Bachelor Thesis

DumpingMapper: Illegal dumping detection from high spatial resolution satellite imagery

Laurens van Helvoort

February 2023

Supervisor: Dr. A. Kamilaris

Critical observer: Dr. F. Ahmed

Bachelor
Creative Technology

University of Twente

Abstract

The illegal dumping of waste materials presents a significant challenge for environmental conservation efforts due to the potential risks it poses to human health and natural habitats. The timely localization and disposal of illegal dumping sites can help prevent further contamination of its surroundings. Currently, the predominantly used method for detecting illegal dumping sites involves manual interpretation of satellite imagery, which is time-consuming, error-prone, and expensive for certain communities. To address this issue, there is a need for more efficient and accurate automated detection methods using satellite imagery. This study aimed to investigate the viability of Adversarial Autoencoder (AAE) models for detecting illegal dumping sites from satellite imagery. Initially, experiments were conducted using Generative Adversarial Networks (GANs), including vanilla GAN, Wasserstein GAN (WGAN) and Wasserstein GAN with gradient penalty (WGAN-GP). However, the study later pivoted towards AAE models, which were found to produce more informative embeddings that improved anomaly detection. The AAE model was trained to produce embeddings that represent the unique features of each satellite image. These embeddings were then visualized using certain techniques, such as dimensionality reduction methods and clustering techniques, to better understand the patterns and structure within the data. The results of this study demonstrate the promising potential of using AAE models for detecting illegal dumping sites from satellite imagery. By providing a more efficient and cost-effective approach to monitoring and identifying potential illegal dumping sites, this technology can help promote sustainable environmental management and public health.

Acknowledgements

I would like to express my thanks and gratitude to dr. Andreas Kamilaris for his insight, advice, feedback and guidance during this graduation project. Furthermore, I would like the thank my critical observer Faizan Ahmed for the useful feedback I received.

I would also like to give special thanks to dr. Savvas Karatsiolis for his profound knowledge and competence in the field of machine learning and computer vision as my daily supervisor in this project. Without their guidance, this project would not be what it is now.

Additionally, I would to thank the CYENS Centre of Excellence for providing the basis of this interesting project. I would also like to thank my fellow student Jochem G. D. Sallander for his involvement in creating the dataset that was used in this project.

Lastly, I would like to thank my friends and family for their support during this graduation project.

Table of Contents

Abstract 1

Acknowledgements 2

Table of Contents 3

List of Figures 5

1 Introduction 6

1.1 Research objectives 7

1.2 Thesis structure 7

2 Background research 8

2.1 Background research structure 8

2.2 Deep learning and applications 8

2.2.1 Deep learning 8

2.2.2 Applications of deep learning 9

2.3 CNNs for image analysis 10

2.3.1 CNN architecture 10

2.3.2 CNN-based object detection models 11

2.4 Satellite datasets 12

2.4.1 Satellite imagery types 12

2.4.2 Characteristics 13

2.5 Training data 14

2.5.1 Satellite training data for CNNs 14

2.5.2 Synthetic data 15

2.6 Related work 16

2.7 Adversarial Autoencoders 17

2.7.1 Autoencoders 17

2.7.2 Generative Adversarial Networks 18

2.7.3 Adversarial Autoencoders 19

2.7.4 Adversarial Autoencoders for anomaly detection 20

3 Methods and techniques 21

3.1 Creative Technology Design Process 21

3.2 CRISP-DM 23

3.3 Approach 24

4 Ideation 25

4.1 Dataset 25

4.1.1 Cropping and annotation 26

4.2 Possible approaches 27

4.2.1 Time-series LSTM 27

4.2.2 Anomaly detection 28

5 Realisation 29

5.1 Tools 29
5.2 Generative Adversarial Network 29
5.2.1 GANs for anomaly detection 29
5.2.2 Vanilla GAN implementation 29
5.2.3 WGAN with gradient penalty 31
5.3 Adversarial Autoencoder 32
5.3.1 New dataset 33
5.3.2 Model implementation 34
5.4 Processing embeddings in latent space 35
5.4.1 Principal Component Analysis 35
5.4.2 K-means clustering algorithm 37
5.4.3 t-SNE 40
5.5 Alterations to initial planning 41
6 Evaluation 42
6.1 AAE performance 42
6.1.1 Images on decision boundary 43
6.2 Comparison to other approaches 45
6.2.1 CNN 45
6.2.2 GAN 45
7 Conclusion 46
7.1 Research findings 46
7.2 Answer to research questions 47
7.2.1 Sub-questions 47
7.2.2 Main question 48
8 Discussion 49
8.1 Limitations 49
8.2 Strengths and weaknesses 50
8.3 Future work 51
8.3.1 Modelling 51
8.3.2 Hyperparameters 51
8.3.3 Generalizability 51
8.3.4 Satellite imagery type 52
8.3.5 Alternative visualisation techniques 52
Appendix A 53
Appendix B 54
Appendix C 55
References 57

