A user-centric approach to dataset augmentation for anomaly detection using Unity

Alexander Rosbak-Mortensen
Master thesis Project
Aalborg, Denmark
arosba18@student.aau.dk

Marco Jansen
Master thesis Project
Aalborg, Denmark
mjanse18@student.aau.dk

Morten Muhlig
Master thesis Project
Aalborg, Denmark
mmuhli18@student.aau.dk

Mikkel Bjørndahl Kristensen Tøt
Master thesis Project
Aalborg, Denmark
mtat18@student.aau.dk

ABSTRACT

Anomaly detection plays a critical role in surveillance systems, particularly in the automation of large-scale monitoring scenarios. Anomaly detection algorithms require datasets comprised of large amounts of annotated data to train and evaluate models. Gathering and annotating this data is a labour intensive task, that can be costly if outsourced to external partners. A method to get large amounts of annotated data explored by current state-of-the-art research is to generate it artificially using 3D applications, which has the advantage of being able to generate new frames in quick succession. This poses new issues for the end-user by being a specialized field, which means low-expertise users rely on external partnerships to have this option. In this paper we propose an application that synthesizes datasets using 3D models and simulates anomalies on real backgrounds using the Unity Engine. Additionally, we introduce a high-usability User Interface attached to a highly customizable simulation that simplifies the process of generating synthetic data without the need for specialized expertise in 3D animation. Testing datasets augmented with synthetic data made using our application gave promising results, with increases in both AUC and F1 scores in all cases. This indicates that synthetic data generation for low-expertise end users is a viable approach, and we recommend future works to focus on creating high variation in their data and to use photorealistic 3D models and lighting.

Author Keywords

Deep learning; anomaly detection; autoencoder model; dataset generation; RITE

INTRODUCTION

In a scene monitored by a camera, abnormal activities, also called anomalies, are activities that are out of the ordinary. Some anomalies are scene dependent, i.e. what might be considered an anomaly in one scene might be considered normal in another. For example, a traffic camera observing an intersection may sometimes need to detect illegal U-turns, while this is not considered illegal elsewhere, and therefore would not be an anomaly. Anomalies that are scene in-dependant also exist, such as brawling or fighting, which would be considered anomalies in most contexts. [35, 19, 6, 5]

Within automation of surveillance systems, anomaly detection is a keen area of interest as it is beneficial for a system to filter away normal behaviour and alert of abnormal behaviour. The advantage of this is cutting down the manual labour of monitoring large scale surveillance systems, such as public monitoring in parks, harbours, streets, or intersections. This can be especially beneficial in areas where quick detection of anomalies is necessary, such as roads and intersections, where traffic accidents can require quick responses to potentially save the people involved. Common use-cases for anomaly detection within traffic surveillance includes monitoring traffic flow and congestion, accidents, and law violations. [25, 26, 24, 6]

Detecting anomalies is done using machine learning models, often using autoencoder models that learn what the normal input in their training set looks like. They then try to recreate new frames from their testing set which includes both normal and anomaly frames, and the optimal outcome is that they are good at recreating normal frames, and poor at recreating anomaly frames as they were not trained to do so. This divide can be detected and used to classify the data. This has the advantage over classical methods that the models do not need to be trained on what every potential anomaly looks like, as they are not made with the purpose of labeling specific anomalies. [36, 15, 17]

Training these models requires large amounts of normal data, contained in datasets, while testing them requires footage of annotated anomalous behaviour which can be difficult to acquire. Datasets which can be downloaded and used already exist, but the number of situations in which anomalies can be present in reality is almost infinite, and datasets do not exist for each type of situation. If a user wants to train such a models to detect anomalies on e.g. a local road outside of their house, they would need to create the dataset themselves, as existing datasets would not accomplish this.

Getting large amounts of annotated real anomaly data can be a difficult task, as manually annotating large amounts of data is a labour intensive task. Due to this, attempts have been made to use synthetically generated data instead, which in some cases gave better results than using real data. Methods here vary from using a video game like GTA to teach a machine learning model to read traffic, i.e. where and how far away other cars and lane markings are, when driving [28, 18, 32], to using 3D artists to make synthetic anomalous behaviour with 3D models in a scene. [25, 3]

Current methods for generating synthetic data relies on time consuming processes, in some cases spending multiple weeks on frame rendering, and dependence on 3D-artists for data generation, often involving external partners. This leads to less flexibility in the data generation process for the end users. Alternatives to doing manual annotation, or hiring external partners e.g. 3D-artists and the expenses that follow, is currently an active research area. [3, 13, 11]

In this paper we present a method to synthesize datasets using 3D models, simulating traffic anomalies on real backgrounds, while lowering the dependency on 3D artists. We also present a User Interface, specialized in high usability through Rapid Iterative Testing and Evaluation (RITE), that simplifies the process of generating synthetic data, without relying on external companies or extensive knowledge about 3D animation. This method innovates on current state-of-the-art in multiple ways:

- Automated generation of normal and anomalous data for use in datasets.
- A highly customizable simulation with a visual user interface specialized in high usability.
- An application that allows users with low expertise to create state-of-the-art synthetic data that can be used to improve their datasets on multiple performance measures.

