Toward individual-sensitive automation for air traffic control using convolutional neural networks

van Rooijen, S. J.; Ellerbroek, J.; Borst, C.; van Kampen, E.

DOI
10.2514/1.D0180

Publication date
2020

Document Version
Accepted author manuscript

Published in
Journal of Air Transportation

Citation (APA)
van Rooijen, S. J., Ellerbroek, J., Borst, C., & van Kampen, E. (2020). Toward individual-sensitive automation for air traffic control using convolutional neural networks. *Journal of Air Transportation*, *28*(3), 105-113. https://doi.org/10.2514/1.D0180

Toward Individual-Sensitive Automation for Air Traffic Control Using Convolutional Neural Networks

Article · May 2020

DOI: 10.2514/1.D0180

4 authors, including:

Joost Ellerbroek
Delft University of Technology
104 PUBLICATIONS **667** CITATIONS

SEE PROFILE

Clark Borst
Delft University of Technology
91 PUBLICATIONS **531** CITATIONS

SEE PROFILE

Erik-Jan Van Kampen
Delft University of Technology
180 PUBLICATIONS **1,236** CITATIONS

SEE PROFILE

Towards Individual-Sensitive Automation for Air Traffic Control using Convolutional Neural Networks

S.J. van Rooijen *, J. Ellerbroek †, C. Borst ‡, E. van Kampen §
Delft University of Technology, Delft, The Netherlands, 2629HS

Lack of trust and acceptance caused by strategic mismatches in problem-solving have been identified as obstacles in the introduction of workload-alleviating automation in air traffic control. One possible way to overcome these obstacles is by creating automation capable of providing personalized advisories conformal to the individual controller. This paper focuses on performing an exploratory investigation into the tools and methodology required for creating conformal automation. Central in the creation of individualized prediction models is the combination of a visual feature, capturing traffic situations, and a tailored convolutional neural network model trained on individual controller data recorded from a human-in-the-loop simulation. The main advantage of using a visual feature is that it could facilitate 'transparency' of the machine learning model. Results show that the trained models can reasonably predict command type, direction and magnitude. Furthermore, a correlation is found between controller consistency and achieved prediction performance. A comparison between individual-sensitive and general models showed a benefit of individually trained models, confirming the strategy heterogeneity of the population, which is a critical assumption for personalized automation. Future research should be done in refining the model architecture, finding richer visual features that capture the breadth of human decision-making behavior and feedback model outputs back to individuals for measuring controller agreement.

I. Introduction

The introduction of automated decision support tools for Air Traffic Control (ATC) is widely seen as unavoidable to keep up with air traffic growth [1, 2]. However, in the past, the introduction of support tools for separation management has shown that such tools are frequently left unused [3–5]. In an extensive review on past efforts in automating parts of the ATC work, Westin *et al.*[6] argue that low automation conformance (i.e., a mismatch between human and computer problem-solving strategies and solutions) can be one of the reasons for low acceptance of automated advisories. For instance in the separation task, often multiple resolution options are available for a given conflict, and resolutions proposed by automation do not always coincide with human strategies and best practices[7].

Recent work has argued that making advisories *strategically conformal* can be an effective way to increase trust and acceptance of automation [6, 8]. If a decision support tool has the ability to adapt to individual strategies, acceptance should increase. In a human-in-the-loop simulation study, Hilburn *et al.*[8] demonstrated that conformal advisories were accepted more often than non-conformal advisories. In that study, however, conformal automation was simulated by replays of an individual's actions instead of real automation.

A promising method to create truly conformal automation that is able to predict human strategies is machine learning [9]. The advantage of having a decision support tool for ATC based on machine learning is that it would be able to adapt to different controller preferences without full knowledge of the underlying decision-making dynamics. However, prerequisites for individual-sensitive automation is that controllers should be sufficiently *consistent* and a group of controllers should be adequately *heterogeneous* in strategies and decisions for individual-sensitive automation to be beneficial.

A first proof-of-concept has been explored by Regtuit *et al.*[10], who showed that machine learning techniques are able to identify and replicate conflict resolution strategies from simple, synthetic traffic situations. While the results from this study were promising, its method is not easily extended to capture more realistic, multi-aircraft situations. The

*MSc-student, Control and Simulation group, Delft University of Technology
†Assistant professor, Control and Simulation group, Delft University of Technology
‡Assistant professor, Control and Simulation group, Delft University of Technology
§Assistant professor, Control and Simulation group, Delft University of Technology. AIAA Member.

work presented in this paper extends on that previous effort by proposing and testing a different machine learning model able to predict Air Traffic Controller (ATCo) commands in more realistic traffic situations.

To capture relevant aspects of realistic traffic situations in a concise form, the Solution Space Diagram (SSD)[11], also known as velocity obstacle diagram [12], will be used as *visual* input to the learning algorithm. The SSD is a visual representation of a velocity space of conflicting and conflict-free velocities and headings. Essentially, the SSD connects conflict situations with all possible actions to solve those situations, all in one diagram, thereby making it an ideal candidate as a learning feature. Numerous previous studies that have used the SSD as a decision-support tool have shown that it contains sufficient information concerning an air traffic situation to make an informed decision regarding conflict resolution [11, 13–16]. Learning from an image can be advantageous as it serves two purposes. First, it eliminates the need for manually designing features based on heuristics and assumptions. Second, a visual feature can make the machine learning algorithm more 'transparent' when presenting it to a human operator. This could facilitate understanding and monitoring what the machine is learning, thereby potentially addressing one of the challenges associated with artificial intelligence[17, 18].

The proposed model will use *convolutional neural networks*, a machine learning technique that is especially well-suited for visual learning [19], and uses the SSD and controller actions as inputs in learning to predict individual decisions. To provide training data, and to test controller consistency and the efficacy of using the SSD as a machine learning feature, a human-in-the-loop experiment is carried out featuring participants with various degrees in ATC expertise and experience. The recorded datasets are used to train individual-sensitive predictive models using a supervised learning algorithm. Individual model performance is compared to controller consistency and an inter-participant comparison and a comparison with general group-based models are performed to test the effect of individual-sensitive automation. Note that this study only focused on modeling and predicting individual controller decisions, and therefore did not analyze the model outputs with participants' acceptance.

The remainder of this paper is structured as follows: The proposed concept is introduced in section II. Experiment design is discussed in section III, and results are provided in section IV, followed by a discussion in section V and conclusions & recommendations in section VI.

II. Concept

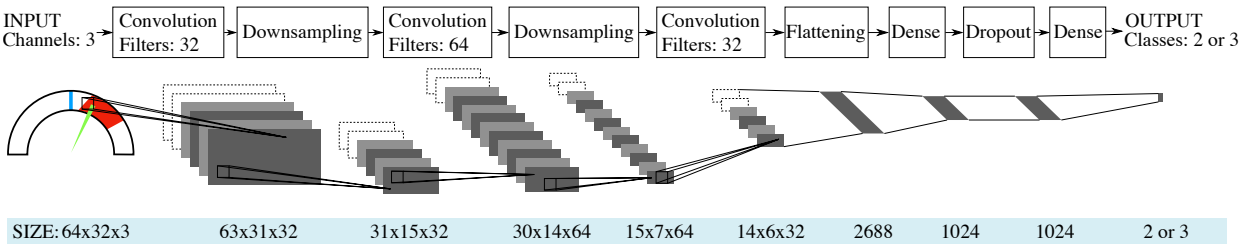


Fig. 1 The Convolutional Neural Network structure. All weights in one plane are identical. Filter size denoted in pixels. Adapted from LeCun (1998)[19].