List of Figures

Figure 1 The basic structure of a deep learning neural network 9
Figure 2 Architecture of an Autoencoder 17
Figure 3 Architecture of a GAN [66] 18
Figure 4 Architecture of an Adversarial Autoencoder 19
Figure 5 The Creative Technology Design Process [53] 22
Figure 6 Diagram of CRISP-DM, by Kenneth Jensen 24
Figure 7 Map of reported dumping sites in Cyprus 25
Figure 8 Original and cropped image 26
Figure 9 Unusable images 26
Figure 10 Vanilla GAN, 200 epochs, batch size of 32 30
Figure 11 Vanilla GAN, 400 epochs, batch size of 64 30
Figure 12 WGAN with gradient penalty, 1500 epochs, batch size 64 31
Figure 13 Bottom right and top left re-cropping of the original image 34
Figure 14 Pseudocode for creating the (dumping) embedding groups 35
Figure 15 Flattening embeddings code 36
Figure 16 k-means clustering implementation 37
Figure 17 k-means clustering scatterplot with k=2 37
Figure 18 Elbow plot 38
Figure 19 Silhouette score plot 39
Figure 20 k-means clustering with k=3 39
Figure 21 t-SNE implementation code 40
Figure 22 t-SNE scatterplot 41
Figure 23 Non-dumping image on decision boundary (cloud) 43
Figure 24 Non-dumping image on decision boundary (body of water) 44
Figure 25 Dumping image on decision boundary (lacking resolution) 44
Figure 26 Brief overview of the experiments performed in chapter 5 46
Figure 27 Table summarizing related works 53
Figure 28 Python centre-cropping script 54
Figure 29 k-means clustering with k=4 55
Figure 30 k-means clustering with k=6 55
Figure 31 k-means clustering with k=8 56
Figure 32 k-means clustering with k=10 56

1 Introduction

Many municipalities have designated organizations or landfills for their inhabitants to safely dispose of their solid waste in a controlled environment. Proper waste management is critical for people that live in urban areas, especially with a 52% urbanization percentage in 2022 [1]. However, illegal waste dumping has become an increasingly common threat to humans, animals and the environment [2]. Illegal waste dumping can be described as the disposal of trash generated at one location and disposed of at another location without legal permission [3]. This could be for instance used household garbage, used tires, old mattresses or garden waste in places such as the roadside, forests and other non-designated dumping sites. However, at a larger scale, construction and industrial waste are especially harmful to the surrounding area, as they can contain toxic substances. This waste then gets illegally buried, disposed of at a landfill or burned.

This has a severe impact on soil quality, surface water, air quality, wildlife and human wellbeing. Due to a lack of access or opportunity to better alternatives, parts of the population are forced to live in close proximity to these illegal sites, causing them to develop severe health complications, both in the long and short term. Short-term health issues may include respiratory diseases, general anxiety, headaches, nausea and eye irritation [2], [4] while long-term exposure to illegally dumped waste may lead to specific types of cancer [5], cardiovascular diseases, malignant neoplasms and birth defects.

In 2015, the UN adopted the Sustainable Development Goals (SDG), one of which being SDG 12: Responsible Consumption and Production [6]. This SDG addresses the growing problem of improper waste management and unlawful toxic and chemical waste disposal among others. It calls for environmentally sound waste management to reduce waste released into soil, air or water to minimize negative health impacts on humans and the environment. Therefore, in consonance with the SDGs, illegal dumping sites must be localized and disposed of responsibly, and the areas affected must be restored and redeveloped for them to be safe for flora and fauna.

One challenge in the aforementioned statement, however, is often the localization of these illegal dumping sites. The GWMO (United Nations Environment Programme and International Solid Waste Association) [7] also voiced the need for data on dumping, stating: "availability and reliability of waste and resource data are dire, and urgently needs attention". Due to the dumping sites being illegally created, it can take significant amounts of time to discover their precise locations. Therefore, to alleviate the grave impact illegal dumping has on its environment, the timely detection of illegal dumping sites is crucial. There have been recent advancements in the field of automatic illegal waste detection, however, the predominantly used method by municipalities and local governments is still manual detection from photos. This process requires expert knowledge and is slow, inaccurate and expensive.

Therefore, an effective solution is needed to detect illegal dumping sites in order for municipalities and local governments to combat this rising problem. There exists a wide range of methods and solutions, however, the use of machine learning, in particular deep learning, has proven to be effective in extracting waste locations from satellite imagery or aerial imagery [8]. Deep learning models have been successfully deployed in the detection of illegal dumping from satellite imagery due to their strong image analysis capabilities. Satellite imaging is readily available and has an adequate resolution for the intended purpose, therefore this will be the input for the model of focus in this solution.

This paper explores the possibility of using deep learning techniques to effectively detect illegally dumped waste near real-time from satellite imagery.

1.1 Research objectives

Thus, the main objective of this paper is to examine the possibility of using a deep learning model to automatically and reliably detect the locations of illegal dumping sites from satellite imagery. The main research question is formulated as follows:

> **RQ:** *How can a deep learning model be developed for the purpose of detecting illegal dumping sites from satellite imagery?*

In order to answer this question, sub-questions must be formulated. First, we need to know what framework or structure is best for this detection task. Furthermore, knowledge is needed on what satellite imagery datasets can be used for training purposes and input of the model. This results in the following sub-questions:

> **SQ1**: *What deep learning framework is optimal for dumping detection from satellite imagery?*
>
> **SQ2**: *What satellite data will be useful for training and as input for the deep learning model?*

1.2 Thesis structure

In the following chapters, the full process of research, ideation, implementation, training and evaluation will be documented. In chapter 2, the conducted background research is recorded. This will provide an overview of state-of-the-art deep learning detection models, relevant satellite datasets and possibilities for training the model. In chapter 3, the methods and techniques for the development of such a model will be outlined. Chapter 4 will document the ideation and design process of the deep learning model. Chapter 5 will discuss the realisation, implementation and results of the project. In chapter 6, the results of chapter 5 will be evaluated. Chapter 7 will outline the conclusion of the project and chapter 8 will discuss the limitations of the project and opportunities for future research.

