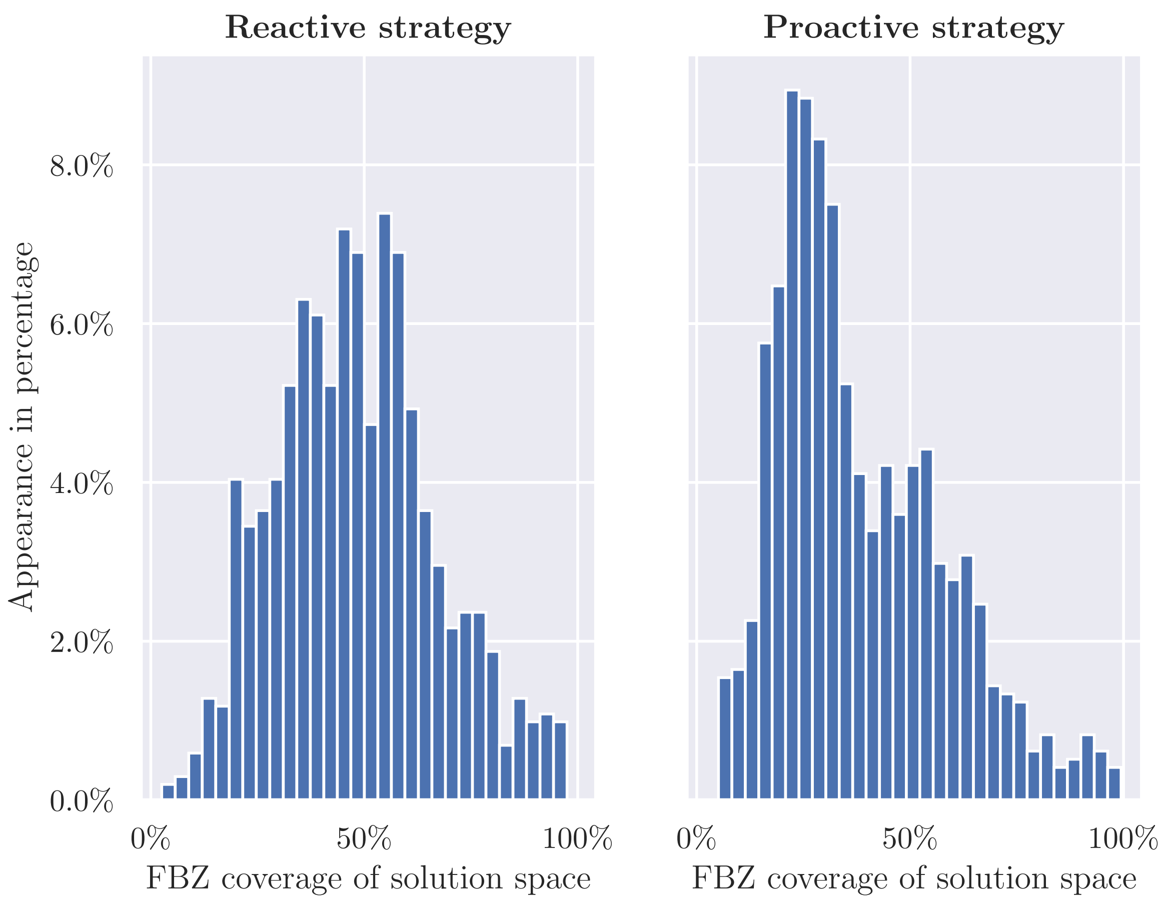


**Figure D.5:** Distribution off SSD complexity in the data sets generated by the model generalization simulations.

## D.5. Predictive Performance

In order to evaluate how the model generalizes to different conflict dynamics, the model generalization data sets are fed into the final model introduced in this research. Figure D.6 compares the true heading and speed error distribution for this evaluation and the final model introduced in this research. It can be observed that true heading and speed errors increase significantly when a data set is used with different conflict dynamics for both strategies. Hence, the model does not generalize well to different conflict dynamics.

