NHL STENDEN
computer vision & data science

Anomaly Detection for UAV Automatic Safe Landing

NHL Stenden Professorship in Computer Vision & Data Science

Jenny Nicolai

Supervisors: Maya Aghaei Gavari and Jaap van de Loosdrecht

Abstract—With new regulations around unmanned aerial vehicles (UAV) use in the European Union, the drone industry is set to bloom. This led to *theBEAST* project, which aims at the automatic fail-safe landing for UAVs. This technical paper tackles a specific part within the module, which revolves around anomaly detection to avoid obstacles for fail-safe landing, where the CutPaste anomaly detection framework is used. For this framework new alterations are introduced, CutPaste-multiple and CutPaste-normalscar. Various experiments try to maximize the model's performance, including choosing the tile size, CutPaste alteration method, hyperparameter tuning, and different data augmentation techniques. The best-performing model achieved an F1-score of 0.8036. Although anomaly detection looks like a suitable approach to resolve this problem, some limitations must be overcome before being used in a real-life application.

Index Terms—anomaly detection, ResNet-18, UAV, drone, CutPaste framework, fail-safe landing, BVLOS

1 Introduction

The drone industry is rapidly growing worldwide. The estimates of this industry vary [1]. However, in the European Drones Outlook Study [2] there is an estimate that by the year 2050, around 400.000 drones will be providing services in our sky.

Strict regulations have hindered the growth of unmanned aerial vehicles (UAV) in commercial activities in most European countries. However, with recent publication of new drone regulations in all European Union members [3], this has been set to change. This change led to the emergence of *Towards the first and Best EU-approved Autonomous Security drone for BVLOS flighT* (The BEAST) project.

This project is a sub-project of The BEAST, which revolves around *object avoidance for fail-safe landing* where the goal is to detect objects which obstruct a safe landing. An unsuccessful landing could, for example, result in a crash or endanger other lives. What these objects are may not always be known beforehand. Hence, we opt to use anomaly detection.

Anomaly detection can be used in many different applications, including medical image analysis [4] aerial surveillance [5] or defect detection [6]. Anomaly detection addresses unique challenges due to the nature of the problem because it is difficult to obtain anomalous data. Because the distribution of the anomalous data is unknown beforehand, models are trained to learn the distribution of the normal data and determine an anomaly if this normal distribution does not represent the test data.

In the drone industry, there are different scenarios regarding the visibility of drones. The first scenario is Visual Line of Sight (VLOS). These flights are operated within the pilot's line of sight; the drone is seen at all times by the pilot. The second term is Extended Visual Line of Sight (EVLOS), which means that a trained observer always has to see the drone, but this does not have to be the pilot. The last one is Beyond Visual Line of Sight (BVLOS), where the flights are beyond the visual range. In this case, the pilot only has the information provided by the UAVs and does not have to see the drone at all times. Therefore, autonomous anomaly detection for safe flying and landing would reduce the risk of an unsuccessful landing [7].

With the BVLOS flight, the UAV is deprived of the ability of the pilot to detect and avoid obstacles. Therefore, a system must be developed to guide the drone along its trajectory and the landing. Nevertheless, this system has to be light enough to be onboard a UAV. Furthermore, the system should also be autonomous as no human support will be available on sight, whereby computation speed and fast response are crucial for a safe flight.

This research focuses on adapting an anomaly detection algorithm to detect and avoid ground obstacles while minimizing the risks of crashing and causing damage. We choose an anomaly detection method to predict unsafe outliers from the safe landing areas for this task.

This research will address anomaly detection for automatic fail-safe landings by answering the main research question:
How can the CutPaste anomaly detection method be used for robust obstacle avoidance in camera-equipped UAVs?

- *What is the effect of tile sizes for detecting anomalous regions in an image?*
- *What hyperparameters of the anomaly detection model have impact on the performance of the model?*
- *What is the performance of the anomaly detection model when data augmentation is performed?*

2 State of the art

The research community has studied detection of safe landing spots for unmanned aerial vehicles and anomaly detection in the past.

2.1 Safe landing

These papers include mostly traditional Computer Vision techniques. In [8] robust homography estimation to correspond a plane to the land was used. The image is thresholded to separate the plane from the obstacles if a plane is detected. Finally, a probabilistic grid is created where the results are stored to make a complete view of the flight. However, this research has a fundamental flaw; it assumes that obstacles are static.

In [9], vision-based detection is used. In this paper, a helipad as a landing target is used. The helipad consists of a large circle with an H-shaped mark in it. The first step to preprocessing the images is converting the RGB images to a greyscale image. After that, the

- *Jenny Nicolai is a master student Computer Vision Data Science at the NHL Stenden University of Applied Sciences, E-mail: jenny.nicolai1@student.nhlstenden.com.*
- *Maya Aghaei Gavari is a senior researcher at the NHL Stenden Professorship in Computer Vision & Data Science, E-mail: maya.aghaei.gavari@nhlstenden.com.*
- *Jaap van de Loosdrecht is a professor at the NHL Stenden Professorship in Computer Vision & Data Science, E-mail: jaap.van.de.loosdrecht@nhlstenden.com.*

images are smoothed using a normalized box filter and provided to a Canny operator to detect edges. Instead of an H-shaped mark, multiple circles or ellipses [10, 11] can also be used to determine the drone's height to the ground and where it has to land. There must be specific landing spots for the drone to land on in these studies. The landing spots are not applicable in this research. Yu et al. [12] focused on the height above the ground with a small helicopter. With a 3D camera, a height estimation is made. Here it is assumed that the ground surface is mostly flat. This study proposed a plane-fitting method to map the high-dimensional depth data into a one-dimensional measurement. However, there existed errors in feature point detecting and matching when the helicopter was high above the ground.

In [5] anomaly detection for aerial surveillance is used. Deep neural networks were used to surveillance critical infrastructures, like airports or harbors. Objects are annotated with a bounding box. All the images are converted to a grid, and this grid contains the centers of the bounding boxes. The used model is UAV-AdNet. Unlike most models, GPS and image data are combined. The anomalies were known beforehand, which is not the case for this research.

2.2 Anomaly detection

Various methods for anomaly detection have been studied, such as reconstruction-based autoencoders [13] or deep one-class classifiers [14], where the deep one-class outperforms the autoencoder.

PANDA [15] is introduced for pretrained anomaly detection adaption. It combines pretrained features with simple anomaly detection, convincingly outperforming much more complex methods, where the primary limitation lies in the requirement of strong pretrained feature extractors. Unfortunately, generic feature extractors are not currently available for all data modalities.