2 Background research

The goal of this background research is to investigate the concept of deep learning, applications of deep learning models and existing deep learning models for object detection. Current deep learning-based solutions for illegal dumping detection will also be explored.

2.1 Background research structure

This background research will be split up into multiple parts. Firstly, an overview of deep learning concepts and detection methods is given. Secondly, types of satellite datasets will be examined along with their features and specifications. Thirdly, training data as an input to the deep learning model will be discussed, as well as the option of synthetic data generation for training. Fourthly, existing studies on dumping detection will be reviewed. Lastly, a brief explanation of Adversarial Autoencoders is given.

2.2 Deep learning and applications

2.2.1 Deep learning

There are multiple definitions that aim to encompass the concept of deep learning (DL). Marcus [9] defines DL as a statistical technique for classifying patterns, based on sample data, using neural networks with multiple layers. Cullel-Dalmau et al. [10] and Sarker [11] agree on the definition that DL is a subset of artificial intelligence (AI) that aims to mimic the way the human brain works, in particular the connections between neurons. Deng et al. [12] provide the most detailed definition, writing that deep learning "uses a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. Each successive layer uses the output from the previous layer as input, learn multiple levels of representations that correspond to different levels of abstraction; the levels form a hierarchy of concepts". This is achieved by making use of several layers in a neural network, as Pak et al. [13] added. In this paper, we will use the definition as formulated by Deng et al. [12]. In 2006, Hinton et al. [14] first proposed the DL algorithm that could process large amounts of data with an impressive learning speed compared to existing approaches at the time. DL has become a subcategory of AI that has risen in popularity over the past five years due to its efficacy and versatility, as observed by Sarker [15]. DL is a commonly used and preferred method in machine learning in an attempt to mimic the way connections between neurons in the human brain work.

Typically, it makes use of digital 'neurons' or nodes, which are connected to other nodes to transmit signals. As defined by Cullel-Dalmau et al. [10], these signals consist of a number, where the output of the nodes is a nonlinear function of the sum of the inputs. The connections between nodes are characterized by weights that dictate how much a signal contributes to the output. Nodes are then structured into different layers, which are interconnected as well. In the most basic form, there are three types of layers in a DL neural network: the input layer, the hidden layers and the output layer, as illustrated in Figure 1. The input layer represents the inputs that the DL model receives, the hidden layers are the layers where the values for nodes are calculated and the output node represents the summation of the previous layers resulting in an output. The lower layers close to the data input learn simple features, while higher layers learn more complex features derived from lower-layer features as described by Srivastava et al. [16] and Shinde et al. [17]. In a DL network, more nodes and layers entail that the model is able to perform more complex tasks, albeit at a higher computational cost.

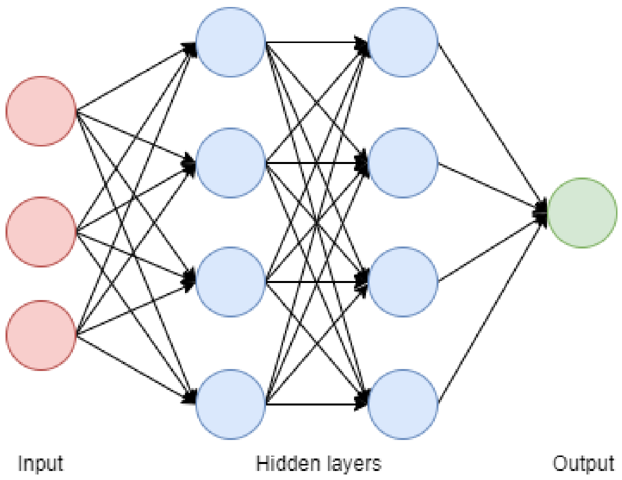

Figure 1 The basic structure of a deep learning neural network

2.2.2 Applications of deep learning

As mentioned before, DL neural networks are highly versatile systems which are applied in many research areas and aspects of society. There are countless examples of DL being used in a variety of fields, four of which will be covered in this section: computing science, healthcare, business intelligence and customer service.

Firstly, DL supports state-of-the-art applications in computing science research areas. These include natural language processing (NLP), computer vision (CV), image analysis, sentiment analysis, speech recognition and many more, as summarized by Sarker [11] and Pathak et al. [18]. In 2016, Google's DL-based AlphaGo famously beat one of the strongest players, Lee Sedol, in the ancient Chinese strategy game of Go, proving DL's fast learning capabilities and further popularizing DL to the public. Thus, DL shows promising results in fields of study in computing science, but many other major research fields are currently exploring the value of DL as well.

Secondly, the use of DL in healthcare has also been explored. DL networks have been deployed in the detection of breast cancer from images with a 0.966 accuracy score as researched by Wang et al. [19]. Prediction of risks in patients based on current clinical status is also a rising topic in research. Lui et al. [20] have also obtained state-of-the-art results in early Alzheimer's disease diagnoses with a DL model. Furthermore, Tran et al. [21] predicted suicide risks in mental health patients and Miotto et al. [22] built a model for the prediction of future diseases from the patient's current clinical records. High-level classifications of skin cancer from images have been achieved by Cullel-Dalmau et al. [10]. Esteva et al. [23] further proved this technique in skin cancer classification reaching dermatologist-level results. This is merely a peripheral overview of recent DL developments in healthcare, but this method has promising prospects.