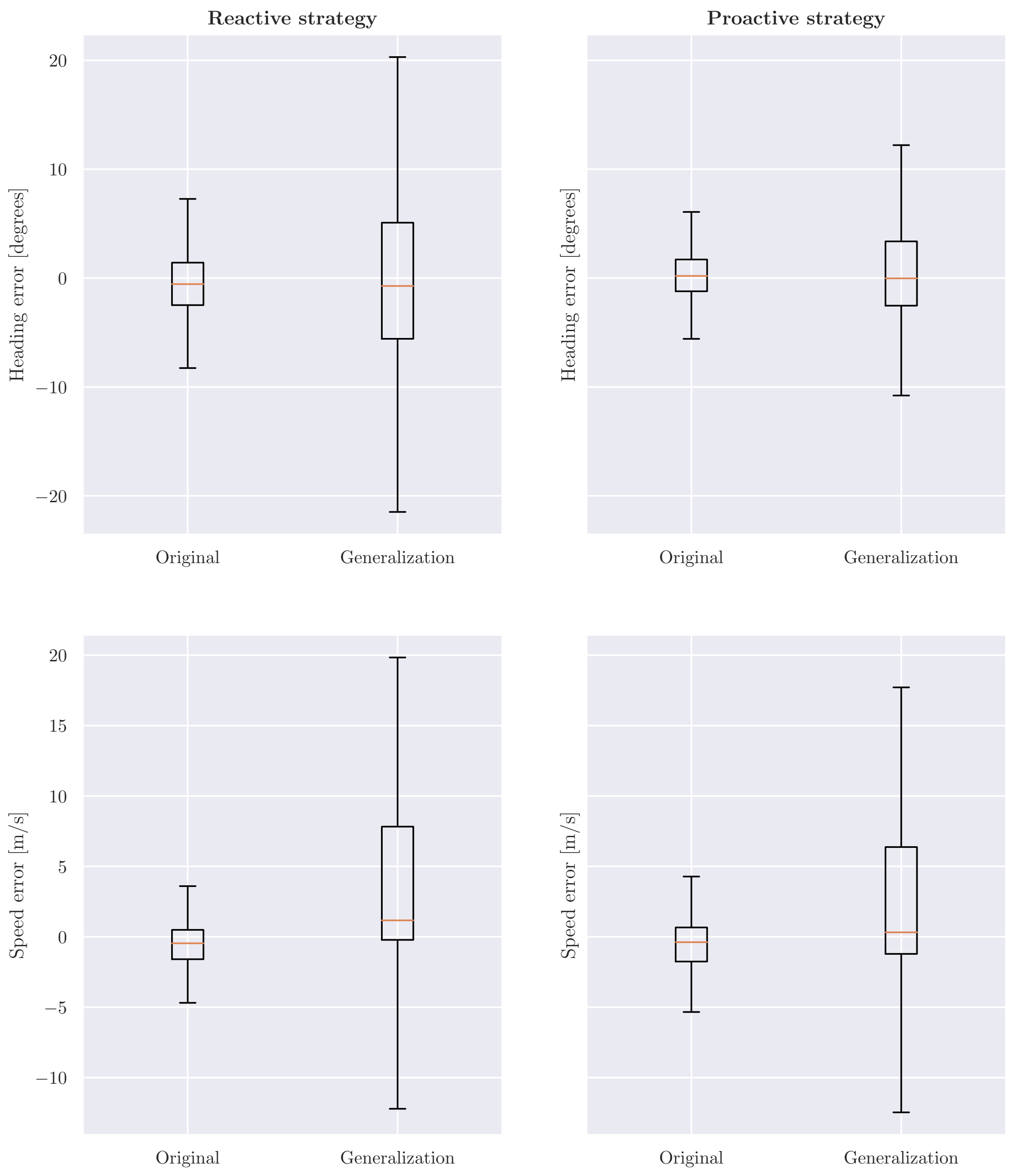


**Figure D.6:** True heading and speed error distribution for the generalization evaluation and the final model.

# Part IV

# Conclusions and Recommendations

# Conclusions and Recommendations

This research evaluates how an individual-sensitive advisory prediction model for Conflict Detection & Resolution can be constructed in order to establish an optimal human-machine system which will be accepted by its controller. To realize this goal, a literature review, a preliminary analysis and a simulation experiment is performed.

## Literature review

For an individual-sensitive advisory prediction model to be accepted by its controller, it should be accurate and present its output such that it is easy interpretable. Reviewing recent work, in which Solution space Diagrams were used as input to a set Convolutional Neural networks, it was found that opportunities exist to improve accuracy in terms of predictive performance. First of all, using regression task instead of classification could possibly predict exact resolutions instead of certain bins. Secondly, using an Solution Space format in which the solution space has a larger area may have a positive effect on predictive performance. Lastly, using a Cartesian Solution Space Diagram format low and high speed solution space can be analysed by the model with equal accuracy.