In [6] self-supervised learning with CutPaste is proposed. CutPaste alteration is cutting a random patch of the image and pasting it back in a random spot. The model is trained and tested on the MVTec dataset, containing textured and object images. An AUC of 96.1 is achieved. CutPaste shows promising results and is worth further exploring.

Defard et al. [16] introduced PaDiM: patch distribution modeling. PaDiM uses convolutional neural networks for patch embedding and multivariate Gaussian distributions to get a probabilistic representation of the normal class. PaDiM uses pretrained CNN, so no deep learning training is required. PaDiM will be tried on a model trained in this research to produce the heatmaps to look at the localization of the anomalies.

3 Materials and Methods

This section will cover the materials and methods that are used throughout the research and are essential for creating the results. This includes the dataset, anomaly detection framework, evaluation metrics and data augmentations. Appendix A shows the hardware used throughout this research.

3.1 Dataset

The public dataset AeroScapes [17] is used for this research. The AeroScapes benchmark for aerial semantic segmentation consists of images captured by a commercial drone from an altitude range of 5 to 50 meters. The dataset provides 3269 colour images (RGB) with sizes 720x1280 pixels and ground-truth masks for 11 classes. Figure 1 shows an example of the images and the segmentations and which classes there are, including a background class.

For this research it is not necessary that all classes are explicitly divided. Because this project focuses on safe landing, there only has to be a division between safe and unsafe landing spaces. To achieve this distinction, certain classes will be combined, which leads to the AeroScapes-modified dataset. The safe class contains: background, vegetation, road and sky. The unsafe class contains: person, bike, car, drone, boat, animal, obstacle and construction. This distinction in safe and unsafe is shown in Figure 1 in the last columnn. The safe class is the normal class and the unsafe class is the anomalous class.

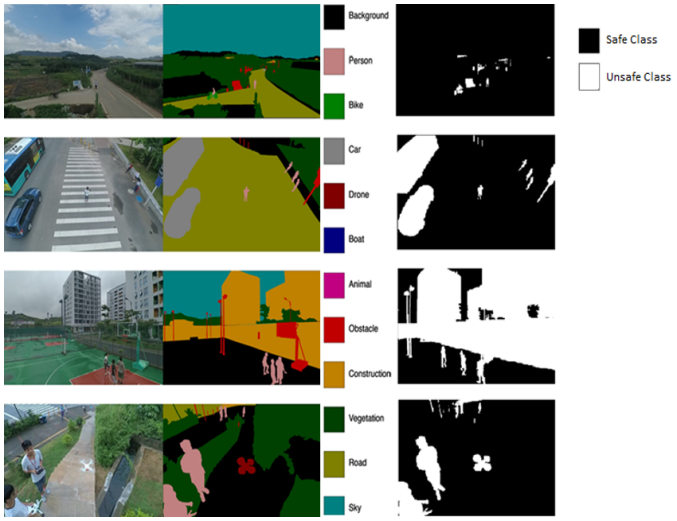


Fig. 1: The AeroScapes Sementic Segmentation Dataset captures outdoor scenes using a drone. The ground truth segmentation maps for 11 different classes including a background class [17]. The last column shows the distinction between the safe and unsafe class

Table 1 shows the distribution between the safe and unsafe class in the AeroScapes-modified dataset.

Table 1: Distribution between the safe and unsafe classes in the AeroScapes-modified dataset.

Class	Number of pixels	Percentage
Safe	2,832,973,493	94.03%
Unsafe	179,736,907	5.97%

The dataset is split following a 60/20/20 ratio which results in 1962, 654 and 653 images for training, validation and testing, respectively.

3.1.1 Tiling

The AeroScapes-modified dataset contains images of 720x1280 pixels. Such large images cannot be fed to the GPU directly. Furthermore, for anomaly detection, we are interested in the abnormal regions of the image instead of the image as a whole. With the use of tiling, this can be accomplished.

The dataset is tiled using fixed tiling with overlap. With a given height, width, and overlap margin, the tiles have a fixed grid format. To illustrate this, in Figure 2, a few tiles are plotted onto the image. The coordinates of the tiles are the same for all the images in the dataset.

The segmentation dataset AeroScapes-modified is converted to a classification dataset. With the ground-truth segmentation mask it is determined if a tile contains the safe or unsafe classes. If a tile only contains the safe class, it is referred to as a *normal tile*. If there are pixels from the unsafe class, the tile is referred to as as *anomaly tile*. This process is shown in Figure 2. The first tile only contains safe pixels, so it is assigned the normal class. The second tile falls partly on top of the car, the unsafe pixels, so this will be assigned the anomaly class. This goed on for all the tiles in the image.

In training, negative fixed tiling is used. This means that only the normal tiles are used. Validation and testing keep both the normal and anomaly tiles.

Different tile sizes are used to train the model. This is because different anomalies have different sizes. If a large anomaly falls into a small tile size, it could lead to information loss because not the whole anomaly is in one tile. The overlap should handle the anomalies that would otherwise be cut in half. All tiles are square and have 25% overlap with the next tile. Table 2 shows the number of normal and

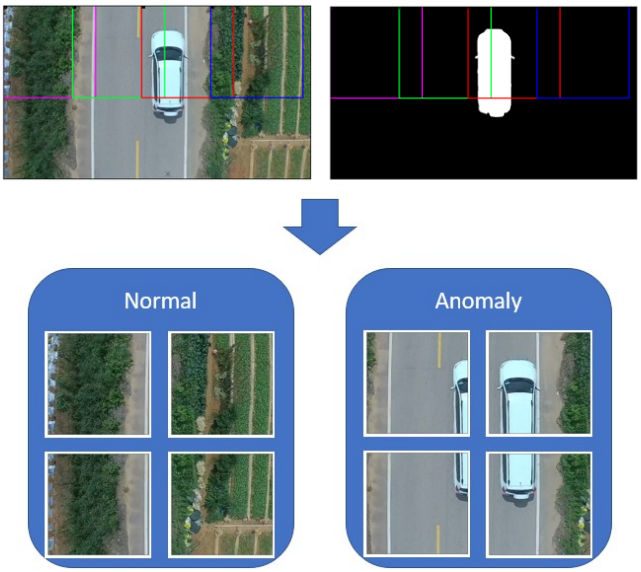


Fig. 2: Example of fixed tiling with overlap. A few tiles are shown on the RGB and the ground-truth images. After that, each tile gets assigned the class 'normal' or 'anomaly' based on the safe and unsafe pixels in the tile.

anomaly tiles for training, validation, and testing for different tile sizes.