BACKGROUND RESEARCH

Anomaly detection

Machine learning for surveillance automation can be split into two groups; those that require human feature extraction and those that have automatic feature extraction. Classical models utilize human feature extraction, while deep learning models utilize automatic feature extraction. Deep learning models use layered algorithms to create an Artificial Neural Network, and are designed to make decisions in a similar fashion to how a human would. This allows them to do complex tasks like optimal feature extraction without human guidance, and they outperform classical methods on classical vision techniques regarding image and video data. [38, 31, 4]

A common pipeline for anomaly detection using deep learning can be seen in figure 1, where the user is responsible for data preprocessing, but the algorithm takes care of feature extraction during training. [3, 2]

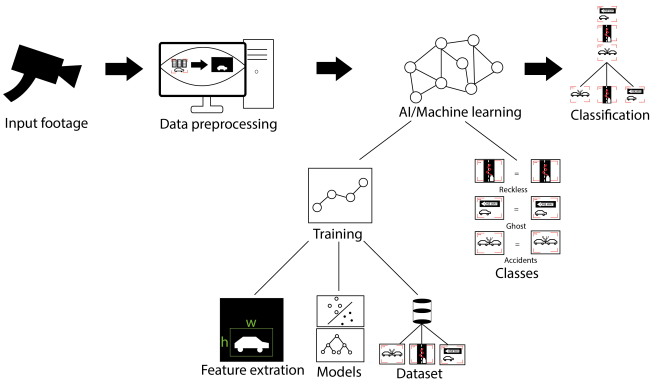


Figure 1. Illustration of a common pipeline for classification using deep learning, which can be used for anomaly detection.

Current deep learning models for anomaly detection often use a system where they do not specifically detect anomalies. Rather, they are given data of how the scene normally looks, and can detect to which extent the scene currently looks normal. This is typically done using either a Prediction or a Reconstruction based method, with prediction methods often having higher accuracies in state-of-the-art experiments [33, 39], and being simpler to train. Detecting anomalies is based on classifying frames that deviate heavily from the learned data as anomalies. [36, 15]

Environment-dependent Anomalies

There are many different types of anomalies. Some anomalies require certain factors in the environment, such as jaywalking being an anomaly that requires a road, while other anomalies can be present anywhere. Furthermore, anomalies can have different frequencies in different places. For instance, common anomalies on campuses include riding a bike, walking on grass, and driving a golf-cart [24], while common anomalies for traffic surveillance include different anomalies, like jaywalking, cyclist out of lane, and illegal U-turns [2, 35].

This means that creating an anomaly detection algorithm also becomes environment-dependent, meaning it can be beneficial, or necessary depending on the model used, to only include data from specific locations in a dataset. The drawback is that the algorithm becomes hyper specific for that location as any frame from a different location will be impossible for the model to predict or reconstruct.

Synthetic Anomalies

Acsintoae et al [3] presents a way to artificially generate data for abnormal behaviour. They take an image from real footage, and remove the foreground elements, such as pedestrians and cars, so they do not appear in every artificially generated frame. 3D-artists then add 3D models onto the scene, the 3D models then perform different scenarios like fighting, seizures, or accidents. They run experiments on existing datasets and add the synthetic data to it, getting increased AUC scores (Lowest increase is 89.3% -> 90.5%, Largest increase is 58.5% -> 68.2%) compared to the non-synthetic counterparts. The advantages of this method is that the research team gains more control over the anomalies in the scene, and since they control the ground truth, automatic annotation of the anomalies becomes trivial. However, it also requires hiring external partners e.g. 3D artists, which can be costly and the logistics time-consuming, due to issues such as miscommunication, rendering time, or multiple iterations for the same footage.

Making synthetic data which matches real footage requires careful placement of 3D objects, as they have to match both the angle and location of how they would appear were they real objects. This is commonly done by hand, but is a time-consuming task [3]. Alternatives exist that accomplish this automatically, such as fSpy, an open source camera matching application [1]. This program is used by state-of-the-art research to match the perspective of virtual cameras to real cameras based on manual placement of vanishing points. [16]

Diaz Da Cruz et al [11] present their paper with a new Synthetic dataset for Vehicle Interior Rear Seat Occupancy (SVIRO) that can be used for detection and classification. Their dataset contains vehicle interiors from ten different cars, the rear sets would be randomly occupied by 3D models of people of different sizes and shapes, children, baby seats, children seats, and bags. Their fully synthetic dataset achieved similar results to a similar dataset comprised of real footage, which is a good result for a fully synthetic dataset. They mention as future works that they believe their good results were partly due to high variation in their 3D models, but that the results could have been even better if their variation was higher, and their models were more photo-realistic.

Existing datasets

Current state-of-the-art research on anomaly detection commonly uses existing datasets to train and evaluate their models [33, 37, 9]. Common datasets include UCSD Ped2 [21], CUHK Avenue [23], ShanghaiTech [22], etc. This is done because creating a new dataset from scratch is a time-consuming task, and because using the same datasets allows for accuracy comparisons between similar research. An issue that arises periodically with existing datasets is that new models become too efficient at classifying the dataset, which makes the dataset 'solved'. Therefore, newer challenging datasets are also often used. These include Street Scene [35], UHTCD [27], etc.

METHOD

Application for synthetic dataset generation

In this paper we propose an application for synthetic dataset generation based on a user-centric design perspective which can be seen in figure 2. Heuristics proposed by Desurvire et. al. [10] were used to ensure high usability for the application, which maximizes the effectiveness, efficiency and satisfaction, by lowering interruptions or challenges in using the program [20]. As the background research highlighted that anomalies are environment-dependent, the proof-of-concept implementation is made for specifically one type of environment. The "Street Scene" dataset was chosen as the dataset used during evaluation, so the proof-of-concept implementation includes objects and anomalies relevant to a traffic environment.