## Preliminary analysis

Using the Multi Voltage Potential Algorithm in the open-source air traffic simulator BlueSky, a simulation is done in which resolutions were issued in order to generate data. A Zoomed and Cartesian Solution Space Diagram formats with are introduced which have a maximized and Cartesian, respectively. With both formats, data sets are created which are fed to a single Convolutional Neural Network which predicts exact resolutions in the form of a velocity vector. Evaluation has shown that the model is able to accurately predict exact resolutions and that a larger solution space results in higher predictive performance. Additionally, the Cartesian format does not notably increase predictive performance compared to the Zoomed format while conformance with the mental model of the controller is lost.

## Experiment

For the final phase of this research it is decided to further expand on improving predictive performance. As the network architecture itself is not considered and more feature engineering options are not explored, a more elaborated computer simulation is executed using two different agents imitating different controller strategies. From this simulation multiple data sets are created consisting of Solution Space Diagrams which use color schemes with 1, 3, 5, 10, 20 and 600 different colors to indicate time to Loss of Separation. After architecture optimization and hyperparameter tuning it was found that all Solution Space Diagram color schemes, except the 1 color schemer, result in similar predictive performance. Evaluating the true heading and speed error for these schemes, average interquartile ranges of 3.4 degrees and 2 m/s are found, respectively. Therefore it can be concluded that the introduced model can predict exact resolutions to satisfying accuracy while adding more colors does not necessarily improve predictive performance. During the literature study it was found that higher accuracy, in terms of predictive performance, results in a higher chance for automation to be accepted by its controller.

## Recommendations

First of all, due to time constraints all simulations are done artificially. As it is assumed that the virtual agents used imitates a human controller flawlessly, all analyses are based on artificially generated data. The model is constructed such that it is optimized to predict the resolutions generated by the algorithm used. There is no guarantee that the model will achieve similar performance if human data is used. Future research could possibly test the model using professional air traffic controllers.

Secondly, all feature engineer applied to the Solution Space Diagram are designed in order to improve predictive performance of the model. The model and the controller are intended to work from a shared mental model meaning the same image will be presented to the controller. It is not yet analyzed how these changes to the Solution Space Diagram influence the readability as perceived by the human operator. Future research could possibly analyze how the Solution Space Diagrams proposed in this research influence the decision making of an Air Traffic controller.

Finally, the initial goal of this research is to increase controller acceptance by increasing predictive performance and presenting the output such that it is interpretable. Due to time constraints, the presentation of the prediction is not considered in this research. Making the model interpretable for its controller is a fundamental characteristic needed in order to achieve acceptance. Future research could possibly experiment with the Explainable Artificial Intelligence techniques considered in the literature review and design a Conflict Detection & Resolution interface accordingly.

# References

[1] Radhakrishna Achanta et al. “SLIC superpixels compared to state-of-the-art superpixel methods”. In: *IEEE transactions on pattern analysis and machine intelligence* 34.11 (2012), pp. 2274–2282.

[2] Amina Adadi and Mohammed Berrada. “Peeking inside the black-box: A survey on Explainable Artificial Intelligence (XAI)”. In: *IEEE Access* 6 (2018), pp. 52138–52160.

[3] Adrian Agogino and Kagan Tumer. “A multiagent approach to managing air traffic flow”. In: *Autonomous Agents and Multi-Agent Systems* 24 (Jan. 2012), pp. 1–25. DOI: `10.1007/s10458-010-9142-5`.

[4] Alexandr Andoni et al. “Practical and optimal LSH for angular distance”. In: *Advances in neural information processing systems*. 2015, pp. 1225–1233.

[5] Alejandro Barredo Arrieta et al. “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI”. In: *Information Fusion* 58 (2020), pp. 82–115.

[6] Lisanne Bainbridge. “Ironies of automation”. In: *Automatica* 19.6 (1983), pp. 775–779. ISSN: 0005-1098. DOI: `https://doi.org/10.1016/0005-1098(83)90046-8`. URL: `http://www.sciencedirect.com/science/article/pii/0005109883900468`.