Table 2: Number of images for each tile size in training, validation and testing where validation and testing are divided between normal and anomaly tiles.

	Train	Validation		Test	
Tile size	Normal	Normal	Anomaly	Normal	Anomaly
128x128	134,898	45,038	14,476	44,838	14,588
256x256	32,338	10,797	7515	10,776	7508
384x384	12,082	4079	5731	4053	5742

3.2 CutPaste anomaly detection framework

The CutPaste anomaly detection framework addresses anomaly detection as a binary classification between normal and anomalous classes. However, it is hard to train a model with full supervision because anomalous data is often missing. Hence, the anomaly detection framework with the CutPaste alternation as self-supervised representation learning is introduced in [6]. CutPaste selects a small rectangular patch randomly in the image (Cut) to be pasted back to another random spot in the image (Paste). An example of this alteration on a tile from the AeroScapes-modified dataset is shown in Figure 5a.

To reformulate this as a classification problem, about 50% of the train data is altered with the CutPaste alteration, while the other half remains normal. ResNet-18 [18] is trained to distinguish images between the altered and normal distribution. This process is shown in 3. This CutPaste alternation strategy aims to create irregular patterns on the normal data, with the patterns resembling a possible anomaly in the image.

3.2.1 Anomaly score

The anomaly score determines whether an image is normal or an anomaly. There exist various ways to compute anomaly scores. In this work, a Gaussian density estimator (GDE) [19] is used to compute the distances.

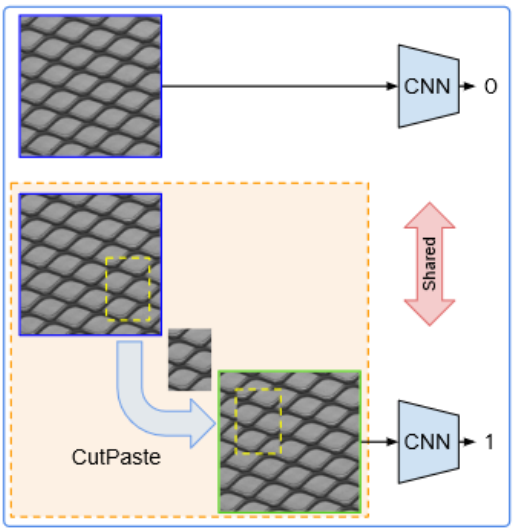


Fig. 3: Learning self-supervised representation [6].

The GDE is fitted with the normal training data. Afterwards, the validation data is used to calculate the threshold distance for the classification. The validation data consists of anomalous and normal data. For each image in these classes, the distance is calculated. The average distance for both classes is computed, after which the average between those distances is determined. An example of this process on a small dataset is shown in Figure 4. This threshold distance is used for classifying the test data.

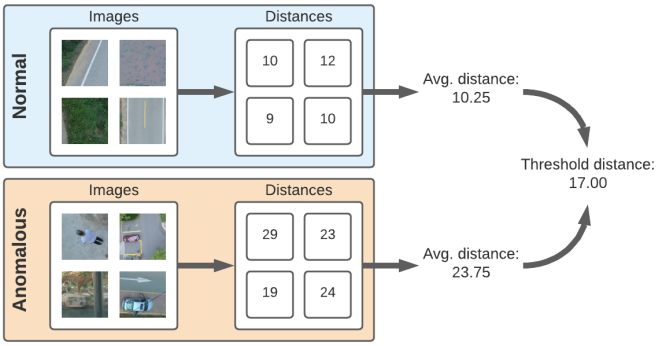


Fig. 4: The distance is calculated for every image in the validation data. For those distances is the average calculated per class. The threshold is the average distance between the average distance per class. In this example it means that if the distance of a test image is above 17.00 it is classified as an anomaly. Below 17.00 means it is classified as normal.

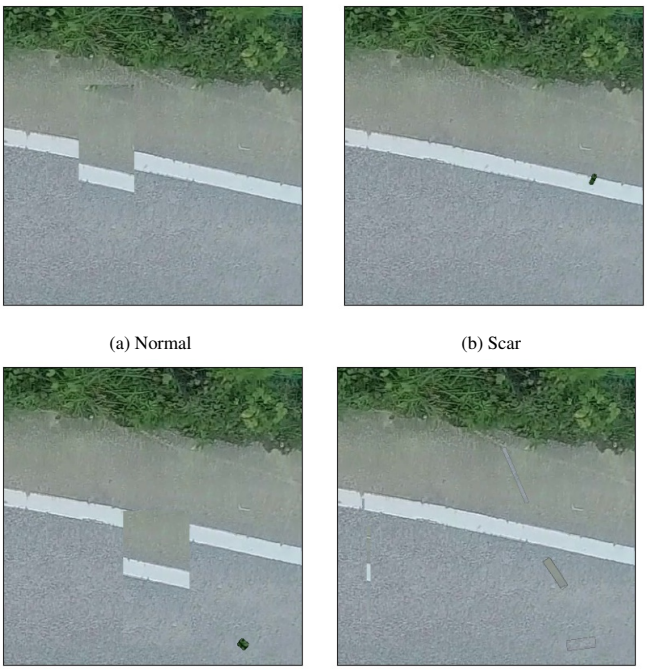
3.2.2 CutPaste variants

In [6] CutPaste and CutPaste-scar are introduced. As mentioned before, CutPaste selects and pastes a random patch on the image. CutPaste-scar is a variation on this, where the patch is much smaller and rotated. An example of the CutPaste-scar is shown in Figure 5b.

Additionally, we introduce variations on the original alterations. The first alteration is the combination between CutPaste and CutPaste-scar. Although CutPaste and CutPaste-scar share a similarity but are also different in size and shape. The combination of the two could exploit the impact of both of them jointly. Where in the original paper [6] a 3-way classifier is introduced, we propose a 4-way classification: altered CutPaste, altered CutPaste-scar, altered

CutPaste & CutPaste-scar and unaltered. For the sake of comparison, we also propose the 2-way classification of the altered and unaltered images.

The second variation we introduce is the CutPaste-multiple. Instead of only one patch from each image as introduced in the original paper, multiple patches are pasted onto the image. There are three different parameters. The first parameter is the number of patches, this is a random number in a predefined interval. The next parameter is the size of the patches, each patch has a random area ratio. The last parameter is the rotation angle, this is a random rotation in the interval $(-45°, 45°)$. Each patch has a chance of 50% of rotation. Because the patches are pasted iteratively, they could overlap on the altered image. An example of CutPaste-multiple is shown in Figure 5d.



