Conformal automation for air traffic control using convolutional neural networks

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

Publication date
2019

Document Version
Accepted author manuscript

Published in
Proceedings of the 13th USA/Europe Air Traffic Management Research and Development Seminar 2019, ATM 2019

Citation (APA)

Important note
To cite this publication, please use the final published version (if applicable).
Please check the document version above.
Conformal Automation for Air Traffic Control using Convolutional Neural Networks

J. Ellerbroek
The problem of conformance

Solving a simple conflict:
The problem of conformance

Put A behind B ...
The problem of conformance

... B behind A
The problem of conformance

... B in front of A
The problem of conformance

... or even changing both headings
Solution: Strategically conformal automation

Strategic conformance is “… the degree to which automation’s behavior and apparent underlying operations match those of the human.”

– Hilburn et al. (2014)
Previous research

1. Ability to adapt to controller preferences without full knowledge of underlying decision-making dynamics

2. Required feature engineering based on prior knowledge and assumptions

3. Only considered simple two-aircraft conflicts

Purpose of this study: predict controller actions in a more realistic setting
Train a model on controller actions using **supervised learning**

**Approach**

**Part A: Data generation**
- ATCo
- ATC simulator
- Dataset run 1-4

**Part B: Model training**
- K-fold validation
- Training data (run 1-3)
- Individual models
  - Type
  - Direction
  - Magnitude
- Supervised learning

**Part C: Measuring conformance**
- Test data (run 4)
- Model testing
- Performance metrics
- Consistency metric

**TABLE I: (Hyper)parameters during training.**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimization algorithm</td>
<td>Adam</td>
</tr>
<tr>
<td>Output activation</td>
<td>Softmax classifier</td>
</tr>
<tr>
<td>Loss function</td>
<td>Categorical entropy</td>
</tr>
<tr>
<td>Train/val/test ratio</td>
<td>60%/15%/25%</td>
</tr>
<tr>
<td>K-folds</td>
<td>5</td>
</tr>
<tr>
<td>Mini batch-size</td>
<td>32 samples</td>
</tr>
<tr>
<td>Steps-per-epoch</td>
<td>$\times$ training samples / batch-size</td>
</tr>
<tr>
<td>Epochs</td>
<td>30</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.01</td>
</tr>
<tr>
<td>Dropout rate</td>
<td>20%</td>
</tr>
<tr>
<td>Input image dimensions</td>
<td>128x128 px</td>
</tr>
</tbody>
</table>

**Figure 6:** Data generation and training & testing of the individual models for one participant. The dataset consists of input (SSD images) and target (commands) data. The models are used to predict a command for a given SSD image. Model performance is based on prediction accuracy.

**Figure 7:** Three validation steps for participant 1 (P1).

**Figure 8:** 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.

**Figure 9:** The achieved MCC scores per control variable. In this figure,...

**Table I: (Hyper)parameters during training.**

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimization algorithm</td>
<td>Adam</td>
</tr>
<tr>
<td>Output activation</td>
<td>Softmax classifier</td>
</tr>
<tr>
<td>Loss function</td>
<td>Categorical entropy</td>
</tr>
<tr>
<td>Train/val/test ratio</td>
<td>60%/15%/25%</td>
</tr>
<tr>
<td>K-folds</td>
<td>5</td>
</tr>
<tr>
<td>Mini batch-size</td>
<td>32 samples</td>
</tr>
<tr>
<td>Steps-per-epoch</td>
<td>$\times$ training samples / batch-size</td>
</tr>
<tr>
<td>Epochs</td>
<td>30</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.01</td>
</tr>
<tr>
<td>Dropout rate</td>
<td>20%</td>
</tr>
<tr>
<td>Input image dimensions</td>
<td>128x128 px</td>
</tr>
</tbody>
</table>

**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,...
Approach: Convolutional Neural Networks

Convolutional Neural Network (CNN)