[7] Marek Bekier, Brett R.C. Molesworth, and Ann Williamson. “Tipping point: The narrow path between automation acceptance and rejection in air traffic management”. In: *Safety Science* 50.2 (2012), pp. 259–265. ISSN: 0925-7535. DOI: `https://doi.org/10.1016/j.ssci.2011.08.059`. URL: `http://www.sciencedirect.com/science/article/pii/S092575351100230X`.

[8] Clark Borst, John M Flach, and Joost Ellerbroek. “Beyond ecological interface design: Lessons from concerns and misconceptions”. In: *IEEE Transactions on Human-Machine Systems* 45.2 (2014), pp. 164–175.

[9] Clark Borst et al. “Ecological interface design: supporting fault diagnosis of automated advice in a supervisory air traffic control task”. In: *Cognition, Technology & Work* 19.4 (2017), pp. 545–560.

[10] Léon Bottou. “Online learning and stochastic approximations”. In: *On-line learning in neural networks* 17.9 (1998), p. 142.

[11] Jason Brownlee. “Discover feature engineering, how to engineer features and how to get good at it”. In: *Machine Learning Process* (2014).

[12] Dan CireşAn et al. “Multi-column deep neural network for traffic sign classification”. In: *Neural networks* 32 (2012), pp. 333–338.

[13] Leonardo L. B. V. Cruciol et al. “Air holding problem solving with reinforcement learning to reduce airspace congestion”. In: *Journal of Advanced Transportation* 49.5 (2015), pp. 616–633. DOI: `10.1002/atr.1293`. eprint: `https://onlinelibrary.wiley.com/doi/pdf/10.1002/atr.1293`. URL: `https://onlinelibrary.wiley.com/doi/abs/10.1002/atr.1293`.

[14] Jurriaan d’Engelbronner et al. “The use of the dynamic solution space to assess air traffic controller workload”. In: *AIAA Guidance, Navigation, and Control Conference*. 2010, p. 7542.

[15] Sanjoy Dasgupta, Charles F Stevens, and Saket Navlakha. “A neural algorithm for a fundamental computing problem”. In: *Science* 358.6364 (2017), pp. 793–796.

[16] Tamraparni Dasu and Theodore Johnson. *Exploratory data mining and data cleaning*. Vol. 479. John Wiley & Sons, 2003.

[17] Pedro Domingos. “A few useful things to know about machine learning”. In: *Communications of the ACM* 55.10 (2012), pp. 78–87.

[18] John Duchi, Elad Hazan, and Yoram Singer. “Adaptive subgradient methods for online learning and stochastic optimization.” In: *Journal of machine learning research* 12.7 (2011).

[19] Joost Ellerbroek et al. “Experimental evaluation of a coplanar airborne separation display”. In: *IEEE Transactions on Human-Machine Systems* 43.3 (2013), pp. 290–301.

[20] Feng Ning et al. “Toward automatic phenotyping of developing embryos from videos”. In: *IEEE Transactions on Image Processing* 14.9 (2005), pp. 1360–1371. DOI: `10.1109/TIP.2005.852470`.

[21] Philipp Fischer, Alexey Dosovitskiy, and Thomas Brox. “Image orientation estimation with convolutional networks”. In: *German Conference on Pattern Recognition*. Springer. 2015, pp. 368–378.

[22] John M Flach et al. “An ecological approach to interface design”. In: *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*. Vol. 42. SAGE Publications Sage CA: Los Angeles, CA. 1998, pp. 295–299.

[23] J Fürnkranz et al. “Mean squared error”. In: *Encyclopedia of Machine Learning* (2010).

[24] Raia Hadsell et al. “Learning long-range vision for autonomous off-road driving”. In: *Journal of Field Robotics* 26.2 (2009), pp. 120–144.

[25] Patrick Hermes et al. “Solution-space-based complexity analysis of the difficulty of aircraft merging tasks”. In: *Journal of Aircraft* 46.6 (2009), pp. 1995–2015.

[26] Brian Hilburn, Carl Westin, and Clark Borst. “Will controllers accept a machine that thinks like they think? The role of strategic conformance in decision aiding automation”. In: *Air Traffic Control Quarterly* 22.2 (2014), pp. 115–136.