(a) Normal (b) Scar

(c) Normal & Scar (d) Multiple

Fig. 5: The different CutPaste variants.

3.3 PaDiM: Patch distribution modeling

For the visualization of the localization of the anomalies, Patch Distribution Modeling (PaDiM) [16] is used. PaDiM makes use of a pretrained ResNet-18 [18]. During the fitting phase, each patch of the images is associated to its spatially corresponding activation vectors. The activation vectors from layers 3 and 4 of the trained ResNet-18 are concatenated to get embedding vectors carrying information from different semantic levels. The generated embedding vectors may carry redundant information. In [16] is noticed that randomly selecting few dimensions is more efficient than a Principal Component Analysis [20]. This simple dimensions reduction significantly decreases the complexity of the fitting and testing time while maintaining the performance.

The parametric representation for the normal class is obtained with multivariate Gaussian distributions. Finally, the patch embedding vectors from the test images are used to predict the anomaly map with the help of the learned representations of the normal class.

An anomaly map is generated for each tile, these maps are stitched back together for the complete image. Because the tiles contain overlap, the highest pixel value in the overlapping parts is chosen. This generated anomaly map can be converted to a heatmap. The red and yellow colors in the heatmap correspond to detected anomalies, whereas blue areas indicate the normal class. Section 4 shows several heatmaps as a result from the anomaly detection model, these can be found in Figure 8.

3.4 Evaluation metrics

This subsection describes the evaluation metrics used. The four possible outcomes in the classification are as follows:

- **True positive (TP):** An anomaly tile is classified as an anomaly
- **False positive (FP):** An anomaly tile is classified as normal
- **True negative (TN):** A normal tile is classified as normal
- **False negative (FN):** A normal tile is classified as an anomaly.

Precision and recall are metrics for model evaluation. Precision is the ratio between the true positives and all the positive predictions. Recall is the measure of the model correctly predicting the positives.

$$\text{Precision} = \frac{TP}{TP+FP} \qquad \text{Recall} = \frac{TP}{TP+FN} \tag{1}$$

F1-score is the harmonic mean between recall and precision. It creates a balance between the two metrics. The formula for the F1-score is given in Equation 2

$$\text{F1-score} = 2 * \frac{\text{Recall * Precision}}{\text{Recall + Precision}} \tag{2}$$

AUC is the area under the ROC curve. AUC ROC indicates how well the predictions of the anomalies are separated from the normal data. The F1 score is chosen as the most important metric in this study. For completeness, all metrics are reported.

3.5 Data augmentations

To increase the performance of the model data augmentations can be used. The augmentations used are from the Albumentations [21] library. There are six different augmentations chosen to train with that seem useful. Because the UAV could encounter different weather conditions, different augmentations for weather are tested. To change the scenery, Gamma and random rotation are also used. The augmentations used are listed below. Examples of the augmentations are shown in Figure 6.

- Random rain
- Random fog
- Random snow
- Random sun flare
- Gamma correction with random gamma
- Random rotation of multiples of 90 degrees

4 Experiments & Results

This section aims to describe the experiments that were performed to answer the research questions of this project, as well as showcasing their results.

Each section focuses on a specific parameter, where the other parameters are untouched to enhance the comparability. All the experiments have a given stop criterion: if the model does not improve for 20 consecutive epochs, the model is stopped. The maximum number of epochs is defined at 512. ResNet-18 uses pretrained weights from the ImageNet [22] dataset and uses a freeze for the first 20 epochs.

(a) Original (b) Rain (c) Fog (d) Snow (e) Sun flare (f) Gamma correction (g) Rotation of 90 degrees

Fig. 6: Data augmentations used. Figure (a) shows the original image, figures (b)-(g) show the augmented images.

Table 3: The baseline parameters of the model for the experiments.

Optimizer	Scheduler	Learning rate	CutPaste technique
SGD	Cosine annealing warm restarts, 128	0.0001	Normal

4.1 Tile size

The baseline parameters for the model are shown in Table 3.

In the first experiment, metrics from different tile sizes are compared with each other. As discussed in Section 3.1, the AeroScapes-modified dataset is tiled three times with different tile sizes. Table 4 shows the results of the three runs. The best metrics are bold-faced. As shown in Table 4, the run with the tile size of 384x384 pixels yielded the highest F1-score compared to the first two experiments, with a performance gain of about 15% and 5% respectively. On top of this, it is notable that all the recall values between the experiments are relatively similar, whereas the precision values vary broadly. Therefore, the biggest tile size (run 3) is chosen to use in further experiments.

Table 4: Parameters and metrics from the tile size experiments.

Run	Tile size	F1	Recall	Precision	AUC
1	128x128	0.5783	0.6533	0.5187	**0.8119**
2	256x256	0.6786	**0.6767**	0.6806	0.8091
3	384x384	**0.7260**	0.6480	**0.8254**	0.8094

4.2 CutPaste variant

The next experiment is about choosing the best performing CutPaste variant. Table 5 shows the variant and the corresponding number of patches and the patch sizes.

Table 5: Parameters for the different CutPaste variant runs

Run	CutPaste variant	Number of patches	Patch size
-3-	Normal	1	2-15%
4	Scar	1	0.01-0.25%
5	Normal-Scar (4-way)	1-2	2-15% & 0.01-0.25%
6	Normal-Scar (2-way)	1-2	2-15% & 0.01-0.25%
7	Multiple	2-5	0.1-0.5%

The results of these runs are shown in Table 6. The F1 scores of the CutPaste-scar and CutPaste-multiple method are quite similar to each other. Because CutPaste-multiple has the highest scores for precision and AUC, this is chosen to experiment further with, which corresponds to run 7.

Table 6: Metrics from the different CutPaste variant runs

Run	F1	Recall	Precision	AUC
-3-	0.7260	0.6480	0.8254	0.8094
4	0.7594	**0.6811**	0.8580	0.8409
5	0.7349	0.6536	0.8394	0.8212
6	0.7520	0.6724	0.8531	0.8383
7	**0.7597**	0.6771	**0.8653**	**0.8532**

4.3 Learning rate and optimizer

The learning rate is an important parameter. It controls how quickly the model is adapted to the problem. If the learning rate is too high, it could cause the model to converge too quickly to a sub-optimal solution, whereas a learning rate that is too small could cause the process to get stuck. Therefore, three different learning rates are chosen to experiment with.