Learned Features

Airplane
Cat
Hoverboard

Source: MathWorks – Deep Learning - Convolutional Neural Networks
Approach: Convolutional Neural Networks
Image convolution

Filter:

\[
\begin{bmatrix}
1 & 0 & 1 \\
0 & 1 & 0 \\
1 & 0 & 1 \\
\end{bmatrix}
\]
A whole convolution layer

<table>
<thead>
<tr>
<th>Input Volume (+pad 1) (7x7x3)</th>
<th>Filter W0 (3x3x3)</th>
<th>Filter W1 (3x3x3)</th>
<th>Output Volume (3x3x2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>x[:, :, 0]</td>
<td>w0[:, :, 0]</td>
<td>w1[:, :, 0]</td>
<td>o[:, :, 0]</td>
</tr>
<tr>
<td>0 0 0 0 0 0 0</td>
<td>-1 0 1</td>
<td>0 1 -1</td>
<td>2 3 3</td>
</tr>
<tr>
<td>0 0 0 1 0 2 0</td>
<td>0 0 1</td>
<td>0 -1 0</td>
<td>3 7 3</td>
</tr>
<tr>
<td>0 1 0 2 0 1 0</td>
<td>-1 1 -1</td>
<td>0 -1 -1</td>
<td>8 10 -3</td>
</tr>
<tr>
<td>x[:, :, 1]</td>
<td>w0[:, :, 1]</td>
<td>w1[:, :, 1]</td>
<td>o[:, :, 1]</td>
</tr>
<tr>
<td>0 0 0 0 0 0 0</td>
<td>-1 0 1</td>
<td>-1 0 0</td>
<td>-8 -8 -3</td>
</tr>
<tr>
<td>0 2 0 2 0 2 0</td>
<td>1 1 -1</td>
<td>1 -1 0</td>
<td>-3 1 0</td>
</tr>
<tr>
<td>0 2 1 2 2 2 2</td>
<td>1 1 1</td>
<td>1 -1 0</td>
<td>-3 8 -5</td>
</tr>
<tr>
<td>x[:, :, 2]</td>
<td>w0[:, :, 2]</td>
<td>w1[:, :, 2]</td>
<td>o[:, :, 2]</td>
</tr>
<tr>
<td>0 0 0 0 0 0 0</td>
<td>-1 1 -1</td>
<td>-1 1 -1</td>
<td>-1 0 0</td>
</tr>
<tr>
<td>0 2 1 2 0 1 0</td>
<td>1 1 0</td>
<td>0 -1 -1</td>
<td>0 0 0</td>
</tr>
<tr>
<td>x[:, :, 3]</td>
<td>b0[:, :, 0]</td>
<td>b1[:, :, 0]</td>
<td>0</td>
</tr>
<tr>
<td>0 0 0 0 0 0 0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

toggle movement
Approach: Choosing the right input

The Solution—Space Diagram

Step 1

Step 2

Step 3

Protected zone

Exit waypoint

$V_{con}$

$V_{int}$

$A_{int}$

$A_{con}$

$V_{min}$

$V_{max}$

$-V_{int}$
Approach: Choosing the right input

The Solution-Space Diagram
Approach: The model
Approach: Example of input layer
Three models for three control variables:

1. Resolution type (heading, speed, direct-to)
2. Resolution direction (left/right, up/down)
3. Resolution magnitude

INPUT: Channels: 3
Convolution Filters: 32
Downsampling
Convolution Filters: 64
Downsampling
Convolution Filters: 32
Flattening
Dense
Dropout
Dense
OUTPUT: Classes: 2 or 3

SIZE: 64x32x3 63x31x32 31x15x32 30x14x64 15x7x64 14x6x32 2688 1024 1024 2 or 3
Getting the data: Experiment setup

Human in the loop experiment:

Control a sector, while

1. Avoiding Loss-of-Separation between aircraft
2. Guiding the aircraft to their exit waypoint as efficiently as possible