[27] Jacco Hoekstra and Joost Ellerbroek. “BlueSky ATC Simulator Project: An Open Data and Open Source Approach”. English. In: *7th International Conference on Research in Air Transportation*. 7th International Conference on Research in Air Transportation, ICRAT’16 ; Conference date: 20-06-2016 Through 24-06-2016. 2016. URL: `http://www.icrat.org/icrat/index.cfm,%20http://www.icrat.org/icrat/7th-international-conference/`.

[28] Jacco M Hoekstra and Joost Ellerbroek. “Bluesky ATC simulator project: an open data and open source approach”. In: *Proceedings of the 7th International Conference on Research in Air Transportation*. Vol. 131. FAA/Eurocontrol USA/Europe. 2016, p. 132.

[29] Jacco M Hoekstra, Ronald NHW van Gent, and Rob CJ Ruigrok. “Designing for safety: the ‘free flight’air traffic management concept”. In: *Reliability Engineering & System Safety* 75.2 (2002), pp. 215–232.

[30] Kevin Hoff and Masooda Bashir. “Trust in Automation: Integrating Empirical Evidence on Factors That Influence Trust”. In: *Human Factors The Journal of the Human Factors and Ergonomics Society* 57 (May 2015), pp. 407–434. DOI: `10.1177/0018720814547570`.

[31] Erik Hollnagel. “Information and reasoning in intelligent decision support systems”. In: *International Journal of Man-Machine Studies* 27.5-6 (1987), pp. 665–678.

[32] ICAO. *Doc 4444 - Procedure for Air Navigation Services - Air traffic Management*. 999 Robert-Bourassa Boulevard, Montréal, Quebec, Canada: ICAO, 2016.

[33] Kouroush Jenab and Joseph Pineau. “Automation of Air Traffic Management Using Fuzzy Logic Algorithm to Integrate Unmanned Aerial Systems into the National Airspace”. In: *International Journal of Electrical and Computer Engineering (IJECE)* 8 (Oct. 2018), p. 3169. DOI: `10.11591/ijece.v8i5.pp3169-3178`.

[34] Yazdi I Jenie et al. “Selective velocity obstacle method for deconflicting maneuvers applied to unmanned aerial vehicles”. In: *Journal of Guidance, Control, and Dynamics* 38.6 (2015), pp. 1140–1146.

[35] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).

[36] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. “Imagenet classification with deep convolutional neural networks”. In: *Communications of the ACM* 60.6 (2017), pp. 84–90.

[37] J. K. Kuchar and L. C. Yang. “A review of conflict detection and resolution modeling methods”. In: *IEEE Transactions on Intelligent Transportation Systems* 1.4 (2000), pp. 179–189. DOI: `10.1109/6979.898217`.

[38] Steve Lawrence et al. “Face recognition: A convolutional neural-network approach”. In: *IEEE transactions on neural networks* 8.1 (1997), pp. 98–113.

[39] Y. Lecun et al. “Gradient-based learning applied to document recognition”. In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324. DOI: `10.1109/5.726791`.

[40] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. “Deep learning”. In: *nature* 521.7553 (2015), pp. 436–444.

[41] Yann LeCun et al. “Backpropagation applied to handwritten zip code recognition”. In: *Neural computation* 1.4 (1989), pp. 541–551.

[42] John D. Lee and Katrina A. See. “Trust in Automation: Designing for Appropriate Reliance”. In: *Human Factors* 46.1 (2004). PMID: 15151155, pp. 50–80. DOI: `10.1518/hfes.46.1.50\_30392`. eprint: `https://doi.org/10.1518/hfes.46.1.50_30392`. URL: `https://doi.org/10.1518/hfes.46.1.50_30392`.

[43] Min Lin, Qiang Chen, and Shuicheng Yan. “Network in network”. In: *arXiv preprint arXiv:1312.4400* (2013).

[44] S Locklin. *Neglected machine learning ideas*. 2017.

[45] Siddharth Mahendran, Haider Ali, and René Vidal. “3d pose regression using convolutional neural networks”. In: *Proceedings of the IEEE International Conference on Computer Vision Workshops*. 2017, pp. 2174–2182.

[46] Arnab Majumdar et al. “The Factors Affecting Airspace Capacity in Europe: A Cross-Sectional Time-Series Analysis Using Simulated Controller Workload Data”. In: *Journal of Navigation* 57 (Sept. 2004), pp. 385–405. DOI: `10.1017/S0373463304002863`.