Thirdly, DL has also solidified its place in business intelligence, with many of the world's largest companies like Amazon, Netflix, Microsoft, Spotify and Facebook making use of this powerful machine learning technique. DL can provide many useful services to companies like the prediction of customer purchase behaviour, as demonstrated by Chaudhuri et al. [24]. Furthermore, streaming services, e-commerce platforms and social media platforms oftentimes utilize DL to design complex recommendation systems that provide users with new relevant content as researched by Da`u et al. [25] and Singhal et al. [26]. These recommendations are based on personal collected historical data to keep users engaged for longer in the platform.

Lastly, DL models have been deployed in intelligent chatbots for customer service. Facebook, WhatsApp and Telegram use DL methods to make context-aware messaging chatbots on their platforms. [27]–[29] demonstrate the use of DLs with NLP for intelligent chatbots in e-commerce among other applications. This underlines the versatility of DL in many, often subtle, aspects of daily life.

2.3 CNNs for image analysis

2.3.1 CNN architecture

For object detection, Convolutional Neural Networks (CNN) are the predominantly used method in DL. The reason for this is mostly its impressive feature extraction. In computer vision, feature extraction is a process by which an initial set of data is reduced by identifying key features of the data [30]. In traditional models, feature extractors were designed manually for specific tasks. The main benefit of the use of CNNs to its predecessors is that they can automatically identify relevant features in data, meaning without manual human labelling on what the model should look for. CNNs are also strongly optimized for processing 2D input-data structures like images as mentioned by Alzubaidi et al. [31]. Accordingly, they are widely used in pattern- and image-recognition problems.

In a CNN neural network, the first hidden layer recognizes a set of primitive patterns in the input, the second layer detects patterns within the patterns of the first layer, the third layer detects patterns of those patterns and so on until it reaches the output layer. Typically, CNNs are composed of distinct layers, meaning that more layers give rise to more complexity. This leads to increased levels of abstraction, meaning that the model eventually is able to recognize entire objects, such as cats, humans, tumours, dumping sites etc.

CNNs have multilayer hierarchical structures, typically featuring alternating convolutional and pooling layers followed by a fully connected layer. The convolutional layer operates on a small area of the original input image. Then there is a feature detector, also known as a kernel or filter which will move across fields of the image for the detection of features. This process is called a convolution [32]. The output of the convolutional layer goes through an activation function which will result in a convolved feature map [33]. More abstract and sophisticated features can be extracted as these feature maps will be the input of subsequent convolutional layers.

Generally, after the convolutional layer comes the pooling layer. They include the maximum, average and random pooling. The maximum and average pooling layers calculate the respective maximum and average values of neighbouring neurons, as explained by Song et al. [33]. The random pooling layer selects values for neurons based on a particular probability. The goal of the pooling layer in a CNN is to capture features in the input received from the previous layer. It is, however, not able to locate the precise location of these features. This means that if there is a shift in the input data, it will still be able to effectively detect the features as stated by Song et al. [33]. This layer also reduces the dimensionality of the feature maps, leading to less computational cost.

Lastly, there is the fully connected layer. This fully connected layer features several hidden layers which are composed of neurons, where each neuron is also interconnected with the neurons of the subsequent layer as laid out by Song et al. [33]. The fully connected layer aims to map the features it has received from the convolutional and pooling layers and map them into linear space and coordinates with the output layer. Then, the output layer typically uses a classification function to output the results of the classification. This classification function is commonly a Softmax function or support vector machine (SVM) [33].

Other components that are crucial to the CNN are the activation and loss functions. The activation function is a nonlinear function that essentially decides whether a neuron in the network should be activated or not. This will dictate whether a neuron’s input is important to the network or not. Commonly used activation functions are the Sigmoid, Rectified Linear Unit (ReLU) or Maxout functions. The loss function represents the difference between the expected outcome of the model and the predicted or detected outcome by the CNN. Typical loss functions include the cross-entropy and mean squared error statistics.

2.3.2 CNN-based object detection models

There are different architectures for building a CNN model for object detection. The three most accomplished of these CNN detection models are Faster R-CNN, SSD and YOLO. Faster R-CNN is the culmination of two of its predecessors: Fast R-CNN and R-CNN. R-CNN (region-based CNN) is a first trial towards building an object detection model that extracts features using a pre-trained CNN. Then, Fast R-CNN was developed, which was faster than R-CNN, but neglected how region-proposals (division of the image in regions) are generated. Ren et al. [34] later solved this with Faster R-CNN, which builds a region-proposal network that can generate region proposals. These are then inputted into the Fast R-CNN detection model to inspect for objects. When an object is detected, it outputs the image with a bounding box (rectangle containing the object) pasted on top of it.

SSD (single-shot detection) is a CNN-based object detection method is another method that yields impressive results. It does the task of localization and classification of objects in one pass of the network. The main benefit of SSD in object detection is that it produces bounding boxes at different scales and aspect ratios, which is a shortcoming of similar object detection algorithms. This makes SSD more accurate, at the compromise of slower speeds.

YOLO (you only look once) is based on the concept that objects within an image can be detected and classified at one glance. In object detection, traditional detection systems apply a model to an image at multiple locations and scales in the image to compare. Then, high-scoring regions of the image, meaning a high possibility of the desired object in the image, are considered detections. What makes YOLO different from similar models is that YOLO applies the neural network to the full image rather than regions of it.