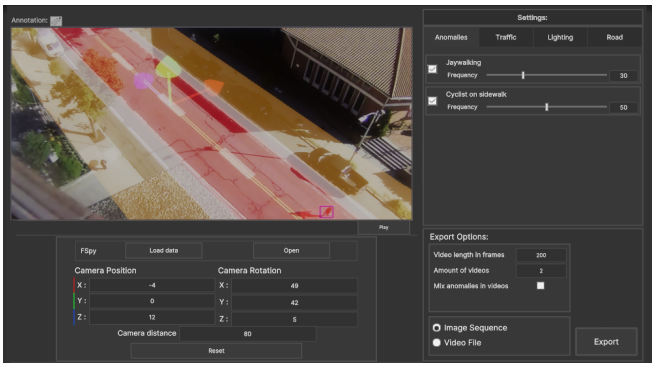
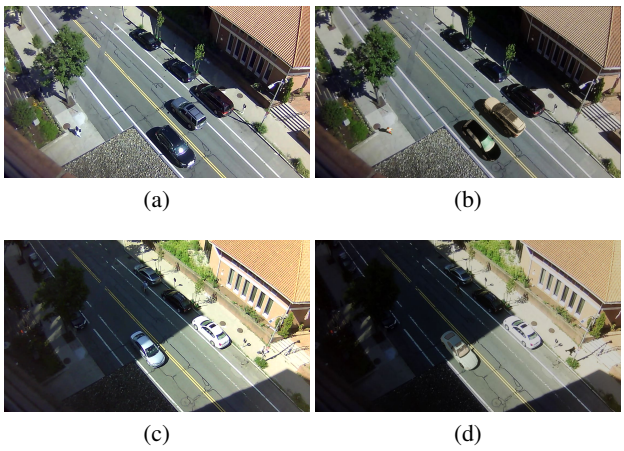


Figure 2. The proposed application for the synthetic data generation

The images created by the application are designed to match images from the chosen dataset by using realistic 3D models and scene-matching lighting setups and foreground shadows. This can be seen in figure 3, where foreground elements such as moving cars, cyclists, and pedestrians are simulated.



(a) (b)

(c) (d)

Figure 3. Two comparisons between the street scene dataset (a) and (c), with recreations done using the application (b) and (d).

The use-case of the application can be seen in figure 4. The user can input footage into the application and rotate the objects in the scene until they match the real footage.

Potential foreground elements such as trees can be marked so they appear on top of the synthetic data. Multiple tabs of settings for adjusting e.g. which objects the user wants in the scene, and how frequently they should appear. The lights and shadows are adjustable, as are the export settings such as how many frames of data the user desires.

As the end-user is involved in the main use-case, and the end-user is not expected to be an expert, a high usability is required for the end-user to achieve good results with the application.

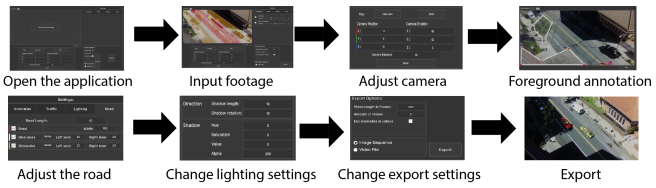


Figure 4. Use of the proposed application for synthetic data generation

Features

To make the use-case possible, a set of features are implemented based on background research, RITE testing, and expert feedback:

- Upload real footage to be used as the background for the synthetic data. The background research highlighted that adding synthetic data on top of real data can improve the results of dataset generation.
- fSpy incorporation where the users can open their footage in fSpy through the application, and can place vanishing points and generate camera angle data that is then loaded by the application to ensure the Unity Camera's angle matches the angle used by the real camera. This is important as the background research highlighted that synthetic data produces better results in this context when the generated footage matches real footage.
- The user can further adjust the camera if the angle and location loaded by fSpy do not perfectly match the real footage.
- Annotation of foreground elements such as trees/poles, which will be rendered on top of the generated 3D objects, which ensures that the synthetic data does not appear in front of objects in the scene that they should not, which results in higher photo realism.
- Selection of anomalies to generate, along with how frequently they should appear. The background research highlighted that anomalies can be environment-dependent, which means the user should be able to customize the selection of anomalies to generate. An example of a simulated jaywalker anomaly can be seen in figure 5.
- Adjusting the traffic flow to match their real footage, which ensures that the synthetic data matches the real footage.
- Adjusting whether environmental factors such as bike lanes and sidewalks are present in the scene, and how wide they are if present.
- A preview of the simulation before exporting the data that allows the user to preview what their changes do. Heuristic Category 3 C1 and C2 notes that immediate visual feedback is important for the user's experience, and feedback from the RITE testing support this claim as it was a commonly requested feature.
- Adjusting the lighting and shadow settings to make the footage match the real footage.
- Adjust how the user wants the data exported, such as number of frames, frame skips, format, etc., which can lower the amount of necessary preprocessing. The data is outputted sequentially with context intact, which was suggested during an expert feedback session.
- Tool-tips for every feature in the application, which explains what it does. This was added due to multiple cases of RITE and Expert feedback, which signals that tool-tips are something users particularly desire.

Figure 5. A synthetic jaywalker anomaly created by the application finds himself in a precarious situation.

Increasing usability

A goal with the design of the interface was to keep the layout simple, while still giving the user the freedom to make the simulation fit their footage. This can make a challenge as giving the user more freedom inherently will lead to a need for more input fields. [8, 12]

The design was influenced by applications for similar areas such as 3D programs like Blender and Unity, and video editing programs like Hitfilm and After Effects. Heurestics Category 3 B1 and B3 claim that using industry standard visuals and controls increase usability as the user has a higher chance to be familiar with how to use certain features of the application. These programs use systems such as tabs and window/box designs to split input fields and information by functionality. This helps keep controls relevant to a feature onscreen, while controls not used for this feature are hidden while not in use. The foreground annotation feature was inspired by annotation software such as Madtagger and labelme.