Optimizers are algorithms that change attributes like network weights to reduce losses. SGD, Adam, and Adamax [23] are tested and compared.

The learning rate scheduler is a predefined framework that adjusts the learning rate based on the number of epochs. The scheduler that is tested is the Cosine Annealing Warm Restarts [24]. The learning rate is set with a cosine annealing schedule which restarts after a specified amount of epochs. Table 7 shows the different experiments.

The experiment is split into three different sections. The first section determines the learning rate, the second section looks at the different optimizers, and lastly, a different amount of epochs is tried for the scheduler.

Table 7: Parameters for the learning rate and optimizers runs.

Run	Learning rate	Optimizer	Scheduler (restarts)
-7-	0.0001	SGD	128
8	0.001	SGD	128
9	0.01	SGD	128
10	0.0001	SGD	-
11	0.0001	Adam	-
12	0.0001	Adamax	-
13	0.0001	SGD	64
14	0.0001	SGD	32

As shown in Table 8, the learning rates of 0.001 and 0.0001 have similar results, where the lower learning rate is a bit better in all of the metrics. However, the three different optimizers perform quite differently, where the SGD has the best results.

Runs 7, 10, 13 and 14 all use the SGD optimizer with the same learning rate. However, the scheduler is set at a different number of epochs, where the restart after 64 epochs performs the best. Run 13 will be used in further experiments.

Table 8: Metrics from the learning rate and optimizers experiments

Run	F1	Recall	Precision	AUC
-7-	0.7597	0.6771	**0.8653**	0.8532
8	0.7562	0.6724	0.8638	0.8506
9	0.6870	0.5700	0.8643	0.8121
10	0.7490	0.6641	0.8587	0.8459
11	0.4903	0.3769	0.7012	0.5972
12	0.5314	0.3939	0.8163	0.7377
13	**0.7634**	**0.6834**	0.8634	**0.8535**
14	0.7563	0.6742	0.8612	0.8525

4.4 CutPaste parameters

As mentioned before in Section 3.2.2, different parameters for the CutPaste can be changed.

4.4.1 Occurrence

The occurrence of the CutPaste is set to three different values. An occurrence value of 0.7 means that there is a 70% chance an image gets the augmentation. The results are shown in Table 9.

Table 9: Metrics from the CutPaste occurrence chance

Run	Occurrence	F1	Recall	Precision	AUC
-13-	0.5	**0.7634**	**0.6834**	0.8634	**0.8535**
15	0.3	0.7538	0.6742	0.8547	0.8440
16	0.7	0.7604	0.6776	**0.8662**	0.8520

The results are quite similar, where the change of 0.5 performs slightly better.

4.4.2 Number of patches and patch size

Image alteration influence the model's performance. Therefore, multiple experiments with different numbers of patches and patch sizes are tested. All of those experiments can be found in Appendix B.

For the experiments discussed in Appendix B, the number of patches and the patch sizes are chosen manually. To substantiate these numbers, some statistics are calculated. First, the number of anomalies per image and the anomaly size are extracted from the train data. To find the number of anomalies, contours are detected in the annotated images, and the area for each contour is calculated. The distribution of the anomalies and the sizes of the anomalies is shown in Figure 7. The distribution of the anomaly size is plotted between 0% and 1% for a better visualisation.

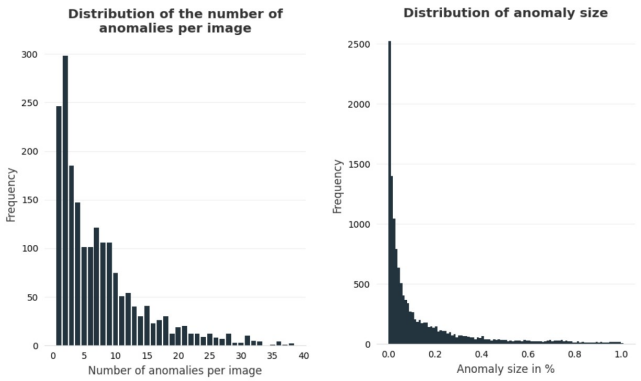


Fig. 7: The distribution of the number of anomalies and the sizes of the anomalies in the train data.

Table 10: Parameters for the number of patches and patch size experiment.

Run	Number of patches	Patch size
-13-	2-5	0.1-0.5%
17	5-60	0.1-0.5%
18	Distribution	Distribution

These distributions are used to randomly choose the number of patches and the patch size for training the model. Table 10 shows the number of patches and the patch size for the different experiments. Run 17 is the best performing run from the experiments in Appendix B. Table 11 shows the results from the runs, where run 17 yields the best results.

Table 11: Metrics from the number of patches and patch size experiment.

Run	F1	Recall	Precision	AUC
-13-	0.7634	0.6834	0.8634	0.8535
17	**0.7788**	**0.7081**	0.8651	0.8608
18	0.7692	0.6872	0.8734	0.8643

4.5 Data augmentation

This section focuses on the effect of image augmentation on the model's performance. As mentioned in Section 3.5, different augmentations are applied to the data. For each run, one augmentation can be applied and has a change of 50% for occurring. Table 12 shows the results of the runs.

Table 12: Metrics from the augmentation experiments.

Run	Augmentation	F1	Recall	Precision	AUC
-17-	None	0.7788	0.7081	0.8651	0.8608
19	Rain	0.7732	0.6987	0.8654	0.8610
20	Fog	0.7712	0.6958	0.8651	0.8620
21	Snow	0.7728	0.7088	0.8495	0.8535
22	Sun flare	**0.8036**	**0.7328**	**0.8895**	**0.8878**
23	Gamma	0.7768	0.7029	0.8680	0.8624
24	Rotation	0.7772	0.7072	0.8624	0.8600

The only augmentation that outperforms the model without augmentation is the sun flare. With an F1 that is almost 2.5% higher, run 22 gets by far the best results.

To test the robustness of the model, the best run is trained multiple times. The parameters of run 17 are run 5 times, each with a different seed. The results of those runs are available in Appendix C. This shows that the standard error in F1 score is approximately 0.0006.

The visualizations of the anomaly localization are shown in Figure 8. Three different images are shown with corresponding ground truth and heatmap. The first and second rows show quite good results in detecting the anomalies. The last row shows less promising results, the anomalies are large, and the model has problems detecting them.

5 Discussion, Conclusion & Future Work

This section sums up the paper by describing the discussing and conclusion and proposing future work.