Using either

1. Heading (HDG)
2. Speed (SPD)
3. Direct To (DCT)
Getting the data: Participants and runs

- 12 participants
  - Novice
  - 'Intermediate'

2 scenarios with 4 repeats
- scenario 1: 10 conflicts x 4
- scenario 2: 10 conflicts x 4

80 conflicts x 12 participants
Getting the results: Training models

- Three models for **type**, **direction**, and **magnitude**
  - Training data: run 1-3
  - Test data: run 4
Getting the results: Training models

K-fold validation: better performance for small datasets

Getting the results: Testing conformance

Two types of models: individual and general

Three tests: individual, cross-validation, and baseline
Getting the results: Measuring performance

Accuracy: 0.95
MCC: 0

\[
\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}
\]

\[
\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}
\]
Results
Results: Let’s start with some eyeballing

Participant 7, Scenario 1, Run 4

Participant 11, Scenario 1, Run 4
Results: Let’s start with some eyeballing
Results: Training performance

Fig. 8: Validation performance 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 per control variable. 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, 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 ($r = 0.75, p = 0.005$) 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 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 used.
Results: Individual conformance
Results: Individual conformance

Fig. 6: Data generation and training & testing of the individual models for one participant. The dataset consists of input (SSD images) and target (commands) data. The models are used to predict a command for a given SSD image. Model performance is based on prediction accuracy.

TABLE I: (Hyper)parameters during training.

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

Individual models were trained. In this section, the training phase is illustrated with an example of convergence of performance in the training phase. Subsequently, 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. Here, performance is measured using the MCC (see section III-E), which ranges between -1 and 1. Because negative correlation never occurred, all MCC result figures are clipped to a range of [0, 1].

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,
Results: Individual conformance

The effect of participant consistency on the performance of individual models was investigated. When a Pearson Correlation Coefficient test is applied across all folds and abstraction levels against the mean consistency, a notable correlation between consistency and model performance can be observed. To illustrate this, the trained model can be evaluated by observing the correlation they choose for their resolutions.

It was found that some participants are relatively consistent in their resolutions (participants 5, 7, and 10), whereas others are less consistent, particularly participant 8. However, when using participant data on which the model is trained, it is evident that individual model predictions are not equally effective across the entire population of participants.

The spread of the data indicates the maximum and minimum sufficient consistency behavior across different runs. It was observed that the model is low participant consistency: in some cases, the participant data on which the model is trained does not show sufficient consistency behavior across different runs. This suggests that the personal model is low, and predictions are not equally effective across the entire population of participants.

Figure 10 shows the normalized consistency (as defined in Section III-E) per participant and control variable. Here, it can be observed that while some participants are relatively consistent, others are less consistent. Nevertheless, the correlation between conflicts that do appear comparable in the SSD is relatively high.

The inter-participant test of the personalized predictions is indeed individual-sensitive. To test the models against all participant test datasets, Figure 12 shows the results of using the models of each participant on the test data of other participants. Figure 8 shows the training performance during training of P1's individual model. The spread indicates the maximum and minimum magnitude of predictions (mean = 0.64, SD = 0.12) for each participant type.

Test data P1 → Individual model P1 → Conformance

Input image dimensions 128x128 px
Dropout rate 20%
Learning rate 0.01
Epochs 30
Train/val/test ratio 60%/15%/25%
Loss function Categorical entropy
Output activation Softmax classifier

The neural network model reaches MCC scores of 0.21 during training, which is a relatively large amount compared to the achieved MCC scores per control variable. In this figure, the spread indeed individual-sensitive, is to test the models against all participant test datasets. Figure 12 shows the results of using the models of each participant on the test data of other participant test datasets. Figure 11: Participant consistency vs individual model performance.
Results: Consistency

![Bar chart showing consistency across different participants and control variables.](image)

- **Control variable**: Type, Direction, Magnitude, Mean
- **Consistency (normalized)**: -2, -1, 0, 1, 2
- **Participant**: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12

The chart illustrates the consistency across different participants and control variables. Each participant is represented by a unique set of bars indicating their performance for each control variable (Type, Direction, Magnitude) and the mean consistency. The normalized consistency values range from -2 to 2, providing a clear visualization of how consistent each participant is in their decisions across different SSD images.