EXPERIMENTS

RITE testing procedure

Rapid Iterative Testing and Evaluation (RITE) was used to test for usability issues in the application, which helps ensure a

high final usability. RITE measures Effectiveness, Efficiency, and Satisfaction through user feedback. [14]
RITE is performed with a smaller sample size, often 3-5 participants per iteration, but the sample size can be as low as one participant. The sample size of each iteration can vary between iterations, depending on the amount and severity of the issues or bugs found. According to research on sample size for traditional usability testing, 4-5 participants will uncover 80% of the problems which have a high likelihood of being detected. [29]

Procedure
For our RITE testing the participants were found at Aalborg University. Each iteration consisted of one to three participants. They performed 10 tasks that guided them though the core loop of the application. The test was concluded after a group of five participants could no longer find issues or bugs that were not considered edge cases.

The RITE testing was conducted by two group members, one whose primary tasks was to conduct the test (e.g. Introduction and keeping the participants engaged in thinking out loud). The other member then observed and noted down the feedback from the user. The screen was recorded during the test for later review and evaluation.

Machine learning procedure
To train and evaluate the dataset, the Memory-Guided Normality for Anomaly Detection (MNAD) algorithm was used [34], using the prediction method on 256x256 resolution input images, with a frame sequence length of 5 and a threshold for anomaly scores of 0.6, doing 60 epochs for training. The implementation used was the official implementation of the algorithm presented in the paper, created by the authors Park et. al. [33], with slight modifications to make the data loading work on the Windows operating system but the modifications did not otherwise impact the function of the algorithm. The MNAD evaluation outputs Area Under Curve (AUC) scores, which are used to determine the classification effectiveness of the models produced by the algorithm.
To test the effect of adding synthetic data to the training-sets, multiple models were trained on different variants of the dataset. Two of which only contained images from the original street scene dataset in different quantities, and two of which added synthetic data generated by the application presented in this paper onto those datasets. The reason different quantities of real data were used was to test the difference in the effect of adding synthetic frames, if a dataset already had a sizable amount of data, versus if a dataset had a small amount of data, and to test if the ratio of real-to-synthetic data would result in different AUC score changes. The models, and information about their datasets, can be seen in table 1.

Model:	RF:	SF:	TF:	RTS-Ratio:
SS	5668	0	5668	1 to 0
SS(synth)	5668	5607	11275	1 to 0.989
SSmini	600	0	600	1 to 0
SSmini(synth)	600	5607	6207	1 to 9.345

Table 1. A table showing the models used, Street Scene (SS), Street Scene w/ Synthetic Frames (SS(synth)), Street Scene mini (SSmini), and Street Scene Mini w/ Synthetic Frames (SSmini(synth)), with their number of real frames (RF), synthetic frames (SF), total frames (TF), and the ratio between real and synthetic frames (RTS-Ratio).

The dataset used for testing was the testing-set of the Street Scene dataset with scene 35 removed, since the labels were formatted differently than the other scenes. This testing-set contains 145,278 images, with 101,547 normal frames and 43,731 abnormal frames. This contains a large class-imbalance, which can negatively impact the validity of the AUC scores[30], so F1-scores are also calculated and compared to compensate.

EXPERT INTERVIEW
In addition to the RITE testing, an interview with two experts in the field of computer vision was conducted. This was as an open interview where they tried the application while giving feedback.

They overall gave positive feedback on the application and said that it had potential. They also gave suggestions for further features they believe would improve the applicability of the implementation.
Some of their suggested features were:

- Adding depth to foreground annotations, allowing some anomalies to be rendered in front of objects while others would still be obstructed.
- A more dynamic system for adding animations or movement for anomalies by adding a folder for a user to place custom behaviour splines for 3D models.
- The ability to change the order of bike lanes and sidewalks, so the sidewalk is next to the road.

RESULTS
RITE results
The RITE testing was done in 5 iterations. The final feedback only included niche things and personal preferences. As these were not considered to be vital for the design, it was decided to end the testing there.

Effectiveness
The figure 6 shows the amount of errors and failures each participant encountered during the five iterations.

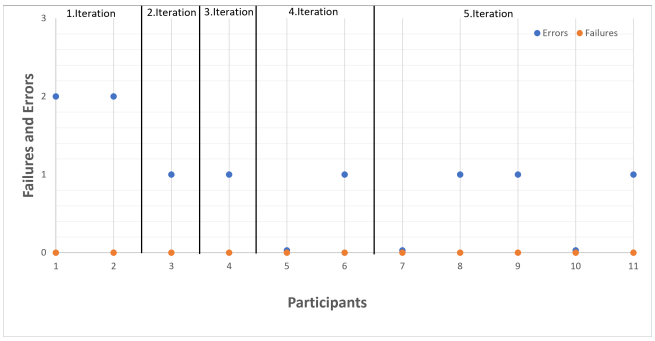


Figure 6. A diagram that shows the amount of errors and failures encountered during each of the iterations. The x axis shows the number assigned to each individual participant, and y axis shows the number of failures and errors.

During the first iteration four errors were encountered. In the second, third, and fourth iteration, one error was encountered in each iteration. In the fifth iteration three errors were encountered.

Efficiency

The two tasks the participants found the hardest were, "annotate two foreground elements" and "camera position and rotation". The score for "camera position and rotation" changed across the iterations from "neither difficult nor easy" to "easy", while the "annotate two foreground elements" task did not improve. The rest of the tasks were found to be between "easy" and "extremely easy".