5.1 Discussion

The first experiment is about tile size. Tile size is an important factor in the performance of the model. A tile size of 384x384 gives an initial F1 of 0.7260, whereas a tile size of 128x128 gives an F1 of 0.5783.

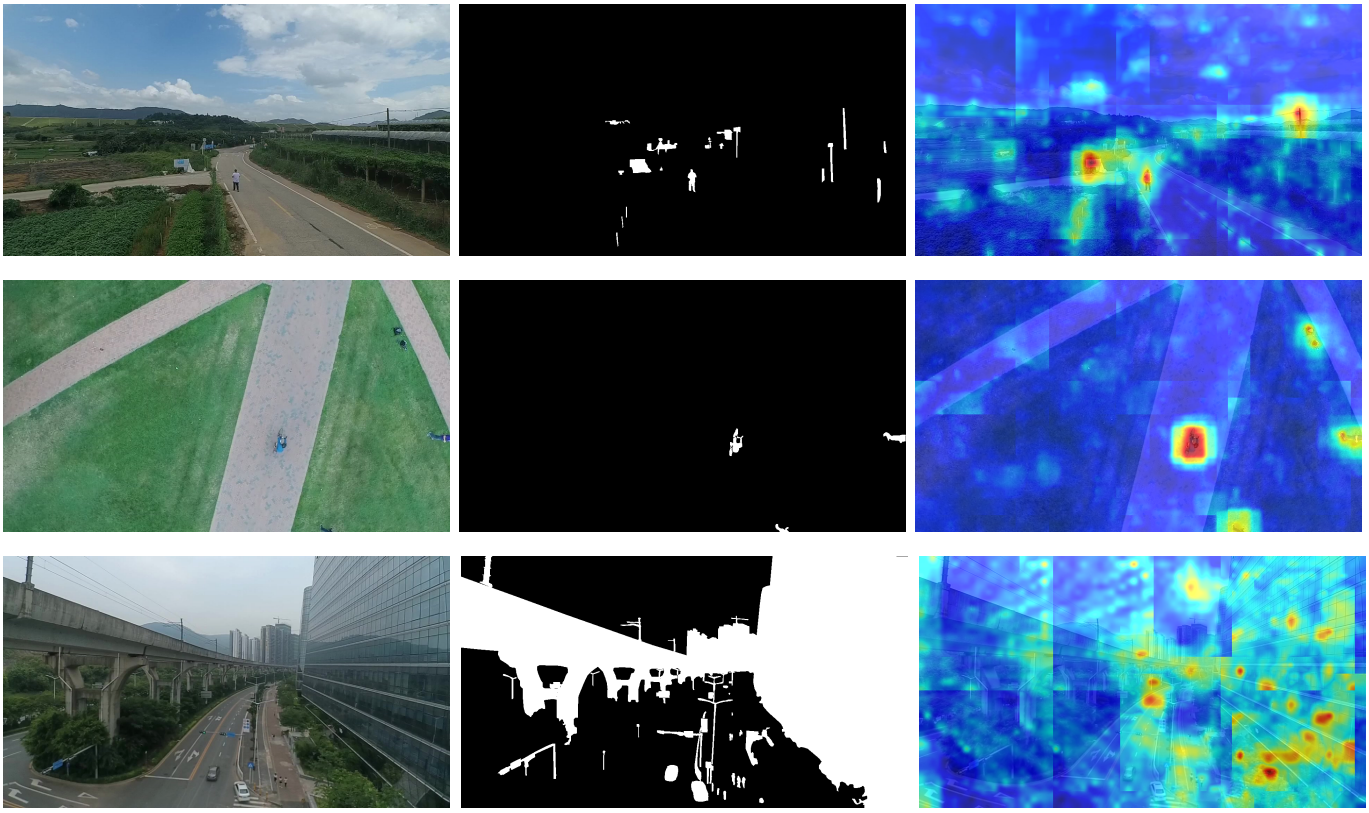


Fig. 8: Image samples. The left column shows the RGB image. The middle column is the ground truth from the corresponding RGB image, where white corresponds to the anomalous areas. The right column is the heatmap obtained by the model. Red and yellow areas indicate anomalous data, whereas blue means that normal data is detected. The heatmaps contain some artifacts where the tiles are stitched back together; in the future, efforts can be dedicated to eliminate these artifacts using techniques such as smoothing.

The size of the tile is dependent on the size of the anomalies and the number of anomalies in the dataset.

The CutPaste alteration is the second experiment. The original CutPaste has the lowest F1 out of all of them. The CutPaste-scar has better performance and yielded the highest recall. The combination between CutPaste and CutPaste-scar has a different performance between the 4-way or 2-way classification. Here, the 2-way performs better. The last experiment is the CutPaste-multiple, which outperforms the other variants. It yielded the highest F1, precision and, AUC. Therefore, CutPaste-multiple is chosen for further experiments, even though CutPaste-scar has similar results with the F1 score.

The third experiment is about the learning rate and the effect of different optimizers. The difference in the performance is slim between the learning rates of 0.001 and 0.0001, where 0.0001 is chosen. Optimizers had a significant effect on the performance. SGD outperforms Adam and Adamax, with a difference of about 0.25 and 0.22 in F1, respectively. The use of a scheduler seems to improve the performance of the model. Only the Cosine annealing warm restarts scheduler is tested with different numbers of epochs when a restart is performed. The results of different epochs are quite similar, but 64 seems to be the best.

CutPaste has its parameters, the occurrence, number of patches, and the patch size must be determined. The occurrence of the augmentation has the best value at 0.5, so the training data is evenly divided. The number of patches and patch size impact the model's performance, but the results are similar. Based on the F1 score, the best performing model has between 5 and 60 patches between 0.1% and 0.5%. The AUC shows that the model with the patch distributions got better results.

Answering the last subquestion, data augmentation could improve the model's performance if the proper augmentation is chosen. The Random Sun flare improved the model's score for the six different augmentations tested. The model without any augmentations scored an F1 of 0.7788, whereas all the different augmentations also scored quite similarly, around 0.77.

5.2 Conclusion

The results aim to answer the question of this research, which is *How can the CutPaste anomaly detection method be used for robust obstacle avoidance in camera-equipped UAVs?*. There are no real-time tests performed to see how the model would perform. However, from analyzing the metrics and visual outputs from the model, there could be concluded that CutPaste gives promising results. Under close supervision, it is worth trying in a real-life situation.