Achieving conformal automation has similarities to a machine learning field called Imitation Learning, where machine learning is used to learn to perform tasks based on expert demonstrations. The aim is to generalize these observations to unseen situations, usually using supervised learning [20]. Artificial neural networks with a deep architecture are a powerful method for supervised learning as they automatically devise learning features, without reliance on pre-engineered parameters [21]. This research follows a similar approach, where the input images are provided in the form of rasterized SSDs. In particular, Convolutional Neural Networks (CNNs)[22] have shown their potential of training on image data, for example in learning to play computer games [23], self-driving cars [24, 25] and competing with world-class board game champions [26]. The most important reason to choose CNNs over other types of deep NN's is that there is a visual representation available (i.e., the SSD) that captures/summarizes conflict situations in a single image and reveals opportunities to solve those situations (in heading and/or speed). CNNs have proven to provide high accuracy in image classification tasks. They were originally inspired by human visual perception [22] and use fewer parameters compared to a fully connected network by re-using the same parameter numerous times.

A. Convolutional Neural Networks

Similar to a regular neural network, a CNN consists of multiple stacked neurons. In the case of a CNN, every input is connected to a pixel-value of the original image, as shown in Figure 1. On each layer of the network, individual neurons are connected to the weighted combination of the outputs of the previous layer, which is passed through a convolution filter. These filters slide over the entire image to create a new output map. By using different filters, specific visual features can be extracted from the input image, such as small corners or edges. In this study, filter size is always 2×2 pixels, and filters progress over the image with a stride of 1 pixel. The visual features are subsequently combined to form compositions while progressing through the net, using successive layers of filters. To avoid overfitting and reduce computational cost, the data is also downsampled in between filter steps [27]. In the final steps, the data is flattened, and reduced in steps to the expected number of output classes (e.g., when a conflict resolution maneuver can be either using heading, speed, or a direct-to, output classes are *heading*, *speed*, and *direct-to*, resulting in an output size of three). To reduce overfitting, a dropout layer is added here that sets the weights of a fraction of the neurons to zero at each epoch during training [28].

The neurons in each layer are specified by an activation function. In this study, all layers but the final layer use a Rectified Linear Unit (ReLU) activation function, because of its computational efficiency [29]. The final layer uses a softmax function [30], which transforms the layer of real values into a vector of probabilities per output class $\sigma(\mathbf{z})$. The softmax function is defined by Equation 1, where all entries of σ are real, within $[0, 1]$, and add up to 1. K is the dimension of input vector $\mathbf{z}$.

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \text{ for } j = 1, ..., K \tag{1}$$

A cross-entropy function is used to take these probabilities into account in the loss function [31]. The cross entropy H is calculated for M output classes by comparing the probability vector σ resulting from the softmax function to the one-hot encoded target vector y_i:

$$H(y, \sigma) = -\sum_{i}^{M} y_i \log \sigma_i \tag{2}$$

The calculated losses, averaged over a minibatch of samples are used to update the network parameters θ. In this study, this update is done using a first-order gradient-based optimization algorithm called Adam. Empirical results show that it outperforms previously popular optimizers such as AdaGrad, RMSProp and SGDNesterov [32].

B. Solution Space Diagram

The performance of CNNs is naturally determined by their inputs, as they should capture all relevant information for the model to make a prediction. This research utilizes the Solution Space Diagram (SSD) as input to the neural network. The SSD was originally designed as a decision-support tool that integrates various critical parameters of the Conflict Detection and Resolution problem [33–35]. This was later extended to complexity and workload analysis [11, 36–38], automated conflict resolution [15, 16, 39] and even ATC training[40]. Based on the findings of these studies, this research hypothesizes that an SSD image as learning feature contains sufficient information concerning air traffic conflicts to make an informed decision. Compared to using raw traffic images (e.g., locations and velocities of blips on a radar display), the SSD seems a more informed approach as it captures the essence of conflict situations (i.e., conflict angle, proximity, passing in front / behind, absolute and relative speeds, protected zone dimension, etc.) and describes the conflict situation directly in an action state-space used by controllers (i.e, heading and/or speed opportunities). Thus, the SSD connects conflict situations with all possible actions to solve those situations, all in one visual diagram.

Figure 2 illustrates how the SSD is constructed. It consists of an area of reachable velocities, bounded by concentric limits of minimum and maximum operating speeds. This reachable space is reduced by triangular velocity obstacles that correspond to the set of velocity vectors that would lead to a loss of separation with a nearby aircraft. The color of the velocity obstacles is determined by the time to closest point of approach (CPA), divided into three ranges: red ($t_{\text{cpa}} < 60s$), orange ($60 < t_{\text{cpa}} < 120s$) and gray ($t_{\text{cpa}} > 120s$). The remaining free space corresponds to the reachable conflict-free speed/heading combinations. In the SSD, the current heading and speed are indicated by the green vector, see Figure 2. As an additional feature, the target heading to the exit waypoint of the respective aircraft is shown with a

blue line. This information is part of the normal tasks of an air traffic controller, and may influence conflict resolution decisions.

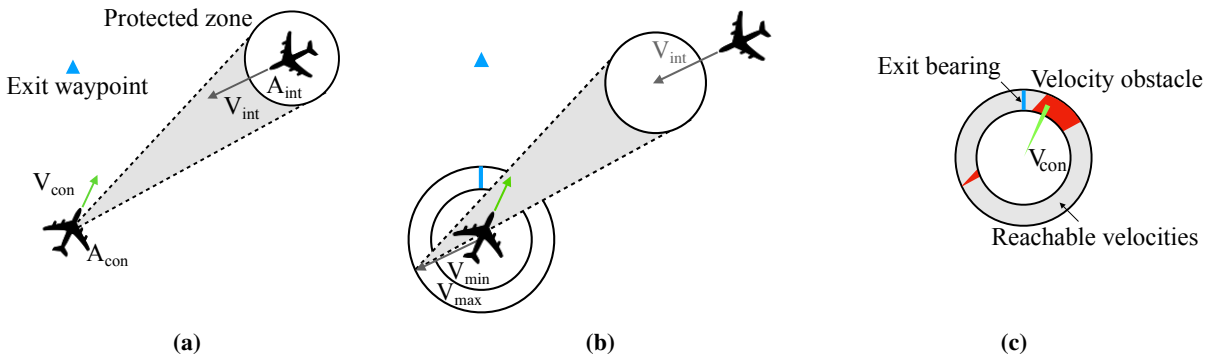


Fig. 2 Construction of the SSD: (a) conflicting relative velocities, (b) velocity vector displacement and (c) limiting the solution area. Adapted from Mercado et al. (2010) [11].

In the current study, the above method is used to learn individual controller strategies for observed traffic conflicts. Following Westin's approach [41], this study defines strategy in terms of three control variables: resolution *type* (heading, speed, or direct-to), resolution *direction* (left or right, speed increase or decrease), and resolution magnitude ($[0, 10]$ deg, $[10, 45]$ deg and > 45 for heading resolutions, and $[200 - 250]$ kts, and $[250 - 290]$ kts for speed resolutions) With the SSDs of all detected conflicts as input, three independent CNNs (as described in Figure 1) are trained to match the SSD of the controlled aircraft to the observed controller strategy: one CNN for resolution type, one for resolution direction, and one for resolution magnitude.

III. Experiment

An experiment was performed to provide training and test data for the model described in the previous section. In this experiment, participants were asked to control a sector with multiple incoming aircraft, where they guided the aircraft to their exit waypoint as efficiently as possible, while avoiding losses of separation. The shape of the sector – inspired by Amsterdam Sector South 1 – is shown in Figure 3, with the main traffic flows north- and west-bound. The circles surrounding the aircraft indicate the protected zones (D = 5nm). Maneuvering was restricted to the horizontal plane, i.e., all aircraft flew at the same flight level and only heading, speed, and direct-to commands were allowed by means of a command interface (Figure 4). Furthermore, no wind was present in the simulation and communication with aircraft was simulated with a digital data link, requiring no radio telephony (R/T). These simplifications decreased the degrees of freedom and potential confounding factors, which enabled better comparison between controllers in terms of consistency and strategy.