Satisfaction

For the overall satisfaction score, there was one participant on the fifth iteration that gave the rating "Best imaginable", the rest of the participants gave the rating "Good".

Machine learning results

The results of the evaluation can be seen in tables 2 and 3. These tables show the calculated values for each model, along with relevant settings used during the calculation like the threshold of the labeling and the weighting coefficient of the F1 score. The True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN) counts are presented, and the Precision and Recall are calculated and then used to calculate the F1-score for each model.

Model:	SS	SS(synth)
Threshold:	0,6	0,6
AUC:	0,5064	0,5106
Guessed A, was A (TP)	35103	37273
Guessed N, was N (TN)	23612	20229
Guessed A, was N (FP)	77843	81226
Guessed N, was A (FN)	8584	6414
Precision	0,31	0,31
Recall	0,80	0,85
F1 score	0,4482	0,4596

Table 2. A table showing different statistics of the Street Scene model, and the version with synthetic data added. SS is Street Scene, SS(synth) is Street Scene w/ Synthetic Data. A and N stands for Anomaly and Normal data respectively.

Comparing table 2, Street Scene and Street Scene with Synthetic Data, we see that the number of True Positives went up by 2170, while the number of True Negatives went down by 3383. The AUC score increased from 0.5064 to 0.5106, and the F1-score increased from 0.4482 to 0.4596, because while the Precision was unchanged, the Recall increased from 0.80 to 0.85.

Model:	SSmini	SSmini(synth)
Threshold:	0,6	0,6
AUC:	0,4779	0,5242
Guessed A, was A (TP)	35615	37665
Guessed N, was N (TN)	22586	15925
Guessed A, was N (FP)	78869	85530
Guessed N, was A (FN)	8072	6022
Precision	0,31	0,31
Recall	0,82	0,86
F1 score	0,4503	0,4514

Table 3. A table showing different statistics of the Street Scene mini model, and the version with synthetic data added. SSmini is Street Scene mini, SSmini(synth) is Street Scene mini w/ Synthetic Data. A and N stands for Anomaly and Normal data respectively.

Comparing table 3, Street Scene mini and Street Scene mini with Synthetic Data, we see that the number of True Positives went up by 2050, while the number of True Negatives went down by 6661. The AUC score increased from 0.4779 to 0.5242, and the F1-score increased from 0.4503 to 0.4514, because while the precision was unchanged, the Recall increased by 0.04.

These results show that in both cases of adding synthetic data, the model showed an increase in ability to correctly classify anomaly frames, with a decrease in ability to correctly classify normal frames. Both AUC and F1-score measurements increased as a result of adding synthetic data in both cases.

DISCUSSION

Discussion of Machine Learning results

The results for the models are overall very poor, having AUC scores near 50%, which is about as good as random guesses. Looking deeper into the data shows that the models are good at classifying an anomaly frame correctly but are very poor at classifying a normal frame as being normal. The models therefore have a large bias for classifying frames as anomalies, with the models classifying about 80% of the dataset as anomaly frames, even though anomaly frames only accounted for 30.1% of all frames. Changing the threshold can adjust this ratio, but at the expense of the models' abilities to classify anomaly frames correctly. Depending on the use-case of the model, different values would be desirable, as some use-cases would be fine with high rates of False Positive's, while in other use-cases such as automatic dispension of traffic violations, that would be unacceptable.

This is likely due to the chosen dataset having location-based anomalies, which makes it very difficult to classify for state-of-the-art algorithm like MNAD. With accuracies for location-based anomalies done in current research being near the 60% range even with large datasets [7], and the Street Scene dataset itself being tested at the frame-level with near 50% accuracy using state-of-the-art models in the paper written by the authors of the Street Scene dataset [35].

However, the results showed an overall increase in AUC and F1-scores, meaning the effectiveness of the model in ability to correctly classify the data went up due to the introduction of synthetic data, which is a promising result. The models' abilities to classify abnormal data as abnormal went up, while their abilities to classify normal data as normal went down. This makes sense, as the synthetic data produced in this paper is not photo-realistic, meaning that while the dataset does gain more representation of what normal data can look like, it also receives a style of data that is not present in the testing set. The increase of the F1-scores is a result of increased recall, not increased precision. This highlights that the introduction of synthetic data did not have a large effect on a model's ability to be correct when guessing that a frame had an anomaly in it, but rather that the anomaly frames were more likely to be identified as anomaly frames by the model.

Based on the overall poor results of the models, it is difficult to conclude if the synthetic data made the models better, even if their performance resulted in increased AUC and F1-scores. It would be interesting to redo the experiment with either a model more suited for location-based anomalies, or with a dataset that does not have those types of anomalies and where current state-of-the-art models already achieve decently high AUC scores, to see if synthetic data can make already good results even better.

Discussion of Design

During the first design iteration of the application, before RITE testing began, it was presented to a group of PhD computer graphics researchers. They gave feedback to the design, and features that were intended to be part of it. The iterations that followed were not presented to them, and were designed based on internal design decisions. The feedback the researchers could have provided on the iterations would have ensured that the application would be more tailored towards expert preference and what they felt would be necessary in the application.

In the RITE testing, the task where errors were still encountered during the fifth iteration was the "annotation" task, despite the changes made during the different iterations. The participants noted that they had previous experience with editing software, but not annotation tools such as Madtagger. This lack of experience with such software, might have caused the difficulties they experienced navigating the tool.