**Key Observations**:
- Participants 5, 7, and 10 show the highest consistency, with mean values close to 0, indicating they choose their resolutions more uniformly.
- Participants 8, 9, and 10 are less consistent in their resolutions, with higher variability in their choices.
- The spread around each line depicts the range between the least and best performance for each participant, highlighting the variability in their decision-making processes.

**Analysis**:
- The personalized predictions are not equally effective across the entire population of participants, as indicated by the range of consistency scores.
- The models show more uniform MCC scores, regardless of which participant the model is applied to, suggesting that the personal model MCC can be found between participant consistency and individual model performance.
- The spread indicates the maximum and minimum performance for each fold per control variable, with MCC values ranging from 0.37 to 0.72.
- Conformance to the test datasets of each participant is high, as indicated by the MCC score of 0.64 with a standard deviation of 0.12.
- The training phase is successful in 'learning' from the data samples, with MCC values increasing over time, indicating the neural network's ability to adapt to the data.

**Conclusions**:
- The study highlights the importance of participant consistency in SSD resolution choices, with significant variability observed among participants.
- Personalized predictions are effective across different runs, indicating the models' adaptability to different decision-making processes.
- The results support the assumption that personal model performance is strongly influenced by participant consistency, with high levels of consistency leading to more uniform model performance.
Results: Consistency

![Graph showing the relationship between Participant Consistency and Mean MCC, with a linear regression line and R = 0.75.](image)

TABLE I: (Hyper)parameters during training.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimization algorithm</td>
<td>Adam</td>
</tr>
<tr>
<td>Output activation</td>
<td>Softmax classifier</td>
</tr>
<tr>
<td>Loss function</td>
<td>Categorical entropy</td>
</tr>
<tr>
<td>Train/val/test ratio</td>
<td>60%/15%/25%</td>
</tr>
<tr>
<td>K-folds</td>
<td>5</td>
</tr>
<tr>
<td>Mini batch-size</td>
<td>32 samples</td>
</tr>
<tr>
<td>Steps-per-epoch</td>
<td>( \times ) training samples / batch-size</td>
</tr>
<tr>
<td>Epochs</td>
<td>30</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.01</td>
</tr>
<tr>
<td>Dropout rate</td>
<td>20%</td>
</tr>
<tr>
<td>Input image dimensions</td>
<td>128x128 px</td>
</tr>
</tbody>
</table>

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,
Results: Cross-validation
Results: Cross-validation
Results: Cross-validation

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

Fig. 13: Performance (in MCC) of each participant’s individual models compared to the mean performance of five general models. The personalized approach is most effective for participant 1, whose individual models score 0.20 MCC higher than the baseline.
Results: Baseline validation

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 strategy by training a set of convolutional neural networks on a visual representation of traffic conflicts. A human-in-the-loop experiment was performed to generate training data for the model creation.

It is a common problem in machine learning that such 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. In addition, altitude differences were not taken into account, nor were altitude changes accepted as...
Discussion: Putting things into perspective

Limitations of the experiment

1. Participating ‘air traffic controllers’ are not professionals

2. Scenarios are constrained in conflict angles and altitude

3. Experiment runs still contain training effects

![Speed commands over time](image-url)
Discussion: Putting things into perspective

Suitability of SSD and machine learning approach

1. Convolutional neural networks converge but overfitting does occur

2. Higher-level decisions and information are not incorporated in the model

3. Neural networks remain a black-box approach
Conclusions: The silver lining

1. **SSD images** contain sufficient information to predict resolutions in horizontal conflict detection and resolution.