A. Conditions

Two traffic scenarios (S1 and S2) were used in the experiment with identical sector geometry, but different traffic flows. The scenarios have been setup in such a way that they represent traffic entering and leaving an airspace sector along major traffic routes / streams, similar to real-life situations. Randomness in the scenarios was not desirable for this study, given that measuring participants' consistency in solving similar conflicts situations was required. Each scenario consisted of 10 conflict pairs. The main aircraft flow was always directed towards the north, and was crossed by traffic on several headings from the east. All conflicts were crossing conflicts, with conflict angles between 45 and 135 degrees. Every conflict angle in the set $\{45, 55, \ldots, 135\}$ was visited at least four times during the entire experiment. The conflicts were chronologically spaced in a way to minimize interference between conflicts.

Both scenarios were performed two times, resulting in a total of four 20-minute scenarios, generating 80 minutes of data per participant. The order of the conditions was balanced between participants to avoid learning biases.

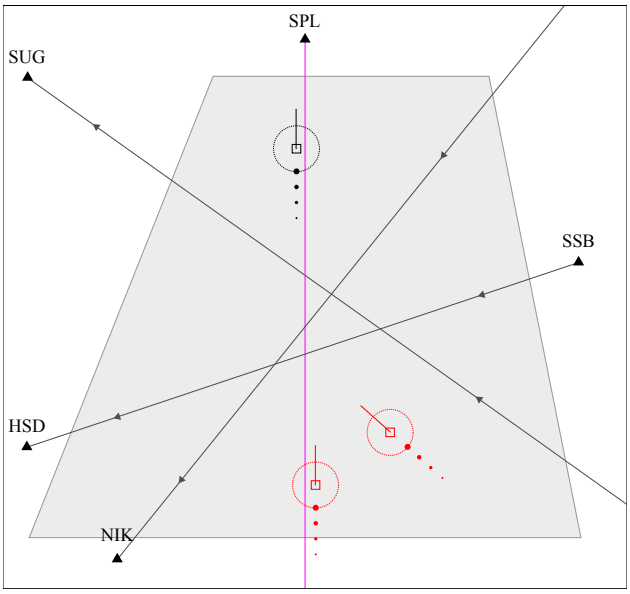


Fig. 3 The 50nm × 60nm sector as displayed in ATC simulator SectorX. Three aircraft are visible of which two are in conflict.

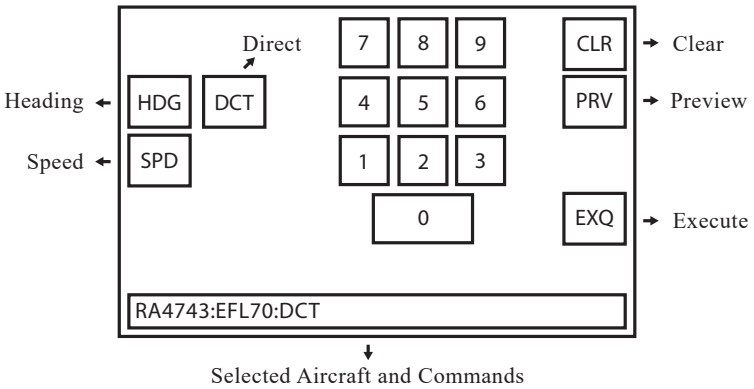


Fig. 4 The command interface that participants used in controlling the aircraft.

B. Participants

The population consisted of 12 participants, all university staff and students, with varying experience levels in performing ATC control tasks. These experiences ranged from participation in previous ATC-related experiments toward a five-day Area Control course at the Royal Netherlands Aerospace Laboratory (NLR). Given the simplified simulation environment, participation of professional controllers was not deemed critical in this exploratory study where the goal is to create prediction models tuned to the individual participant.

C. Procedure

The experiment was performed on a personal computer with a 30" screen showing the sector under control, and a touch control device window similar to the touch control devices used at the Dutch air navigation service provider LVNL. Participants could control aircraft through this interface using a computer mouse. Before measurements, each participant performed three training runs of increasing difficulty to get acquainted with the ATC simulator. Training runs lasted 90s, but could be prolonged when required.

Each time a command was given by a participant, the application saved the command plus an image of the controlled aircraft's SSD. Commands were defined in terms of *type* (heading, speed, or direct-to), *direction* (left or right, speed increase or decrease) and *magnitude* (magnitude of the heading or speed change). During supervised model training, the SSD image functions as input while the given command functions as target.

D. Data preprocessing

The SSD dataset consists of 128x128 pixel RGB samples which were preprocessed for computing time improvements. First, the SSD images were rotated track-up, so that the velocity vector of the controlled aircraft always points upward. Second, because the most relevant information in the SSD is located in the upper half of the image (aircraft are not likely to make turns larger than 90 deg), the lower half of the SSD was cropped away to decrease training times.

E. Dependent measures

To avoid misleading results from an imbalanced dataset, model performance is evaluated using the *Matthews Correlation Coefficient (MCC)* [42] for training and validation. For example, if a controller chooses to use heading instead of speed commands in 95% of the situations, simply always guessing 'heading' would result in a 0.95 accuracy score. This would give a false sense of model performance, because 0% of the speed commands are predicted. In such cases, the MCC metric gives less biased results than accuracy [43]. The definitions of accuracy and the two-class MCC are shown in Equations (3) and (4) respectively, where TP = true positive, TN = true negative, FP = false positive, FN = false negative.

$$\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{3}$$

with possible values [0, 1].

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \tag{4}$$

with possible values [-1, 1]. As MCC is a more critical metric, MCC values are often lower compared to accuracy values for identical models.

In this study, consistency is considered as the relative occurrence of the most often-used command. For resolution *type* and resolution *direction* this is determined by summing over all commands per participant:

$$\text{consistency}_{\text{type, direction}} = \max\left(\frac{\sum \text{class 1}}{\sum \text{class 1} + \ldots + \text{class n}}, \ldots, \frac{\sum \text{class n}}{\sum \text{class 1} + \ldots + \text{class n}} \right) \tag{5}$$

where for example class 1, 2, and 3 are respectively HDG, SPD, and DIRECT-TO for command *type* consistency. Using this definition, consistency equals 1 when a single command type is used and 0.5 when they are evenly balanced. For example, when a participant steers the aircraft to the right 90 out of 100 times and left 10 out of 100 times, the consistency is calculated to be $0.9 = \max(\frac{90}{100}, \frac{10}{100})$.

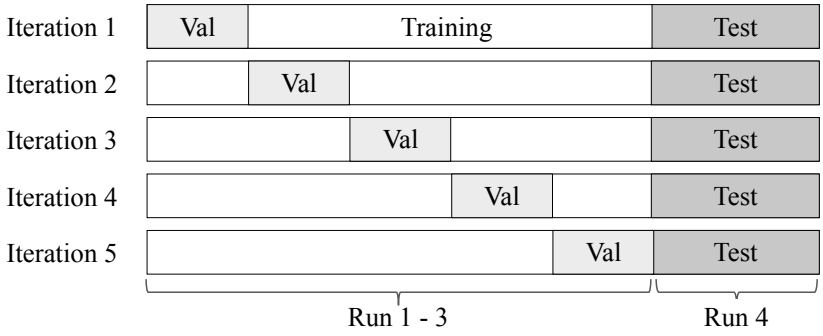


Fig. 5 Performance is validated using stratified 5-fold cross-validation with a separate test set. One bar represents all available data per participant.