The two tasks the participants found most difficult were "annotation" and "camera position and rotation". The task for adjusting the camera did improve in ratings with the addition of further labels and tool tips to guide the user. The annotation tool despite being given further feedback, tool tips, and interactions did not improve in ratings. This does not necessarily mean that the tool itself did not improve however, as part of what may have caused this is the task given for the tool being confusing. The task "annotate two foreground elements" was questioned by multiple participants with confusion about what makes a 'foreground element'. It is seen throughout the tests that different interpretations of this task and of 'a foreground element' were made, with some participants marking cars and others marking the roofs and trees. It is therefore possible that participants gave a lower rating based on having to interpret this 'foreground element' aspect.

The participants that found the task "annotate two foreground elements" difficult, did comment that while it was difficult in the beginning, it got easier after using it for a minute or two. A way to make the annotation features easier for the participants would be to add a tutorial, explaining the entirety of the application to the users, which was suggested during the expert feedback.

The overall satisfaction score staying at the rating of "good" throughout the iterations could indicate that changes did little to change satisfaction. With RITE naturally leading to smaller sample sizes between iterations, and our initial design already receiving a score of "good", it does become difficult to draw a proper trend of improvement. With a participant giving the score "Best imaginable" for the fifth iteration, it could be possible that satisfaction was improving, however one score is not enough to conclude this. Overall, participants did give consistent "good" ratings, which indicates an overall acceptable satisfaction score.

Some potential features were excluded from the application due to being deemed not vital. These features are considered potential future work. This included features such as a save/load system, tutorials, multiple roads, cross-walks, etc. These features could be implemented in future works to further expand the capabilities of the application, and would

be necessary if the product was released as a fully developed general-purpose application.

Discussion of RITE

One of the key differences between the RITE and traditional usability testing is the number of participants per iteration. RITE is based on the principle that multiple iterations with fewer participants per iteration, increase the efficiency of discovering new usability problems. The advantages of this is that it shortens the time between testing and fixing problems. However, a weakness of this method is that it primarily discovers problems that have a high likelihood of detection. Meaning that even after the conclusion of the RITE testing there might still be some usability problems undiscovered. A solution to this problem could be an increase in the numbers of participants per iteration. However, increasing the numbers of participants also lengthens the time between testing and fixing problems, which ultimately moves away from the point of being Rapid Testing.

It was observed that the participants interpreted some of the tasks differently, which makes comparison difficult. For instance, when asked to change certain settings, some participants changed one thing and deemed the task complete, while others fiddled with the settings until they were satisfied with the result. This makes comparing the time taken between these participants meaningless, as they were not functionally doing the same task. This was a problem of the tasks not being concrete enough, as in the example above no direct limit on how many times the settings should be changed was specified to the participants.

When analysing the data from the RITE, it was found that some of the wording in the tasks was biased towards the application. E.g. The task "Play simulation", was completed by pressing the button "Play". The bias of the wording could have affected the ratings and should be avoided if the experiment is redone. This could have been fixed by changing the task "Play simulation" to "Start simulation". Another solution would be to give the users a scenario instead of tasks. E.g. a scenario such as: "You are in a research team, you would like to use this application to generate data for your machine learning algorithm. The data would need to contain, marked foreground elements, and anomalies" and observe how the participants would complete the scenario. This approach would however be even more open to interpretation, and the participants might not test the entire core loop of the application. If this approach is taken, more consideration would need to be done to ensure the participants test all of the desired elements. An issue that was encountered when analysing the RITE data was the lack of audio recordings. The screen recording that was taken as the participants went through the tasks lacked context. The notes that were taken at the same time did not always specify where the participants were in the application when giving the feedback.

Discussion of Expert Interview

The experts provided a list of features that could be beneficial for the application. A few smaller features, for the usability, were implemented. However the majority of the features were deemed to not be vital for the application. The reason for this was due to the application presented in this paper being a proof-of-concept, so extensive changes were deemed out of scope.

One feature from the interview that was partly implemented was larger variation in the application. There was an increase in the variance of the 3D models in the application. It was increased from one pedestrian model and one car model to four pedestrian models and four car models. It can be argued that an even larger variation could produce even better results, which is supported by Diaz Da Cruz et. al. [11]. The experts also requested the features to add anomalies and 3D models themselves. This would solve the problem of low variance in the application.

Simulation limitations

The data generated by the application suffers from issues that could have lowered the quality of the synthetic data. Ones caught during testing include jaywalker anomalies clipping into cars seen on figure 7, simulated 3D-objects appearing on top of unmarked foreground objects seen on figure 8, and improperly masked foreground objects resulting in poorly hidden objects seen on figure 9, which is an especially prevalent issue as the current system does not support gradient masks.

Figure 7. A jaywalker clipping into a car instead of avoiding it. This cannot happen in reality and is therefore considered poor quality data.

Figure 8. A pedestrian appearing over a foreground object, as the object was not marked as one.

Figure 9. A car being hidden behind a foreground object with a poorly made mask using the user interface's foreground annotation tool.

CONCLUSION

This paper concludes that adding synthetic data made using a high-usability User Interface has the potential to increase the AUC and F1 scores when evaluated with the MNAD algorithm, which indicates that satisfyingly realistic data can be created by non-experts and used to improve their datasets.

The two datasets augmented with synthetic data achieved 0.83% and 9.68% respective increases in AUC score, with 2.45% and 0.24% respective increases in F1 score. However, the four models had generally poor results around 50%, so we recommend for similar experiments to use a dataset/model combo that achieves a better base result, to see if good results can be made great.

A heuristics-based user-centric design approach utilizing RITE testing and expert interviews resulted in an application with high usability scores, as many usability issues were found during the rapid iterative testing. This results in an application usable by users with low levels of expertise, which is ideal for an application that could be produced into a commercial product.