The first subquestion is *What is the effect of tile sizes for detecting anomalous regions in an image?*. As seen in Section 4.1, tile size has a significant impact on the F1 score and the precision of the model. This could be because the size of the anomalies impacts the size of the tiles.

The second subquestion is *What hyperparameters of the anomaly detection model have an impact on the performance of the model?*. Section 4.2 determined the first parameter, which is the CutPaste variant to use where our own CutPaste-multiple scores best. This probably has to do with the anomaly distribution in the dataset. The AeroScapes dataset contains multiple anomalies per image, whereas the original paper focuses on small defects. This is probably due to the fa Section 4.4 focuses on different learning rates, optimizer and schedulers. A learning rate of 0.0001 performs the best with the SGD optimizer. The Cosine Annealing Warm Restarts got the best results if the restarts were set at 32 epochs. The last Section 4.4 focuses on the parameters of the CutPaste-multiple. These are occurrence, the number of patches, and patch size. The model scores the highest

AUC using the number of patches and the patch size distributions. The possibility of a relationship between the two distributions is not explored in this research and could be a good next step.

The last subquestion, *What is the performance of the anomaly detection model when data augmentation is performed?*, is answered in Section 4.5. Six different augmentations methods have been tested, where random sun flare improved the model's performance significantly. However, further research on the augmentations is needed to see if the model could be further improved.

5.3 Future Work

This project aims to inspire future work, some suggestions for further work will be made. First of all, there are no tests completed to test the real-time predictions of the model. However, deployment on a UAV and monitoring the speed and accuracy of the system is necessary to determine the system performance in an actual scenario.

A second suggestion would be to expand the dataset. AeroScapes contains varied data, but the model performance could be improved if more data is available.

Testing different anomaly detection frameworks could be insightful. This research focused on the CutPaste framework and optimizing this with a new alteration method. However, it would be good to look at different architectures and compare these results.

Something else that still needs work is the distribution between the number of patches and the patch size for the CutPaste-multiple framework. The connection between the two is not explored and is considered two independent distributions, with an underlying relationship between them.

In preprocessing the dataset, each tile gets assigned the class normal or anomaly based on the number of unsafe pixels in the tile. In this research, the tile gets the class anomaly if one pixel is unsafe. It is worth to note that, to classify a tile as an anomaly with only a few unsafe pixels might be is in fact a drastic choice. Hence, in the future, it might be interesting to determine the performance if a higher threshold is defined.

For the PaDiM heatmaps, the highest pixel value is chosen among other possible options, such as the average when stitching back the tiles. This choice is to account for any possible anomaly even if it is only recognized in one of the overlapped tiles, as it is better to have a false positive (accounting for any possible unsafe region) than a false negatives in this research.

Another suggestion is to use the F_β metric instead of the F_1 metric, where β is chosen such that recall is considered β times more important than precision. In this project, recall is more important than precision.

Acknowledgements

We want to thank the CVDS Professorship and, in particular, Willem Dijkstra for providing code to run the experiments.

This project is financially supported by the RAAK MKB The Beast program.

References

[1] SESAR Joint Undertaking. Supporting safe and secure drone operations in europe. 2020.

[2] SESAR Joint Undertaking. European drones outlook study, unlocking the value of europe. 2016.

[3] Commission implementing regulation (eu) 2019/947 of 24 may 2019 on the rules and procedures for the operation of unmanned aircraft. *OJ L 152, 11.6.2019, p. 45–71*.

[4] Yu Tian, Guansong Pang, Fengbei Liu, Yuanhong Chen, Seon-Ho Shin, Johan W. Verjans, Rajvinder Singh, and Gustavo Carneiro. Constrained contrastive distribution learning for unsupervised anomaly detection and localisation in medical images. *CoRR*, abs/2103.03423, 2021.

[5] Ilker Bozcan and Erdal Kayacan. Uav-adnet: Unsupervised anomaly detection using deep neural networks for aerial surveillance, 2020.

[6] Chun-Liang Li, Kihyuk Sohn, Jinsung Yoon, and Tomas Pfister. Cutpaste: Self-supervised learning for anomaly detection and localization. *CoRR*, abs/2104.04015, 2021.

[7] The SOARIZON Team. What are vlos, evlos and bvlos? why do they affect drone operations? *Soarizon*, Sep 2020.

[8] Sebastien Bosch, Simon Lacroix, and Fernando Caballero. Autonomous detection of safe landing areas for an uav from monocular images. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5522–5527, 2006.

[9] C Patruno, M Nitti, A Petitti, E Stella, and T D'Orazio. A vision-based approach for unmanned aerial vehicle landing. *Journal of Intelligent Robotic Systems*, 2018.

[10] Sven Lange, Niko Sünderhauf, and Peter Protzel. Autonomous landing for a multirotor uav using vision. In *In SIMPAR 2008 Intl. Conf. on Simulation, Modeling and Programming for Autonomous Robots*, pages 482–491, 2008.

[11] Youeyun Jung, Hyochoong Bang, and Dongjin Lee. Robust marker tracking algorithm for precise uav vision-based autonomous landing. In *2015 15th International Conference on Control, Automation and Systems (ICCAS)*, pages 443–446, 2015.

[12] Zhenyu Yu, Kenzo Nonami, Jinok Shin, and Demian Celestino. 3d vision based landing control of a small scale autonomous helicopter. *International Journal of Advanced Robotic Systems*, 4(1):7, 2007.

[13] Jonathan Masci, Ueli Meier, Dan Cireşan, and Jürgen Schmidhuber. Stacked convolutional auto-encoders for hierarchical feature extraction. In Timo Honkela, Włodzisław Duch, Mark Girolami, and Samuel Kaski, editors, *Artificial Neural Networks and Machine Learning – ICANN 2011*, pages 52–59, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[14] Philipp Liznerski, Lukas Ruff, Robert A. Vandermeulen, Billy Joe Franks, Marius Kloft, and Klaus-Robert Müller. Explainable deep one-class classification. *CoRR*, abs/2007.01760, 2020.

[15] Tal Reiss, Niv Cohen, Liron Bergman, and Yedid Hoshen. PANDA - adapting pretrained features for anomaly detection. *CoRR*, abs/2010.05903, 2020.

[16] Thomas Defard, Aleksandr Setkov, Angelique Loesch, and Romaric Audigier. Padim: a patch distribution modeling framework for anomaly detection and localization. *CoRR*, abs/2011.08785, 2020.