For resolution *magnitude*, a different equation is used:

$$\text{consistency}_{\text{magnitude}} = \frac{\sum \text{unique values possible}}{\sum \text{unique values used}} \tag{6}$$

where the consistency of a participant decreases when a broader range of magnitude values is used.

F. Model training

The data from the experiment is used in two ways: First, the dataset from each participant is used to train individual models. Then, five general models are trained on random samples drawn from the combined participant data.

The individual models are trained following the procedure illustrated in Figure 6. The dataset consists of input (SSD images) and target (commands) data. Model performance is based on prediction accuracy. After preprocessing, the test data is separated into a training set (experiment runs 1-3) and a test set (experiment run 4). The training data is used to train three personalized models per participant, one for each resolution dimension (type, direction, and magnitude), see Table 1 for a summary of the selected training procedure settings. During training, K-fold validation is used to select a model that obtains the highest prediction performance. With K-fold validation, $1/k^{th}$ of the data is iteratively reserved for validation, as illustrated in Figure 5. In this case 5 folds are used, which means that the data is shuffled into 5 different training–validation sets. Using this validation data, the best model is then selected as a final model. This combination of model training and validation has proven to have lower variance and bias of performance measures compared to other methods [44]. In the current experiment, five folds are applied during training.

The five general models are trained on random samples of all (between-subject) data, in the same way that the individual models are trained. An equal number of random samples is used as is available in the individual model training. The mean performance of these five models will be taken, and will be compared to the individual models as a (non-individually sensitive) baseline.

G. Model testing

After training, each individual model is tested with the participant's corresponding test dataset (i.e. the 4^{th} run), to test the individual prediction accuracy of the trained models, as shown in Figure 7a. Additionally, each individual model is tested with all *other* participants' test datasets, to evaluate how individually-sensitive each model is (Figure 7b). The general models are also tested with all participants' test datasets (Figure 7c). In all three above-mentioned cases, the test data is data unseen up to this point.

IV. Results

Using the data from twelve participants in the human-in-the-loop experiment, twelve individual models and five general models were trained. This section presents the individual model results, individual model performance as a function of participant consistency, an inter-participant test of model performance, and a comparison of individual models to the average general model performance. Performance is measured using the *MCC* (see section III.E), which ranges between -1 and 1. Because negative correlation did not occur, all MCC result figures are clipped to a range

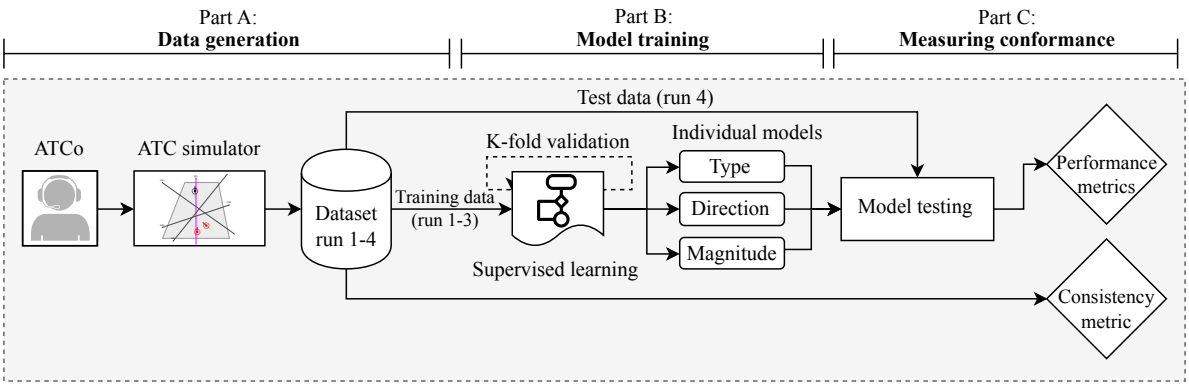
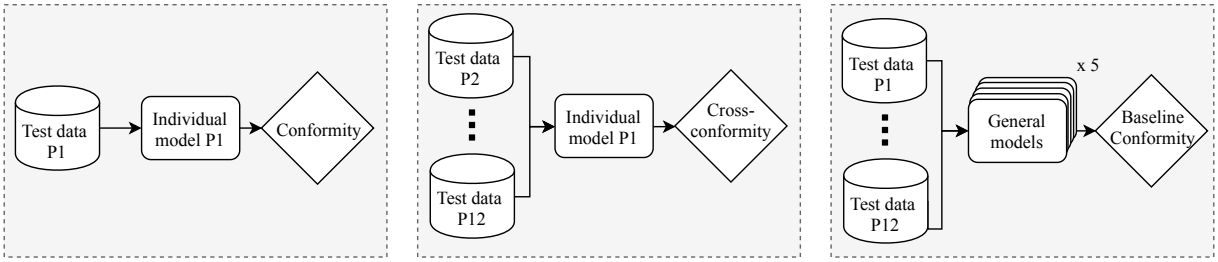


Fig. 6 Data generation and training & testing of the individual models for one participant. The models predict a command for a given SSD image.



(a) Regular **(b) Inter-participant** **(c) Baseline**

Fig. 7 Three validation steps for participant 1 (P1).

of $[0, 1]$. To test for robustness of the results, the models for Participant 1 were trained 50 times (i.e., 50 runs). The validation performance has a standard deviation of 0.018, 0.014 and 0.017 for direction, type and value respectively, indicating that the training method is indeed robust.

A. Training convergence

In the training phase, data from the first three experiment runs is used to train several candidate models. Using the K-fold method illustrated in Figure 5, five candidate models are trained, of which the performance is validated using five different subsets of the data. Figure 8 shows the training progress in terms of these validation results for the individual model of Participant 1, with training epoch on the x-axis, and the resulting MCC score on the y-axis. Here, the spread around each line depicts the range between the least and best performing folds per control variable during training, which lasts 25 epochs. It can be seen that with successive epochs, MCC values increase, which indicates that the neural network successfully 'learns' from the data samples. In most cases, the models reach MCC scores > 0.95 during *training*, a performance level that is not achieved in the validation steps, as can be seen in Figure 8. This difference between training and validation performance indicates overfitting on the training data. The spread shows that validation MCC can differ more than 0.2 per fold, which is a relatively large amount compared to the mean value.

B. Model performance on individual test data

After training (Figure 8), the individual models are applied to the test datasets of each participant (Run 4). Figure 9 shows the achieved MCC scores per control variable. In this figure, the large variability in performance (particularly for the *type* control variable) indicates that the personalized predictions are not equally effective across the entire population of participants. The *direction* prediction shows the highest MCC score (mean = 0.76, SD = 0.11), while *type* (mean = 0.52, SD = 0.21) and *magnitude* (mean = 0.64, SD = 0.12) predictions achieve lower performances.