REFERENCES

[1] 2019. fSpy. (2019). https://fspy.io/

[2] Sabrina Aberkane and Mohamed Elarbi. 2019. Deep Reinforcement Learning for Real-world Anomaly Detection in Surveillance Videos. In *Proceedings - 2019 6th International Conference on Image and Signal Processing and their Applications, ISPA 2019*. Institute of Electrical and Electronics Engineers Inc. DOI: http://dx.doi.org/10.1109/ISPA48434.2019.8966795

[3] Andra Acsintoae, Andrei Florescu, Mariana-Iuliana Georgescu, Tudor Mare, Paul Sumedrea, Radu Tudor Ionescu, Fahad Shahbaz Khan, and Mubarak Shah. 2022. Ubnormal: New benchmark for supervised open-set video anomaly detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 20143–20153. DOI: http://dx.doi.org/https://doi.org/10.48550/arXiv.2111.08644

[4] Ioannis D Apostolopoulos and Mpesiana A Tzani. 2022. Industrial object and defect recognition utilizing multilevel feature extraction from industrial scenes with Deep Learning approach. *Journal of Ambient Intelligence and Humanized Computing* (2022). DOI: http://dx.doi.org/10.1007/s12652-021-03688-7

[5] A. Balasundaram and C. Chellappan. 2020. An intelligent video analytics model for abnormal event detection in online surveillance video. *Journal of Real-Time Image Processing* 17, 4 (aug 2020), 915–930. DOI:http://dx.doi.org/10.1007/s11554-018-0840-6

[6] Nyan Bo Bo, Maarten Slembrouck, Peter Veelaert, and Wilfried Philips. 2020. Distributed Multi-class Road User Tracking in Multi-camera Network For Smart Traffic Applications. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 12002 LNCS. Springer, 517–528. DOI: http://dx.doi.org/10.1007/978-3-030-40605-9_44

[7] Congqi Cao, Yue Lu, Peng Wang, and Yanning Zhang. 2023. A New Comprehensive Benchmark for Semi-Supervised Video Anomaly Detection and Anticipation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 20392–20401.

[8] Gong Chao. 2009. Human-Computer Interaction: Process and Principles of Human-Computer Interface Design. (March 2009), 230–233. DOI: http://dx.doi.org/10.1109/ICCAE.2009.23

[9] Manaswi Chebiyyam, Rohit Desam Reddy, Debi Prosad Dogra, Harish Bhaskar, and Lyudmila Mihaylova. 2018. Motion anomaly detection and trajectory analysis in visual surveillance. *Multimedia Tools and Applications* 77 (2018), 16223–16248. Issue 13. DOI: http://dx.doi.org/10.1007/s11042-017-5196-6

[10] Heather Desurvire and Charlotte Wiberg. 2009. Game Usability Heuristics (PLAY) for Evaluating and Designing Better Games: The Next Iteration. 557–566. DOI:http://dx.doi.org/10.1007/978-3-642-02774-1_60

[11] Steve Dias Da Cruz, Oliver Wasenmüller, Hans-Peter Beise, Thomas Stifter, and Didier Stricker. 2020. SVIRO: Synthetic Vehicle Interior Rear Seat Occupancy Dataset and Benchmark. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*. DOI: http://dx.doi.org/https://doi.org/10.48550/arXiv.2001.03483

[12] John J. Dudley and Per Ola Kristensson. 2018. A Review of User Interface Design for Interactive Machine Learning. *ACM Trans. Interact. Intell. Syst.* 8, 2, Article 8 (jun 2018), 37 pages. DOI: http://dx.doi.org/10.1145/3185517

[13] Alessandro Flaborea, Guido D'Amely, Stefano D'Arrigo, Marco Aurelio Sterpa, Alessio Sampieri, and Fabio Galasso. 2023. Contracting Skeletal Kinematic Embeddings for Anomaly Detection. (2023). DOI: http://dx.doi.org/https://doi.org/10.48550/arXiv.2301.09489

[14] GitLab. 2022. Rapid Iterative Testing and Evaluation (RITE) | GitLab. (2022). https://about.gitlab.com/handbook/product/ux/ux-research/rite/#a-sample-rite-study-approach

[15] Matheus Gutoski, Marcelo Romero Aquino, Manassés Ribeiro, André Lazzaretti, and Heitor Lopes. 2017. Detection of Video Anomalies Using Convolutional Autoencoders and One-Class Support Vector Machines. DOI:http://dx.doi.org/10.21528/CBIC2017-49

[16] Chris Hamill. 2021. The Atlas of Lost Rooms: Digitally Reconstructing Dark Heritage Sites in Ireland. In *Emerging Technologies and the Digital Transformation of Museums and Heritage Sites: First International Conference, RISE IMET 2021, Nicosia, Cyprus, June 2–4, 2021, Proceedings 1*. Springer, 199–216.

[17] Mahmudul Hasan, Jonghyun Choi, Jan Neumann, Amit K. Roy-Chowdhury, and Larry S. Davis. 2016. Learning Temporal Regularity in Video Sequences. (2016).

[18] Matthew Johnson-Roberson, Charles Barto, Rounak Mehta, Sharath Nittur Sridhar, Karl Rosaen, and Ram Vasudevan. 2017. Driving in the Matrix: Can Virtual Worlds Replace Human-Generated Annotations for Real World Tasks? (2017). DOI:http://dx.doi.org/https://doi.org/10.48550/arXiv.1610.01983

[19] Harpreet Kaur and Neeru Jindal. 2020. Image and Video Forensics: A Critical Survey. *Wireless Personal Communications* 112, 2 (may 2020), 1281–1302. DOI:http://dx.doi.org/10.1007/s11277-020-07102-x

[20] Sauli Laitinen. 2005. Better games through usability evaluation and testing. *Gamasutra.* (2005).