[17] Ishan Nigam, Chen Huang, and Deva Ramanan. Ensemble knowledge transfer for semantic segmentation. In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1499–1508, 2018.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[19] Oliver Rippel, Patrick Mertens, and Dorit Merhof. Modeling the distribution of normal data in pre-trained deep features for anomaly detection. *CoRR*, abs/2005.14140, 2020.

[20] Karl Pearson F.R.S. Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 2(11):559–572, 1901.

[21] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, and Alexandr A. Kalinin. Albumentations: Fast and flexible image augmentations. *Information*, 11(2), 2020.

[22] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[23] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

[24] Ilya Loshchilov and Frank Hutter. SGDR: stochastic gradient descent with restarts. *CoRR*, abs/1608.03983, 2016.

A HARDWARE

Table 13 shows the hardware specifications of the computer which it used to run the experiments throughout the project.

Table 13: Hardware specifications

Hardware	Virtual Machine hosted at NHL Stenden
CPU	16 Cores @ 2.2 GHz
RAM	30 GB
GPU model	Tesla P100-SXM2
GPU memory	16 GB
CUDA version	11.0

B EXPERIMENT: NUMBER OF PATCHES AND PATCH SIZE

For the CutPaste-multiple alteration, the number of patches and the patch size parameters should be optimized. The experiments are divided into different experiment groups where they all focus on different aspects. Table 15 shows the results from the experiments of the number of patches and the patch sizes. Run A2 corresponds to run 13 from Section 4.3. Experiments in group A focus on different patch sizes, here the number of patches is held constant at 2 to 5. Experiments in group B encounter the same patch sizes as in group A, but the maximum number of patches is increased to 10.

Out of groups A and B, the best performing patch size is chosen, this is 0.1-0.5%. This patch size is used in the experiments in C. All of the models with a maximum of 10 patches perform better than the models with a maximum of 5 patches, respectively. Experiment C increases the minimum number of patches to see if the model's performance improves.

Experiment B2 and the experiments from group C are compared to each other. A minimum number of patches of 2 performs the best out of them, even if the results are similar.

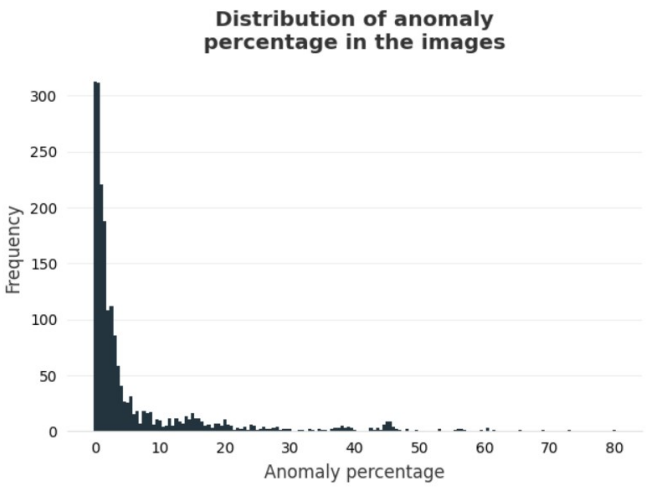


Fig. 9: Frequency of the anomaly percentage in the images

Experiment D has a different goal. Here the percentage augmented on the whole tile is tried to keep constant on 0.5% to 20%. The number of patches and corresponding patch sizes are changed. This number is based on the distribution of the anomalies in the dataset. Figure 9 shows the frequency of the percentage that is an anomaly in an image. This shows that most of the images have less than 20% anomalies in them.

Experiment D1 outperforms D2 and D3, which sets the patch size again on 0.1% to 0.5%. Because the frequency of the percentage of the anomalies is still present at around 50%, experiments E1 to E3 increase the maximum of the image augmented. The four experiments all have quite similar results, all having a F1-score of around 0.77. Experiment E1 outperforms a little bit, so this model is chosen to experiment further with.

C REPEATING BEST MODEL

The best model from Section 4.5 is repeated multiple times to determine the variance in the performance. The results are found in Table 14. The mean and standard error are shown in the last row.

Table 14: Metrics from the patch distributions experiment

Run	F1	Recall	Precision	AUC
1	0.8036	0.7328	0.8895	0.8876
2	0.8034	0.7325	0.8896	0.8879
3	0.8054	0.7389	0.8849	0.8858
4	0.8055	0.7355	0.8902	0.8891
Mean (st. error)	0.8045 (0.0006)	0.7349 (0.0015)	0.8886 (0.0013)	0.8876 (0.0007)

Table 15: Metrics from the experiments on the number of patches and patch sizes

Exp. group	Number of patches	Patch size	Percentage augmented	F1-score	Recall	Precision	AUC
A1	2-5	0.1-0.3%	0.2-1.5%	0.7619	0.6830	0.8614	0.8557
-A2-	2-5	0.1-0.5%	0.2-2.5%	0.7634	0.6834	0.8634	0.8535
A3	2-5	0.1-0.7%	0.2-3.5%	0.7542	0.6736	0.8567	0.8461
A4	2-5	0.1-1.0%	0.2-5.0%	0.7582	0.6785	0.8591	0.8488
B1	2-10	0.1-0.3%	0.2-3.0%	0.7623	0.6815	0.8649	0.8564
B2	2-10	0.1-0.5%	0.2-5.0%	0.7674	0.6860	**0.8707**	0.8606
B3	2-10	0.1-0.7%	0.2-7.0%	0.7623	0.6830	0.8624	0.8523
B4	2-10	0.1-1.0%	0.2-1.0%	0.7688	0.6919	0.8648	0.8571
C1	4-10	0.1-0.5%	0.4-5.0%	0.7658	0.6895	0.8610	0.8534
C2	6-10	0.1-0.5%	0.6-5.0%	0.7664	0.6877	0.8654	0.8569
C3	8-10	0.1-0.5%	0.8-5.0%	0.7664	0.6944	0.8550	0.8469
D1	5-40	0.1-0.5%	0.5-20.0%	0.7764	0.7045	0.8647	0.8591
D2	5-20	0.1-1.0%	0.5-20.0%	0.7678	0.6938	0.8594	0.8546
D3	5-10	0.1-2.0%	0.5-20.0%	0.7555	0.6811	0.8480	0.8396
E1	5-60	0.1-0.5%	0.5-30.0%	**0.7788**	**0.7081**	0.8651	**0.8608**
E2	5-80	0.1-0.5%	0.5-40.0%	0.7779	0.7074	0.8641	0.8595
E3	5-100	0.1-0.5%	0.5-50.0%	0.7743	0.7052	0.8586	0.8564