A potential reason for poor performance of the trained model is low participant consistency: in some cases, the participant data on which the model is trained does not show sufficiently consistent behaviour across different runs,

Table 1 (Hyper)parameters used during training of the convolutional neural networks

Parameters	Value
Optimization algorithm	Adam
Output activation	Softmax classifier
Loss function	Categorical entropy
Train/val/test ratio	60%/15%/25%
K-folds	5
Mini batch-size	32 samples
Steps-per-epoch	2 × training samples / batch-size
Epochs	30
Learning rate	0.01
Dropout rate	20%
Input image dimensions	128x128 px

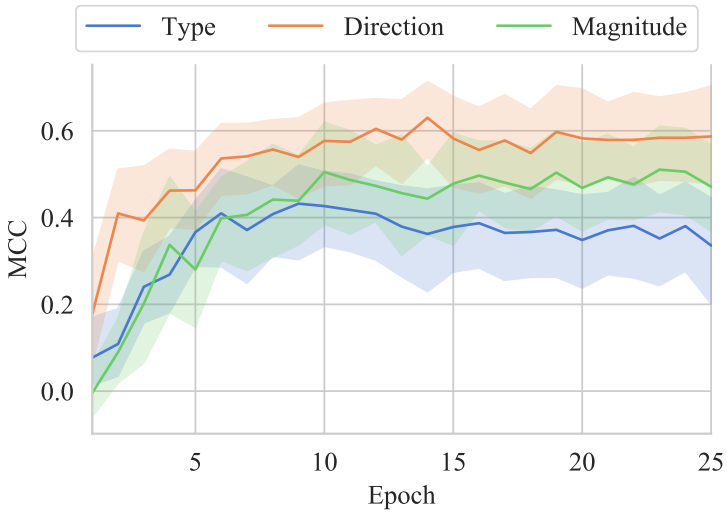


Fig. 8 Validation performance, expressed in the Matthews Correlation Coefficient (MCC), during training of P1's individual model. The spread indicates the maximum and minimum performance for each fold per control variable.

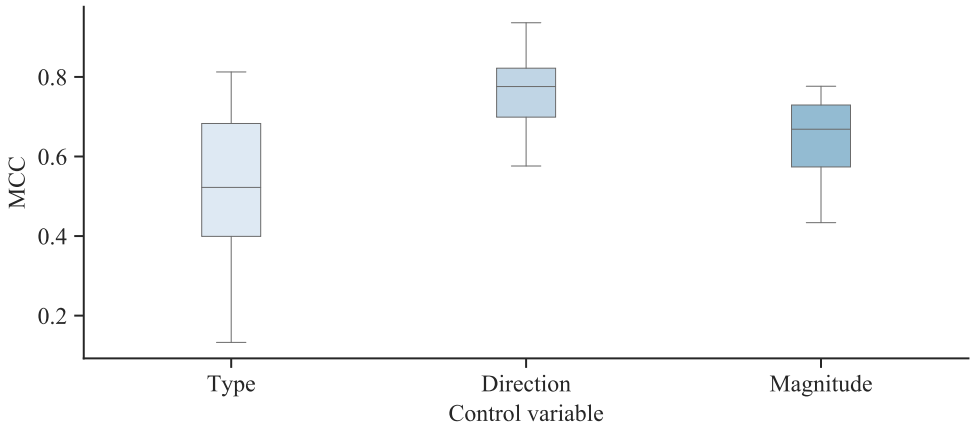


Fig. 9 Model test-performance for all participants per control variable, expressed in the Matthews Correlation Coefficient (MCC). The bars contain the second and third quartile of participants by performance

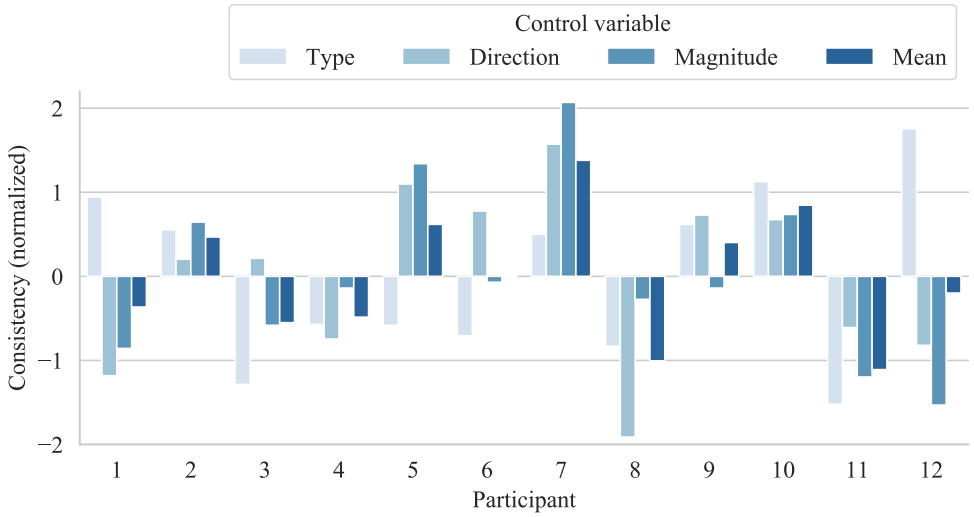


Fig. 10 Consistency scores per participant split per control variable.

and between conflicts that do appear comparable in the SSD. Figure 10 shows the normalized consistency (as defined in Section III.E) per participant and control variable. Here, it can be seen that while some participants are relatively consistent (participants 5, 7 and 10), other participants (particularly 8 and 11) show more erratic decision-making. Figure 10 also shows that participant consistency varies per control variable. For instance, participants can be very consistent in the *type* of resolution they choose, but are less consistent in the *direction* they choose for their resolutions.

The effect of participant consistency on the performance of the trained model can be evaluated by observing the correlation between consistency and model performance. To illustrate this, Figure 11 shows the mean model performance (the mean over all folds and abstraction levels), against the mean consistency per participant. When a Pearson Correlation Coefficient test is applied to this data, a positive correlation ($r = 0.75$, $p = .005$) can be found between participant consistency and individual model MCC. This supports the assumption that the personal models of more consistent participants perform better than the models of their less consistent counterparts.

C. Model performance on inter-participant data

A way to evaluate whether the personalized models are indeed individual-sensitive, is to test the models against all other participant test datasets. Figure 12 shows the results of using the models of each participant on the test data of all

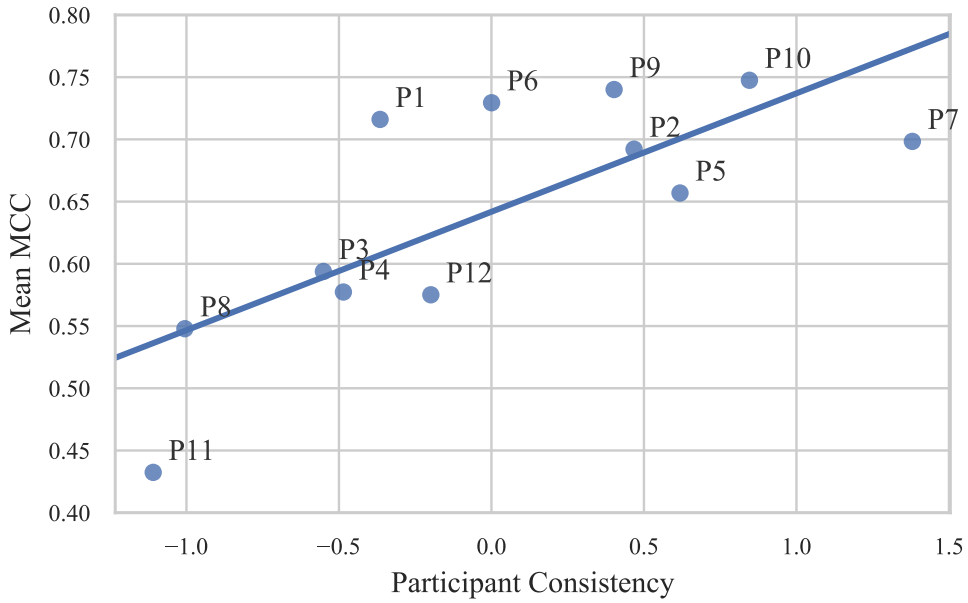


Fig. 11 Participant consistency vs individual model performance. $R^2 = 0.56$**.**

participants. In these spider-plots, the model performance (MCC value) in terms of *type* (blue), *direction* (orange), and *magnitude* (green) is shown for each participant's test data, along twelve radials of each chart. For the individual models of participants 1, 6, 7, 9, and 10 it can be seen that overall performance is highest when the model is applied to the test data of the corresponding participant. For instance, for the individual model of participant 1, a mean performance of MCC = 0.72 is achieved when the model is applied on the test data of participant 1, compared to an average MCC of 0.37 when testing with other participants' data. This difference indicates that participant 1 makes different decisions in similar situations compared to the rest of the population. Other participants' models show more uniform MCC scores, regardless of which test set is use.