When comparing the three approaches for object detection from satellite imagery, we can look at several documented approaches to it. Van Etten [35] used the aforementioned techniques (or slight variants) to detect ships and airplanes from satellite images and found that YOLO was the overall best-performing technique. Compared to Faster R-CNN and SSD, Van Etten [35] wrote that YOLO was by a great amount the fastest and most accurate of the three, with Faster R-CNN showing the worst results. Li et al. [36] researched the detection of greenhouses from satellite images, comparing the three techniques. They conclude that although YOLO, Faster R-CNN and SSD all show promising results, Faster R-CNN and SSD fail to satisfy the accuracy and speed requirements associated with high-resolution satellite imagery. Furthermore, Cheng et al. [37] successfully deployed a YOLO model for the detection of landslides in China from satellite imagery, with an accuracy of 94.08%. As landslides share similar features to dumping sites, this further supports the use of a YOLO model in waste detection. Moreover, Liu et al. [38] used a YOLO model for plastic waste detection from regular surveillance cameras but stated that their results lead them to believe there is great potential for satellite images as well. Therefore, for the use of large-scale images such as satellite imagery, YOLO is the best currently available object detection method.

2.4 Satellite datasets

In this section, factors that are important in selecting a satellite dataset will be discussed. The type of imaging sensor used, along with other important characteristics, will be addressed.

2.4.1 Satellite imagery types

On the topic of satellite datasets, there are several types to consider. For our purposes, Zhu et al. [39] distinguish three main types of satellite imagery: satellite images made with optical imaging sensors, radar imaging sensors and non-imaging sensors.

Firstly, optical imaging sensors. This remote-sensing equipment operates in the visible and infrared (IR) ranges. These sensors typically produce panchromatic, multispectral, and hyperspectral imagery. Panchromatic images, as explained by Zhu et al. [39], are captured by a sensor that is a monospectral channel detector that is sensitive to radiation within a broad wavelength range. The resulting image is a grayscale image. Multispectral means that the sensor is sensitive to a few spectral bands. Here, the resulting image is a multilayer image containing colour information, as well as brightness. Hyperspectral images are captured with sensors that are sensitive to 10 up to 100 spectral bands. The result is a set of images, where each image contains one spectral band. According to Zhu et al [39], the set of images can be easily used for purposes such as object recognition and material identification. Additionally, it is noteworthy that recording more spectral bands corresponds to a decrease in resolution.

Secondly, radar imaging sensors typically operate in the electromagnetic spectrum. As the name suggests, it utilizes radar technology to gather data on targets. An advantage of this imaging technique is that it is unaffected by weather such as clouds or fog, as explained by Zhu et al. [39]. Furthermore, it is able to measure through water, sand and walls.

Thirdly, non-imaging sensors are sensors that record the visible, IR and microwave spectral bands. According to Zhu et al. [39], typical non-imaging sensors include radiometers, altimeters, spectrometers, spectroradiometers, and LIDAR. However, the applications of these non-imaging sensors mainly focus on atmospheric features such as temperature and wind speed, therefore it may not be as applicable to this project.

2.4.2 Characteristics

In the context of detecting illegal waste dumping sites, three key factors to consider in satellite datasets include spatial resolution, temporal resolution, and environmental factors.

Spatial resolution is an essential feature in the detection of illegal waste from satellite imagery. Luyendyk et al. [40] define spatial resolution as an area on the ground represented by each pixel of the satellite images. This is usually represented by a number of meters, where a lower number represents a finer resolution. High spatial resolution satellite imagery can capture smaller features and provide more detailed information about the area of interest. In the context of detecting illegal dumping sites, high spatial resolution imagery enables the identification of smaller waste dumping sites. Additionally, high spatial resolution imagery can reveal specific details about the waste materials being dumped, such as the type, quantity, and location of the waste. This information can be used to develop targeted efforts to address the issue of illegal dumping in the area.

Temporal resolution is also critical for identifying and monitoring illegal dumping sites from satellite imagery. Temporal resolution refers to the frequency at which images are captured. Frequent image captures allow for the identification of patterns and changes in waste dumping activity over time. By analysing satellite imagery over time, we can identify areas where illegal dumping is most prevalent and track changes in dumping activity. Temporal resolution is also essential in monitoring the effectiveness of interventions aimed towards the culpable parties contributing to illegal dumping.

Environmental factors are another critical feature for identifying and monitoring illegal dumping sites from satellite imagery. Changes in land use, vegetation health, and surface water quality can all be indicators of waste dumping. For example, the presence of barren earth or disturbed vegetation may indicate the location of an illegal dumping site. Similarly, changes in vegetation health can indicate the presence of waste materials that may be affecting plant growth. Water quality changes can also indicate the presence of waste materials that are affecting water ecosystems. By monitoring these environmental factors over time, these indicators can hint towards areas where illegal dumping is likely occurring and track changes in dumping activity.

2.5 Training data

While there are numerous satellite imagery datasets to choose from, transforming these raw satellite images into datasets fit for training a CNN model presents challenges that are not often encountered in computer vision problems using 'regular images'. This requires more pre-processing and other techniques in order to curate a representative dataset for the model. In this section, challenges encountered by researchers in the field will be discussed, as well as data augmentation, using weakly labelled sample data in the dataset and synthetic data.

2.5.1 Challenges and solutions in satellite data for CNNs

Large amounts of training data are essential to developing an accurate prediction or detection model. However, there is a shortage of adequate remote sensing training data as observed by Song et al. [33]. Remote sensing/satellite datasets are more time-consuming to produce than regular computer science image datasets. Furthermore, Padubidri et al. [41] also noted that a large portion of the remote sensing datasets that are available are biased towards non-dumping-related applications. This is a major limitation faced by researchers in this domain.