[21] Weixin Li, Vijay Mahadevan, and Nuno Vasconcelos. 2014. Anomaly Detection and Localization in Crowded Scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36, 1 (2014), 18–32. DOI:http://dx.doi.org/10.1109/TPAMI.2013.111

[22] W. Liu, D. Lian W. Luo, and S. Gao. 2018. Future Frame Prediction for Anomaly Detection – A New Baseline. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[23] Cewu Lu, Jianping Shi, and Jiaya Jia. 2013. Abnormal Event Detection at 150 FPS in Matlab.

[24] Pranav Mantini, Zhenggang Li, and K. Shishir Shah. 2021. A Day on Campus - An Anomaly Detection Dataset for Events in a Single Camera. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, Vol. 12627 LNCS. 619–635. DOI:http://dx.doi.org/10.1007/978-3-030-69544-6_37

[25] Pranav Mantini and Shishir K Shah. 2019a. Camera tampering detection using generative reference model and deep learned features. In *VISIGRAPP 2019 - Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, Vol. 5. 85–95. DOI:http://dx.doi.org/10.5220/0007392100850095

[26] Pranav Mantini and Shishir K. Shah. 2019b. UHCTD: A comprehensive dataset for camera tampering detection. In *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance, AVSS 2019*. DOI:http://dx.doi.org/10.1109/AVSS.2019.8909856

[27] Pranav Mantini and Shishir K. Shah. 2019c. UHCTD: A Comprehensive Dataset for Camera Tampering Detection. In *2019 16th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. 1–8. DOI:http://dx.doi.org/10.1109/AVSS.2019.8909856

[28] Mark Martinez, Chawin Sitawarin, Kevin Finch, Lennart Meincke, Alex Yablonski, and Alain Kornhauser. 2017. Beyond Grand Theft Auto V for Training, Testing and Enhancing Deep Learning in Self Driving Cars. (2017). DOI:http://dx.doi.org/https://doi.org/10.48550/arXiv.1712.01397

[29] Michael C Medlock, Dennis Wixon, Mark Terrano, Ramon Romero, and Bill Fulton. 2002. Using the RITE method to improve products: A definition and a case study. *Usability Professionals Association* 51 (2002), 1963813932–1562338474. https://www.jpattonassociates.com/wp-content/uploads/2015/04/rite_method.pdf

[30] Faezeh Movahedi, Rema Padman, and James F. Antaki. 2023. Limitations of receiver operating characteristic curve on imbalanced data: Assist device mortality risk scores. *The Journal of Thoracic and Cardiovascular Surgery* 165, 4 (apr 2023), 1433–1442.e2. DOI:http://dx.doi.org/10.1016/j.jtcvs.2021.07.041

[31] Ravil I Mukhamediev, Adilkhan Symagulov, Yan Kuchin, Kirill Yakunin, and Marina Yelis. 2021. From Classical Machine Learning to Deep Neural Networks: A Simplified Scientometric Review. *Applied Sciences* 11 (2021). Issue 12. DOI:http://dx.doi.org/10.3390/app11125541

[32] Xinlei Pan, Yurong You, Ziyan Wang, and Cewu Lu. 2017. Virtual to Real Reinforcement Learning for Autonomous Driving. (2017). DOI:http://dx.doi.org/https://doi.org/10.48550/arXiv.1704.03952

[33] Hyunjong Park and Jongyoun Noh. Cvlab-yonsei/MNAD: An official implementation of "Learning memory-guided normality for anomaly detection" (CVPR 2020) in pytorch. (????). https://github.com/cvlab-yonsei/MNAD

[34] Hyunjong Park, Jongyoun Noh, and Bumsub Ham. 2020. Learning Memory-guided Normality for Anomaly Detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 14372–14381.

[35] Bharathkumar Ramachandra and Michael J Jones. 2020. Street scene: A new dataset and evaluation protocol for video anomaly detection. In *Proceedings - 2020 IEEE Winter Conference on Applications of Computer Vision, WACV 2020*. 2558–2567. DOI: http://dx.doi.org/10.1109/WACV45572.2020.9093457

[36] Mayu Sakurada and Takehisa Yairi. 2014. Anomaly Detection Using Autoencoders with Nonlinear Dimensionality Reduction. In *Proceedings of the MLSDA 2014 2nd Workshop on Machine Learning for Sensory Data Analysis (MLSDA'14)*. Association for Computing Machinery, New York, NY, USA, 4–11. DOI: http://dx.doi.org/10.1145/2689746.2689747

[37] Waqas Sultani, Chen Chen, and Mubarak Shah. 2018. Real-world Anomaly Detection in Surveillance Videos App We should not only judge whether there is an accident in the image, but also determine the location of the accident accurately. (2018). http://arxiv.org/abs/1801.04264

[38] M. Suresha, S. Kuppa, and D. S. Raghukumar. 2020. A study on deep learning spatiotemporal models and feature extraction techniques for video understanding. *International Journal of Multimedia Information Retrieval* 9 (6 2020), 81–101. Issue 2. DOI:http://dx.doi.org/10.1007/S13735-019-00190-X/FIGURES/11

[39] Yao Tang, Lin Zhao, Shanshan Zhang, Chen Gong, Guangyu Li, and Jian Yang. 2020. Integrating prediction and reconstruction for anomaly detection. *Pattern Recognition Letters* 129 (2020), 123–130. DOI: http://dx.doi.org/https://doi.org/10.1016/j.patrec.2019.11.024