D. Comparison between individual and general models

A second way to test whether the trained models are individual-sensitive is to compare individual model performance to the performance of the general models, when applied to the test data of each respective participant. Figure 13 shows the average individual model performance per participant, compared to the average general model performance per participant. The chart shows that most individual models outperform the mean of the general models, but some cases show near equal or even worse (P4 and P8) performance, possibly caused by a strategy change in the final run.

A paired t-test shows that the individual models perform significantly better ($t(11) = 2.9$, $p = 0.02$) than the general models in terms of MCC, see Figure 14. The individual models provide a mean 0.08 (SD = 0.10) MCC improvement over the general models. The personalized approach is most effective for participant 1, whose individual models score 0.20 MCC higher than the baseline.

V. Discussion

The aim of this study was to create individual-sensitive models of controller actions by training a set of convolutional neural networks on a visual representation of traffic conflicts. The main advantage of using a visual learning feature is that it could eventually be used to make machine learning more 'transparent' and explainable. A human-in-the-loop experiment was performed to generate training data for the model creation. The results are promising in predicting individual controller actions compared to a general "one-size-fits-all" model, especially when controllers are more consistent. For some participants, however, the individual model did not outperform the general model in terms of prediction performance, requiring a discussion on the possible areas for improvement in terms of the proposed methodology.

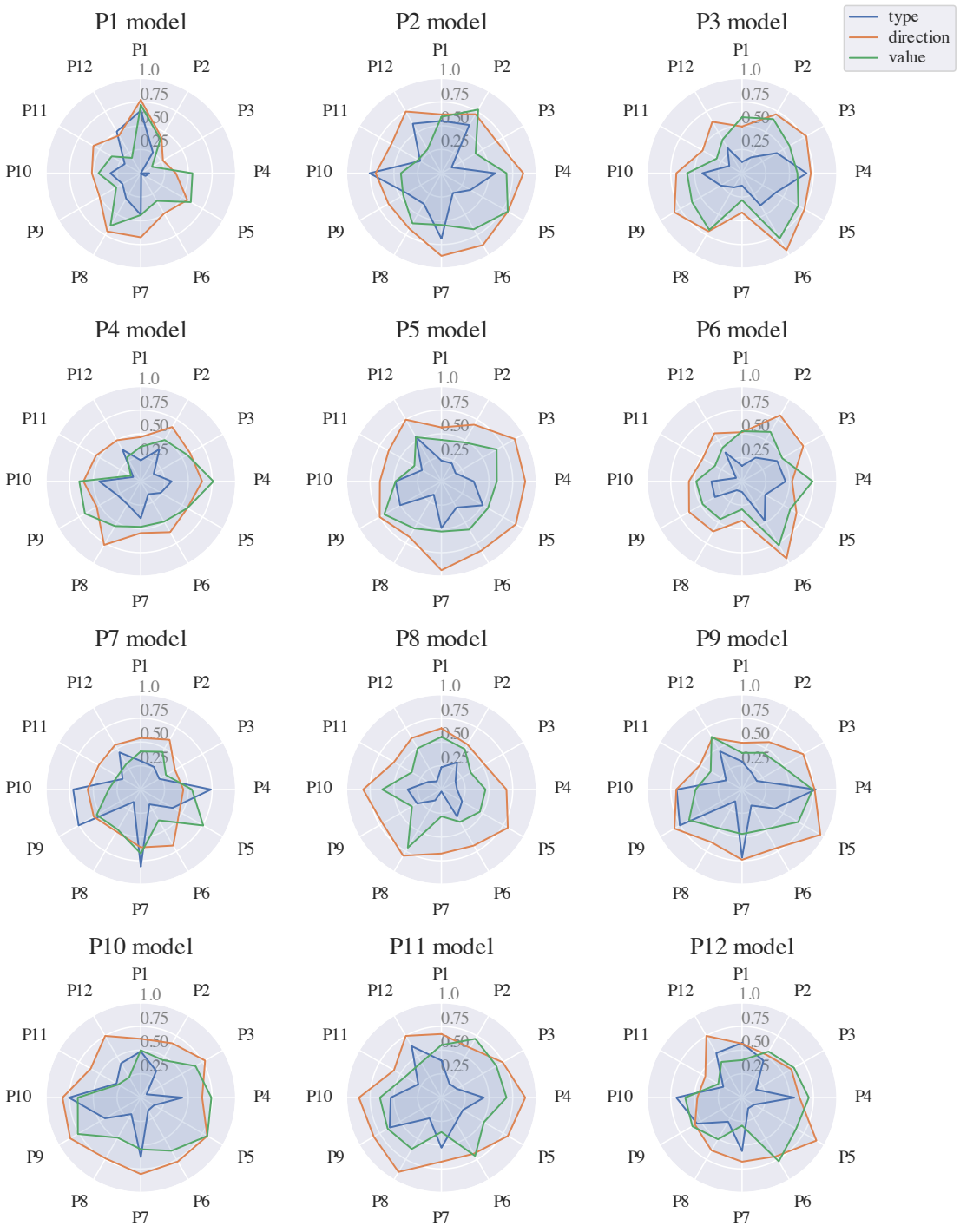


Fig. 12 Performance (in MCC) of individual models tested on the test datasets of all other participants.

One source of diminished prediction performance could be attributed to the visual model, the SSD, that was used to train the models. In its currently used form, the SSD is a state-based representation of the instantaneous traffic state at the time a controller command was given. This corresponds to a controller who only looks one step ahead when making a decision to solve a conflict. In the experiment, it was observed that some participants were planning several steps ahead. As an example, a deliberate strategy was temporarily vectoring an aircraft inside the no-go velocity obstacle of a

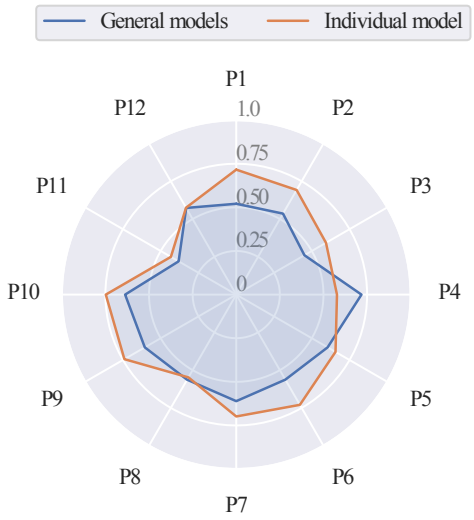


Fig. 13 Performance (in MCC) of each participant's individual model compared to the mean performance of five general models, averaged over all abstraction levels.