Satellite images are fundamentally different from regular images. They can contain more spectral information such as optical imaging, thermal and LIDAR imaging and typically have a far greater resolution. Most of the CNN models in existence, however, were developed for the use of ordinary images as opposed to satellite imagery. This leads to challenges, as pre-trained models are accustomed to objects in front view which take up a large portion of the image, as observed by Song et al. [33]. Compare this to satellite imagery, where the model has to detect a tiny object from a large-scale image at a top-down view.

There have been improvements made to CNN models to account for the differing input images. Here, the type of dataset used is important, as covered in section 2.4. According to Song et al. [33], data augmentation can also greatly help in the training process. Data augmentation is the process of supplementing the training set with slightly different copies of images already in the training set. Long et al. [42] experimented with rotation, translation and scaling of the training satellite images containing oil barrels. After the data augmentation, the authors ended up with 60 times the original data as a training source and a detection score of 96.7%. Furthermore, Youssef et al. [43] also used augmented data in their remote sensing training set of aircraft classification, increasing their test accuracy from 72.4 to 97.2%. This means that using augmented data can effectively be used to improve accuracy.

Another improvement that can be made to CNNs to use weakly labelled sample data. One of the major challenges in creating a satellite imagery dataset, as stated by [44], is the manual annotation of the images in the dataset. This process is very labour-intensive, time-consuming and error-prone. According to Song et al. [33], weakly labelled training can moderately be used to achieve greater accuracy and IoU scores. As accurate sample labelling is a time and labour-intensive process, there are also many weakly labelled datasets. This means that the labelling is not complete or not accurate or that the data is of low quality. Song et al. [33] state that the inclusion of these weakly labelled images improves accuracy in the testing phase as the training set is more representative and extensive. Maggiori et al. [45] used a dataset containing errors and mislabelling and finetuned the prediction model based on the correctly labelled data. This led to the model having greater accuracy in object extraction. As remote sensing/satellite data can be sparse for deep learning purposes, this technique can be used to remedy that problem. It should be noted that only moderate use of weakly labelled data leads to improved results.

2.5.2 Synthetic data

Due to the lack of sufficient training data, another solution that can alleviate this problem is the use of synthetic data for training data. Padubidri et al. [41] explore the use of generating synthetic data to compensate for the shortcoming of dumping-related satellite imagery for training purposes. The authors use Blender, an open-source 3D modelling software, to produce the synthetic data. Publicly available 3D models of dumped objects and garbage were placed randomly on non-dumping satellite images. This process was automated via the use of Blender's Python scripting feature along with various add-ons. Generated synthetic data was iteratively tested on the CNN model that the authors used, and the synthetic data generation system was finetuned. Ultimately, 2000 synthetic dumping satellite images were created. The authors found that the CNN model performed better with the synthetic data as opposed to the performance using only authentic dumping satellite imagery. Padubidri et al. [41] obtained precision and recall scores of 0.98 and 0.90 respectively with a basic CNN model.

[44] also examines the possibility of combining synthetic data with real data to improve detection results in satellite imagery. The authors state that synthetic data can offer limitless customization. Any specification can be accommodated as the synthetic data is purpose-built. The authors [44] built this model in order to achieve more accurate vehicle detection from satellite imagery. They used 3D models of city blocks in the game development software Unity, along with randomized buildings and roads, crosswalks and bus lanes and even small imperfections such as road oil spills. Furthermore, they configured the option to change the time of day, cloud cover and intensity of the sun. The authors found that by using a combination of this synthetic data with the real dataset, superior results were achieved with respect to only using synthetic data or only using real data. This shows that there is great potential in the method of synthetic data for training and testing purposes.

2.6 Related work

Thus far, a few studies have attempted to automate the illegal waste dumping detection process utilizing deep learning or another method in machine learning. In [46], the authors use multi-spectral satellite images from the WorldView and GeoEye-1 satellites. Then, they use a CNN with a U-Net structure with multiple variations to detect landfills on multiple pre-trained models. The paper also puts forward a high-resolution landfill dataset which may be useful during this project for training purposes. Devesa et al. [47] use multi-spectral satellite imagery from the Sentinel-1 earth observation mission, with a 10m spatial resolution. Then, the authors also use the CNN-based U-Net segmentation model to detect and classify Argentinian urban solid waste. The authors use the evaluation metric of Intersection over Union (IoU) to assess the results. This metric is a commonly used method to quantify the overlap of the ground truth area and the predicted masked area in a percentage. Devesa et al. [47] achieved an IoU of 0.673 using RGB, IR, and certain shortwave IR (SWIR) bands as input for their model.

Another approach used by Silvestri et al. [48] is the maximum likelihood estimation algorithm (MLE) for the detection of dumped waste, in particular buried waste. They utilized high-resolution multispectral satellite images from the IKONOS dataset to detect areas where the soil is bare or vegetation is sparse. According to the authors, the presence of stressed or scarce vegetation is a good indicator to infer buried waste locations. They also underline the importance of GIS in their project as an auxiliary data source. GIS (Geographic Information System) is a type of database containing geographic data combined with powerful software tools for managing, analysing, and visualization purposes.

Furthermore, the use of drones for illegal waste detection has also been explored. Youme et al. [49] explore the use of drones at varying altitudes ranging from 5-30m for waste detection in West Africa. The drones, equipped with an L1D-20c RGB colour camera, have a very high spatial resolution of up to a few centimetres and are convenient for fast configuration of shutter speed, ISO and GPS coordinates. After the data acquisition through the drones, an SSD algorithm is used for the actual waste detection, with varying IoU scores up to 0.64. Mager et al. [50] also conducted a feasibility study on the use of drones for this purpose. They used drones equipped with cameras featuring a 2cm spatial resolution flying at a 50m altitude to gather data about their region of interest. Thereafter, they used GIS software (ArcGIS) to manually map the types of waste observed in the drone imagery.