[47] Gustavo Mercado-Velasco, Max Mulder, and Marinus Van Paassen. “Analysis of Air Traffic Controller Workload Reduction Based on the Solution Space for the Merging Task”. In: *AIAA Guidance, Navigation, and Control Conference* (2010). DOI: `10.2514/6.2010-7541`. eprint: `https://arc.aiaa.org/doi/pdf/10.2514/6.2010-7541`. URL: `https://arc.aiaa.org/doi/abs/10.2514/6.2010-7541`.

[48] Tom M Mitchell et al. “Machine learning. 1997”. In: *Burr Ridge, IL: McGraw Hill* 45.37 (1997), pp. 870–877.

[49] Urs Muller et al. “Off-road obstacle avoidance through end-to-end learning”. In: *Advances in neural information processing systems*. 2006, pp. 739–746.

[50] W James Murdoch et al. “Interpretable machine learning: definitions, methods, and applications”. In: *arXiv preprint arXiv:1901.04592* (2019).

[51] M. Nguyen-Duc, J. -. Briot, and A. Drogoul. “An application of multi-agent coordination techniques in air traffic management”. In: *IEEE/WIC International Conference on Intelligent Agent Technology, 2003. IAT 2003.* 2003, pp. 622–625. DOI: `10.1109/IAT.2003.1241159`.

[52] International Civil Aviation Organization. *Facts and figures*. 2019. URL: `https://www.icao.int/sustainability/Pages/Facts-Figures_WorldEconomyData.aspx`.

[53] Nicolas Papernot and Patrick McDaniel. “Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning”. In: *arXiv preprint arXiv:1803.04765* (2018).

[54] Forough Poursabzi-Sangdeh et al. “Manipulating and measuring model interpretability”. In: *arXiv preprint arXiv:1802.07810* (2018).

[55] Jens Rasmussen. “Skills, rules, and knowledge; signals, signs, and symbols, and other distinctions in human performance models”. In: *IEEE transactions on systems, man, and cybernetics* (1983), pp. 257–266.

[56] Robert Regtuit, Clark Borst, and Erik-Jan Van Kampen. “Building Strategic Conformal Automation for Air Traffic Control Using Machine Learning”. In: *2018 AIAA Information Systems-AIAA Infotech @ Aerospace* (2018). DOI: `10.2514/6.2018-0074`. eprint: `https://arc.aiaa.org/doi/pdf/10.2514/6.2018-0074`. URL: `https://arc.aiaa.org/doi/abs/10.2514/6.2018-0074`.

[57] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. “” Why should I trust you?” Explaining the predictions of any classifier”. In: *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016, pp. 1135–1144.

[58] Herbert Robbins and Sutton Monro. "A stochastic approximation method". In: *The annals of mathematical statistics* (1951), pp. 400–407.

[59] S. Rooijen et al. "Toward Individual-Sensitive Automation for Air Traffic Control Using Convolutional Neural Networks". In: *Journal of Air Transportation* (May 2020), pp. 1–9. DOI: `10.2514/1.D0180`.

[60] Stanley N Roscoe. "Airborne displays for flight and navigation". In: *Human Factors* 10.4 (1968), pp. 321–332.

[61] Frank Rosenblatt. "The perceptron: a probabilistic model for information storage and organization in the brain." In: *Psychological review* 65.6 (1958), p. 386.

[62] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. "Learning representations by back-propagating errors". In: *nature* 323.6088 (1986), pp. 533–536.

[63] Thomas L. Seamster et al. "Cognitive Task Analysis of Expertise in Air Traffic Control". In: *The International Journal of Aviation Psychology* 3.4 (1993), pp. 257–283. DOI: `10.1207/s15327108ijap0304\_2`. eprint: `https://doi.org/10.1207/s15327108ijap0304_2`. URL: `https://doi.org/10.1207/s15327108ijap0304_2`.

[64] Ramprasaath R Selvaraju et al. "Grad-cam: Visual explanations from deep networks via gradient-based localization". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 618–626.

[65] Pierre Sermanet et al. "Pedestrian Detection with Unsupervised Multi-stage Feature Learning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2013.

[66] Sagar Sharma. "Activation functions in neural networks". In: *Towards Data Science* 6 (2017).