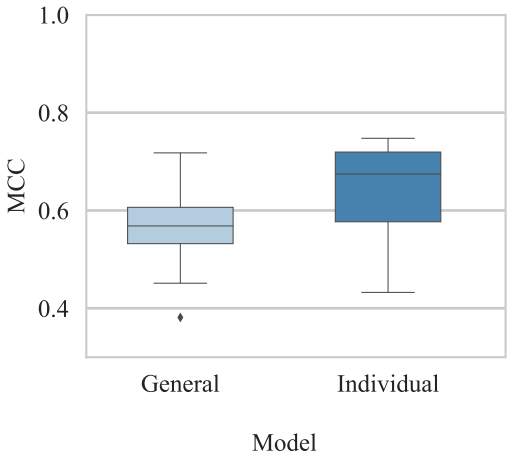


Fig. 14 Comparison of model performance between general and individual models.

far-away aircraft when solving a conflict with a near-by aircraft. Such intentional behavior is not supported by the SSD in its current form. It is therefore recommended to revise the SSD or find an alternative visual representation that can "look forward" and thus capture planning activities. It would also be interesting to look at the use of raw conflict images, which the human operator would use, as input to the CNN's, to see if the CNN's can extract any relevant and useful features from this data. Additionally, our current machine learning model also does not "look backwards" as it contains no memory of the commands given in the past. In dynamic control problems, actions are usually impacted by actions undertaken in the past. One way to solve this issue could be by combining the CNN with a Recurrent Neural Network (RNN) to account for temporal dynamic behavior.

Other sources of diminished prediction performance can be attributed to the quantity and quality of the data as well as the model architecture. It is a common problem in machine learning that model training requires a large amount of data. To mitigate this problem, the performed experiment considered only a subset of the types of conflict that controllers can encounter in their sector. Throughout the experiment, similar conflicts were presented to each participant multiple times, by only introducing conflicting traffic from the east, with a limited number of crossing angles. Increasing the breadth of encountered conflicts could increase the validity and applicability of the trained models, but would require more training data. Results show that model performance per validation fold can fluctuate up to 0.2 MCC, which shows sensitivity to random state-action pair sampling from the training set. This indicates that even for this limited subset of conflicts, performance improvements can be gained with exposure to more training data. Data quantity, however, is often a limiting factor in machine learning experiments with human data. Also in this case study, more samples could have been generated per participant in the experiment, but participant availability was limited. Part of the goal of this research was to show heterogeneity between participants and to demonstrate this, one initially does not need very large sample sizes. However, future research could aim to obtain higher accuracy by measuring one participant for a longer period of time to generate more training samples.

Evidence that our CNN model is able to interpret the SSD is shown by the accuracy increase during training. This might, however, also indicate that the network overfits on the training data at pixel-level without generalization to new samples. For the current dataset, the ability to generalize is demonstrated by the fact that the validation performance during training follows an upward trend similar to the training performance. Test results using the separate test set further confirm this. On the other hand, overfitting on the training samples does occur to a certain extent, indicated by the performance difference between the training and test sets.

It is expected that performance improvements can be achieved by a formal grid search to optimize the hyper-parameters of the model. However, since every training iteration generates 360 models (12 participants $\times$ 3 control variables $\times$ 2 repetitions $\times$ 5 cross-validation folds), iterating over parameters is computationally expensive and meticulous. The network architecture used in this research is kept constant for all control variables, i.e., it can predict command *type*, *direction* and *magnitude* by only altering the last fully connected layer. Designing separate network architectures that are tailored to each control variable could consequently improve performance. Resolution *magnitude* predictions are obtained using classification (i.e., dividing possible magnitudes over a limited number of bins) to more easily compare results to *type* and *direction* predictions. As resolution magnitude values are, in reality, more finely grained, this caused the accuracy of *magnitude* predictions to be reliant on the classification, i.e., the granularity of the bins. Therefore, more precise predictions are expected to be achieved using regression instead of classification.

VI. Conclusions

This research evaluated to what extent prediction models of actions undertaken by individual air traffic controllers can be achieved using convolutional neural networks. A 12-participant experiment was devised to generate training data consisting of solution space diagram (SSD) images, capturing both conflict situations and the action state-space, and conflict resolutions. Results show a correlation between the controller consistency metric and achieved model performance, confirming the hypothesis that consistent controllers would be more suited for strategic conformal automation support. Personalized models obtained significantly higher prediction accuracy than general models, indicating that controllers in this experiment exhibit differentiating strategies, i.e., are not homogeneous as a group. This is a critical assumption for strategic conformal automation. However, the performance improvement due to individual modeling substantially differed per participant, indicating the need to refine the proposed methodology by finding better ways to capture more aspects of human decision-making behavior and by exploring different machine learning architectures.

References

[1] SESAR Consortium, "The Concept of Operations at a glance," *Single European Sky*, 2007.

[2] Joint Planning and Development Office (JPDO), and Next Generation Air Transportation System (NextGen), "Concept of operations for the next generation air transportation system," Tech. rep., 2011.

[3] Bekier, M., Molesworth, B. R., and Williamson, A., "Tipping point: The narrow path between automation acceptance and rejection in air traffic management," *Safety science*, Vol. 50, No. 2, 2012, pp. 259–265. doi:https://doi.org/10.1016/j.ssci.2011.08.059.

[4] Bolic, T. S., "Automation adoption and adaptation in air traffic control," Ph.D. thesis, University of California, 2006.

[5] Ehrmanntraut, R., "Full automation of air traffic management in high complexity airspace," Ph.D. thesis, Technical University of Dresden, 2010.

[6] Westin, C. A. L., Borst, C., and Hilburn, B. G., "Strategic Conformance: Overcoming Acceptance Issues of Decision Aiding Automation?" *IEEE Transactions on Human-Machine Systems*, Vol. 46, No. 1, 2016, pp. 41–52. doi:10.1109/THMS.2015.2482480.

[7] Prevot, T., Homola, J. R., Martin, L. H., Mercer, J. S., and Cabrall, C. D., "Toward automated air traffic control: investigating a fundamental paradigm shift in human/systems interaction," *International Journal of Human-Computer Interaction*, Vol. 28, No. 2, 2012, pp. 77–98. doi:https://doi.org/10.1080/10447318.2012.634756.

[8] Hilburn, B. G., Westin, C. A. L., 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. doi:https://doi.org/10.2514/atcq.22.2.115.

[9] Sutton, R. S., Barto, A. G., and Williams, R. J., "Reinforcement Learning is Direct Adaptive Optimal Control," *IEEE Control Systems*, Vol. 12, No. 2, 1992, pp. 19–22. doi:https://doi.org/10.1109/37.126844.

[10] Regtuit, R. M., Borst, C., Van Kampen, E.-J., and van Paassen, M. M., "Building Strategic Conformal Automation for Air Traffic Control Using Machine Learning," *AIAA SciTech Forum*, AIAA Information Systems, Kissimmee, Florida, 2018. doi:https://doi.org/10.2514/6.2018-0074.

[11] Mercado-Velasco, G., Mulder, M., and Van Paassen, M. M., "Analysis of Air Traffic Controller Workload Reduction Based on the Solution Space for the Merging Task," *AIAA Guidance, Navigation, and Control Conference*, Vol. AIAA 2010-, No. August, 2010, pp. 1–18. doi:10.2514/6.2010-7541.

[12] Fiorini, P., and Shiller, Z., "Motion planning in dynamic environments using velocity obstacles," *The International Journal of Robotics Research*, Vol. 17, No. 7, 1998, pp. 760–772. doi:https://doi.org/10.1177/027836499801700706.

[13] Van Dam, S. B. J., Abeloos, A. L., Mulder, M., and Van Paassen, M. M., "Functional presentation of travel opportunities in flexible use airspace: An EID of an airborne conflict support tool," *Conference Proceedings - IEEE International Conference on Systems, Man and Cybernetics*, Vol. 1, No. January, 2004, pp. 802–808. doi:10.1109/ICSMC.2004.1398401.

[14] Ellerbroek, J., Brantegem, K., Van Paassen, M., de Gelder, N., and Mulder, M., "Experimental evaluation of a coplanar airborne separation display," *IEEE Transactions on Human-Machine Systems*, Vol. 43, No. 3, 2013, pp. 290–301. doi:https://doi.org/10.1109/ICSMC.2012.6377838.