Research has also been done on non-aerial imagery for the use of waste detection. Dabholker et al. [51] use hundreds of images from security cameras which locally run a CNN model (AlexNet & GoogleNet) to localize and identify the type of (domestic) garbage found. The model can distinguish between certain classes such as electronics, matrasses or furniture, achieving varying results depending on the class, ranging between a 0.7 and 0.95 accuracy score. Furthermore, Anjum et al. [52] make use of scene images in public and residential areas. They then input these images into a deep CNN model which can produce a segmented image with masks for garbage and non-garbage. The authors obtained a score of 4.1 on a 5-point scale on their survey of 500 collected images.

A table containing several studies in this area along with their techniques and findings can be found in Appendix A.

2.7 Adversarial Autoencoders

In this project, an approach that was later explored is the use of anomaly detection for the detection of illegal waste dumping from satellite imagery, explained further in section 4.2. Two common techniques used for anomaly detection are the generative adversarial network and adversarial autoencoder models. Adversarial autoencoders are a relatively new approach, which utilizes certain concepts of autoencoder models and generative adversarial network models. First, the latter two networks will be explained, followed by an explanation of adversarial autoencoders.

2.7.1 Autoencoders

An autoencoder (AE) is an unsupervised neural network introduced by Hinton et al. [59] and was traditionally used for feature learning and dimensionality reduction [60]. Its main goal is to reconstruct its original input as closely as possible. The autoencoder consists of an encoder and a decoder. The encoder is trained to map the input data to a code, or latent variable, and the decoder reconstructs the original data based on the code received from the encoder [59]. A diagram of this structure is depicted in Figure 2. An optimal autoencoder would perform as close to perfect reconstruction as possible, with "close to perfect" defined by the reconstruction quality function $\boldsymbol{d}$, which it will try to optimize.

The autoencoder cannot directly copy the input, rather it must copy approximations of the input data. To enforce this, the code space $\boldsymbol{Z}$ typically has a lower dimensionality than input space $\boldsymbol{X}$, compressing the data [59]. This way, the model can also be used for the dimensionality reduction of data, such as images. Dimensionality reduction can improve performance on tasks such as classification or anomaly detection and can make data more interpretable.

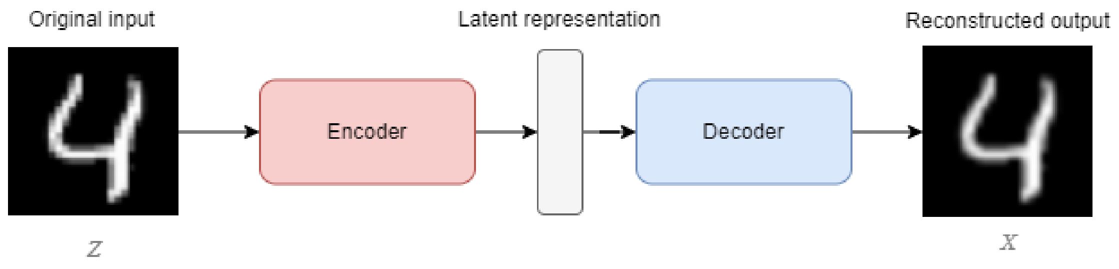

Figure 2 Architecture of an Autoencoder

2.7.2 Generative Adversarial Networks

A Generative Adversarial Network (GAN) is a system of two neural networks competing in an adversarial manner. The framework consists of two models that are trained concurrently, the generator and the discriminator.

The generator receives a dataset, after which its purpose is to generate new data as close to the input data, starting with random noise. Conversely, the discriminator is trained to differentiate between real and synthetic data samples. The networks are trained in an adversarial fashion, in which the generator tries to produce samples that are indistinguishable from real samples, and the discriminator tries to correctly classify the generated samples as fake.

Over a large training period, the generator will generate samples with such similarity to the real data that the discriminator cannot reliably distinguish the real images from the generated images. Thereafter, the discriminator can be discarded and a generator capable of generating images close to the inputted dataset remains. GANs can be used for data generation, such as generating faces [61], text-to-image translation [62], style transfer and upscaling low-resolution images to high-resolution images [63], but also anomaly detection [58], [60], [64], [65], which will be elaborated on in section 5.2.1. A diagram of the architecture of a GAN model is given in Figure *3*.

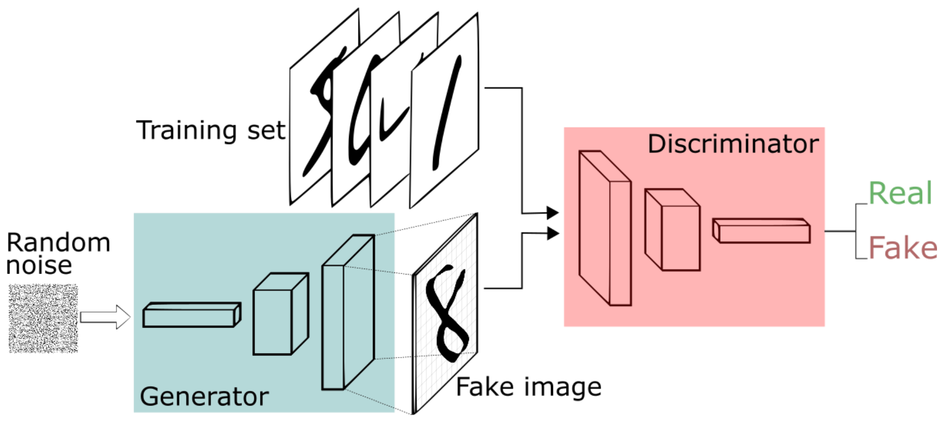

Figure 3 Architecture of a GAN [66]