2. **Convolutional Neural Networks** are a feasible approach to achieve individual-sensitive automation.

3. Human controllers are **sufficiently consistent** to train a machine learning algorithm and are **strategy heterogeneous** as a group.
Conformal Automation for Air Traffic Control using Convolutional Neural Networks

J. Ellerbroek
Preliminary analysis

Simulation parameters:
- ATM simulator: BlueSky
- Resolutions: Heading changes only
- Max. number of A/C: Two
- Resolution algorithm: Modified Voltage Potential

Consistency metric

Type and direction:

\[
\text{consistency} = \max \left( \frac{\sum \text{class I}}{\sum \text{class I+II}}, \frac{\sum \text{class II}}{\sum \text{class I+II}} \right)
\]

Value:

\[
\text{consistency} = \frac{\sum \text{unique values possible}}{\sum \text{unique values used}}
\]
# Network Architecture

<table>
<thead>
<tr>
<th>Layer</th>
<th>Input</th>
<th>Filter size</th>
<th>Stride</th>
<th>Num filters</th>
<th>Activation</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conv2D</td>
<td>32x64x3</td>
<td>2x2</td>
<td>1</td>
<td>32</td>
<td>ReLU</td>
<td>31x63x32</td>
</tr>
<tr>
<td>MaxPool</td>
<td>31x63x32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>15x31x32</td>
</tr>
<tr>
<td>Conv2D</td>
<td>15x31x32</td>
<td>2x2</td>
<td>1</td>
<td>64</td>
<td>ReLU</td>
<td>14x30x64</td>
</tr>
<tr>
<td>MaxPool</td>
<td>14x30x64</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>7x15x64</td>
</tr>
<tr>
<td>Conv2D</td>
<td>7x15x64</td>
<td>2x2</td>
<td>1</td>
<td>32</td>
<td>ReLU</td>
<td>6x14x32</td>
</tr>
<tr>
<td>Flatten</td>
<td>6x14x32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>2688</td>
</tr>
<tr>
<td>Dense</td>
<td>2688</td>
<td></td>
<td></td>
<td></td>
<td>ReLU</td>
<td>1024</td>
</tr>
<tr>
<td>Dropout</td>
<td>1024</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1024</td>
</tr>
<tr>
<td>Dense</td>
<td>1024</td>
<td></td>
<td></td>
<td></td>
<td>Softmax</td>
<td>3</td>
</tr>
</tbody>
</table>
### Network Training

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Optimization algorithm</td>
<td>Adam</td>
<td>-</td>
</tr>
<tr>
<td>Output activation</td>
<td>Softmax classifier</td>
<td>-</td>
</tr>
<tr>
<td>Loss function</td>
<td>Categorical entropy</td>
<td>-</td>
</tr>
<tr>
<td>Train/val/test ratio</td>
<td>60/15/25</td>
<td>-</td>
</tr>
<tr>
<td>K-folds</td>
<td>5</td>
<td>-</td>
</tr>
<tr>
<td>Mini batch-size</td>
<td>32</td>
<td>samples</td>
</tr>
<tr>
<td>Steps-per-epoch</td>
<td>$2 \times$ training samples / batch-size</td>
<td>samples</td>
</tr>
<tr>
<td>Epochs</td>
<td>30</td>
<td>-</td>
</tr>
<tr>
<td>Learning rate</td>
<td>0.01</td>
<td>-</td>
</tr>
<tr>
<td>Dropout rate</td>
<td>20</td>
<td>%</td>
</tr>
<tr>
<td>Input image dimensions</td>
<td>128x128</td>
<td>px</td>
</tr>
</tbody>
</table>
Crossing conflicts

$A_{\text{con}}$  

0° - 45°  

45° - 135°  

135° - 180°  

$A_{\text{int}}$  

same-path  

head-on  

crossing
Increase of MCC and accuracy by using individual models
Performance of P4’s model