[15] Jenie, Y. I., van Kampen, E.-J., de Visser, C. C., Ellerbroek, J., and Hoekstra, J. M., "Selective velocity obstacle method for deconflicting maneuvers applied to unmanned aerial vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 38, No. 6, 2015, pp. 1140–1146. doi:https://doi.org/10.2514/1.G000737.

[16] Borst, C., Bijsterbosch, V. A., van Paassen, M. M., and Mulder, M., "Ecological interface design: supporting fault diagnosis of automated advice in a supervisory air traffic control task," *Cognition, Technology & Work*, Vol. 19, No. 4, 2017, pp. 545–560. doi:10.1007/s10111-017-0438-y.

[17] European Union Aviation Safety Agency, "Artificial Intelligence Roadmap: A Human-Centric Approach to AI in Aviation," Tech. Rep. February, 2020. URL `https://www.easa.europa.eu/newsroom-and-events/news/easa-artificial-intelligence-roadmap-10-published`.

[18] Adadi, A., and Berrada, M., "Peeking Inside the Black-Box: A Survey on Explainable Artificial Intelligence (XAI)," *IEEE Access*, Vol. 6, 2018, pp. 52138–52160. doi:10.1109/ACCESS.2018.2870052.

[19] LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P., "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, Vol. 86, No. 11, 1998, pp. 2278–2324. doi:https://doi.org/10.1109/5.726791.

[20] Piot, B., Geist, M., and Pietquin, O., "Bridging the gap between imitation learning and inverse reinforcement learning," *IEEE transactions on neural networks and learning systems*, Vol. 28, No. 8, 2017, pp. 1814–1826. doi:https://doi.org/10.1109/TNNLS.2016.2543000.

[21] Sutton, R. S., and Barto, A. G., *Reinforcement Learning: An Introduction*, 2nd ed., MIT Press, Cambridge, MA, 2018.

[22] LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W., and Jackel, L. D., "Backpropagation Applied to Handwritten Zip Code Recognition," *Neural computation*, Vol. 1, No. 4, 1989, pp. 541–551. doi:https://doi.org/10.1162/neco.1989.1.4.541.

[23] Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D., "Human-level control through deep reinforcement learning," *Nature*, Vol. 518, 2015, p. 529. doi:10.1038/nature14236, URL `http://dx.doi.org/10.1038/nature14236`.

[24] Pomerleau, D. a., "Alvinn: An autonomous land vehicle in a neural network," *Advances in Neural Information Processing Systems 1*, 1989, pp. 305–313. doi:10.5555/89851.89891.

[25] Chen, Z., and Huang, X., "End-To-end learning for lane keeping of self-driving cars," *IEEE Intelligent Vehicles Symposium, Proceedings*, 2017, pp. 1856–1860. doi:10.1109/IVS.2017.7995975.

[26] Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D., "Mastering the game of Go with deep neural networks and tree search," *Nature*, Vol. 529, No. 7587, 2016, pp. 484–489. doi:10.1038/nature16961, URL `http://dx.doi.org/10.1038/nature16961`.

[27] LeCun, Y., Pfeifer, R., Schreter, Z., Fogelman, F., and Steels, L., "Generalization and network design strategies," Tech. rep., Elsevier, Zurich, Switserland, 1989.

[28] Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. R., "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," *Journal of Machine Learning Research*, Vol. 15, 2014, pp. 1929–1958. doi:10.1214/12-AOS1000.

[29] Jarrett, K., Kavukcuoglu, K., and LeCun, Y., "What is the best multi-stage architecture for object recognition?" *Computer Vision, 2009 IEEE 12th International Conference on*, IEEE, 2009, pp. 2146–2153. doi:https://doi.org/10.1109/ICCV.2009.5459469.

[30] Bishop, C. M., *Pattern Recognition and Machine Learning*, Springer, New York, USA, 2006.

[31] Goodfellow, I., Bengio, Y., and Courville, A., *Deep Learning*, MIT Press, 2016.

[32] Kingma, D. P., and Ba, J., "Adam: A method for stochastic optimization," *International Conference on Learning Representations (ICLR)*, 2015.

[33] Van Dam, S. B. J., Mulder, M., and van Paassen, M. M., "Ecological Interface Design of a Tactical Airborne Separation Assistance Tool," *IEEE Transactions on Systems, Man, and Cybernetics Part A:Systems and Humans*, Vol. 38, No. 6, 2008, pp. 1221–1233. doi:10.1109/TSMCA.2008.2001069.

[34] Ellerbroek, J., Visser, M., van Dam, S. B. J., Mulder, M., and van Paassen, M. M., "Design of an Airborne Three-Dimensional Separation Assistance Display," *IEEE Transactions on Systems, Man, and Cybernetics, part A: Systems and Humans*, Vol. 41, No. 6, 2011, pp. 863–875. doi:10.1109/TSMCA.2010.2093890.

[35] Ellerbroek, J., Brantegem, K. C. R., van Paassen, M. M., and Mulder, M., "Design of a Co-Planar Airborne Separation Display," *IEEE Transactions on Human-Machine Systems*, Vol. 43, No. 3, 2013, pp. 277–289. doi:10.1109/TSMC.2013.2242888.

[36] Hermes, P., Mulder, M., Van Paassen, M. M., and Huisman, H., "Solution-Space-Based Analysis of the Difficulty of Aircraft Merging Tasks," *Journal of Aircraft*, Vol. 46, No. 6, 2009, pp. 1995–2015. doi:10.2514/1.42886.

[37] Rahman, S. M. A., Borst, C., Mulder, M., and van Paassen, M. M., "Sector Complexity Measures: A Comparison," *Jurnal Teknologi*, Vol. 76, No. 11, 2015, pp. 131–139. doi:10.11113/jt.v76.5923.

[38] D'Engelbronner, J. G., Borst, C., Ellerbroek, J., van Paassen, M. M., and Mulder, M., "Solution Space-Based Analysis of Dynamic Air Traffic Controller Workload," *Journal of Aircraft*, Vol. 52, No. 4, 2015, pp. 1146–1160. doi:10.2514/1.C032847, URL `http://arc.aiaa.org/doi/10.2514/1.C032847`.

[39] Jenie, Y. I., van Kampen, E.-J., de Visser, C. C., Ellerbroek, J., and Hoekstra, J. M., "Three-Dimensional Velocity Obstacle Method for Uncoordinated Avoidance Maneuvers of Unmanned Aerial Vehicles," *Journal of Guidance, Control, and Dynamics*, Vol. 39, No. 10, 2016. doi:10.2514/1.G001715.

[40] Borst, C., Visser, R. M., van Paassen, M. M., and Mulder, M., "Exploring Short-Term Training Effects of Ecological Interfaces: A Case Study in Air Traffic Control," *IEEE Transactions on Human-Machine Systems*, Vol. 49, No. 6, 2019, pp. 623–632. doi:10.1109/THMS.2019.2919742.

[41] Westin, C., *Strategic Conformance: Exploring Acceptance of Individual-Sensitive Automation for Air Traffic Control*, PhD Thesis. Delft University of Technology, Netherlands, 2017. doi:10.4233/uuid.

[42] Matthews, B. W., "Comparison of the predicted and observed secondary structure of T4 phage lysozyme," *Biochimica et Biophysica Acta (BBA)-Protein Structure*, Vol. 405, No. 2, 1975, pp. 442–451. doi:https://doi.org/10.1016/0005-2795(75)90109-9.

[43] Powers, D. M., "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation," *Journal of Machine Learning Technologies*, Vol. 2, No. 1, 2011, pp. 37–63.

[44] Kohavi, R., "A study of cross-validation and bootstrap for accuracy estimation and model selection," *International Joint Conference on Artificial Intelligence*, Vol. 14, Montreal, Canada, 1995, pp. 1137–1145. doi:10.5555/1643031.1643047.