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

[68] Statista. *Revenues from the artificial intelligence for enterprise applications market worldwide, from 2016 to 2025*. 2016. URL: `https://www.statista.com/statistics/607716/worldwide-artificial-intelligence-market-revenues/`. (accessed: 26.11.2020).

[69] Pierre Stock and Moustapha Cisse. "Convnets and imagenet beyond accuracy: Understanding mistakes and uncovering biases". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 498–512.

[70] Richard Sutton. "Two problems with back propagation and other steepest descent learning procedures for networks". In: *Proceedings of the Eighth Annual Conference of the Cognitive Science Society, 1986*. 1986, pp. 823–832.

[71] Christian Szegedy et al. "Going Deeper With Convolutions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[72] Tijmen Tieleman and Geoffrey Hinton. "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude". In: *COURSERA: Neural networks for machine learning* 4.2 (2012), pp. 26–31.

[73] Jonathan Tompson et al. "Efficient Object Localization Using Convolutional Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[74] Srinivas C Turaga et al. "Convolutional networks can learn to generate affinity graphs for image segmentation". In: *Neural computation* 22.2 (2010), pp. 511–538.

[75] Amos Tversky. "Features of similarity." In: *Psychological review* 84.4 (1977), p. 327.

[76] SESAR Joint Undertaking. *European ATM Master Plan Executive view*. Luxembourg: Publication Office of the European Union, 2020.

[77] SESAR Joint Undertaking and EUROCONTROL. *The Fly AI Report*. 2020. URL: `https://www.eurocontrol.int/sites/default/files/2020-03/eurocontrol-fly-ai-report-032020.pdf`.

[78] Régis Vaillant, Christophe Monrocq, and Yann Le Cun. “Original approach for the localisation of objects in images”. In: *IEE Proceedings-Vision, Image and Signal Processing* 141.4 (1994), pp. 245–250.

[79] Leslie G Valiant. “What must a global theory of cortex explain?” In: *Current opinion in neurobiology* 25 (2014), pp. 15–19.

[80] Stijn BJ Van Dam et al. “Functional presentation of travel opportunities in flexible use airspace: An EID of an airborne conflict support tool”. In: *2004 IEEE International Conference on Systems, Man and Cybernetics (IEEE Cat. No. 04CH37583)*. Vol. 1. IEEE. 2004, pp. 802–808.

[81] Kim J Vicente and Jens Rasmussen. “The ecology of human-machine systems II: Mediating direct perception in complex work domains”. In: *Ecological psychology* 2.3 (1990), pp. 207–249.

[82] Mark Visser et al. “Towards an Ecological Design of a 4-Dimensional Separation Assistance Interface”. Undefined/Unknown. In: *Proceedings of the 3rd International Conference on Human Centered Processes (HCP-2008), Workshop ” Supervisory Control in Critical Systems Management”, Delft, The Netherlands*. Ed. by Coppin Gilles Brézillon Patrick and Lenca Philipe. null ; Conference date: 08-06-2008 Through 12-06-2008. Telecom Bretagne, 2008, pp. 1–6. ISBN: 978-2-908849-22-6.

[83] A. Waibel et al. “Phoneme recognition using time-delay neural networks”. In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 37.3 (1989), pp. 328–339. DOI: 10.1109/29.21701.

[84] Li Weigang et al. “Decision support system in tactical air traffic flow management for air traffic flow controllers”. In: *Journal of Air Transport Management* 14.6 (2008), pp. 329–336. ISSN: 0969-6997. DOI: https://doi.org/10.1016/j.jairtraman.2008.08.007. URL: http://www.sciencedirect.com/science/article/pii/S0969699708001051.

[85] Carl Westin, Clark Borst, and Brian Hilburn. “Strategic Conformance: Overcoming Acceptance Issues of Decision Aiding Automation?” In: *IEEE Transactions on Human-Machine Systems* 46 (Nov. 2015), pp. 41–52. DOI: 10.1109/THMS.2015.2482480.

[86] Christopher D Wickens et al. *Introduction to Human Factors Engineering: Pearson New International Edition*. Pearson Higher Ed, 2013.

[87] Hsiang-Fu Yu et al. “Feature engineering and classifier ensemble for KDD cup 2010”. In: *KDD cup* 11 (2010).

[88] Bolei Zhou et al. “Learning deep features for discriminative localization”. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2921–2929.