2.7.3 Adversarial Autoencoders

The Adversarial Autoencoder (AAE) is an approach introduced by Makhzani et al. [67] that combines Autoencoders with Generative Adversarial Networks. An AAE is comprised of an encoder, decoder, generator and discriminator. In an AAE, the encoder component of the AE is trained to produce latent codes that are similar to the noise input of the GAN. The generator component is then trained to produce new samples from the latent codes. The discriminator component is trained to differentiate between the synthetic samples generated by the generator and real samples. This results in the generator producing synthetic samples that are similar to real samples, while the encoder preserves the structural properties of the input data in the latent codes.

Furthermore, the encoder is trained to map the input data to a probability distribution that is similar to the prior distribution. The encoder is trained to minimize the divergence between the encoded distribution and the prior distribution. This encourages the encoder to produce latent codes that are similar to the noise input and to preserve the most important features of the input data in the latent codes [67]. The AAE is trained by training the encoder, generator, and discriminator in parallel. This results in a model that can produce new samples that are similar to the input data and preserves the structure of the input data in the latent codes.

Figure 4 shows the architecture of an AAE. In this figure, $\boldsymbol{x}$ is the input data, $\boldsymbol{q}(\boldsymbol{z}|\boldsymbol{x})$ is the encoding distribution and $\boldsymbol{x}$ is the latent representation as an output of the encoder. In the generator component, $\boldsymbol{p}(\boldsymbol{z})$ is the prior distribution that we want to impose on the codes [60]. $\boldsymbol{q}(\boldsymbol{z})$ is given by $q(z) = \int_x q(z|x)\, p_d(x)\, dx$ where $p_d(x)$ is the data distribution.

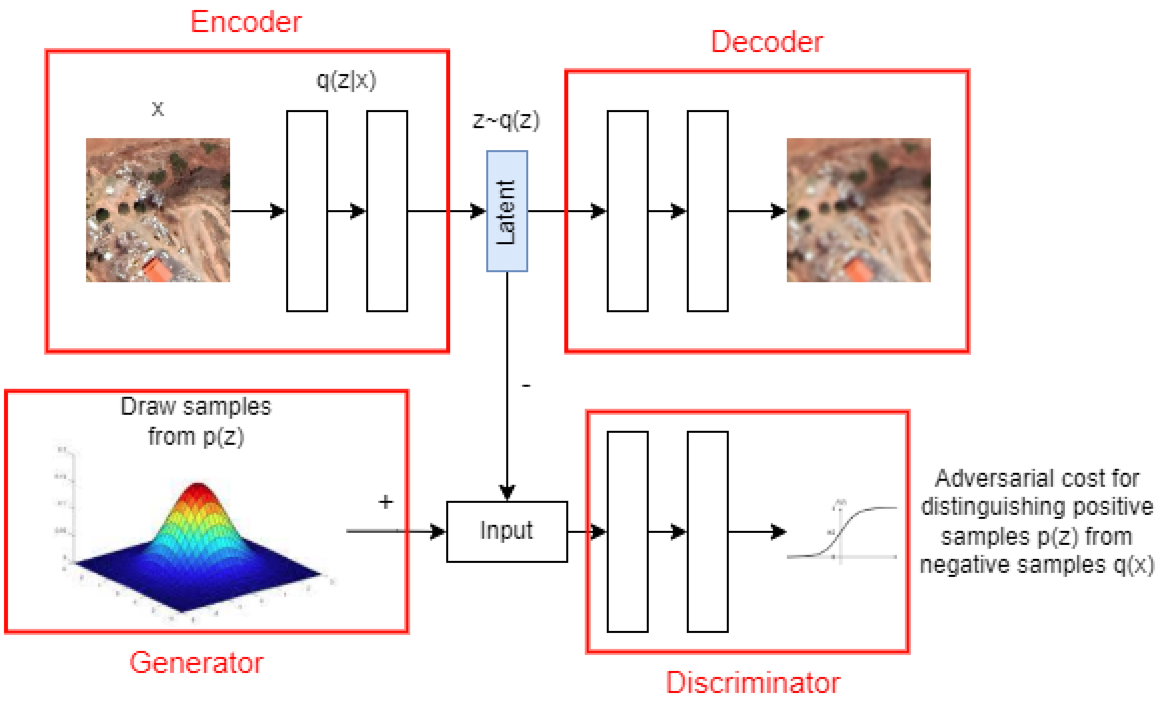
Encoder
q(z|x)
x
z~q(z)
Latent
Decoder
Draw samples from p(z)
+
-
Input
Adversarial cost for distinguishing positive samples p(z) from negative samples q(x)
Generator
Discriminator

Figure 4 Architecture of an Adversarial Autoencoder

2.7.4 Adversarial Autoencoders for anomaly detection

The goal of anomaly detection is to identify observations in a dataset that significantly deviate from the remaining observations [68]. In most cases of anomalies, it is not feasible to construct a representative dataset of possible forms of anomalies, as they vary significantly in nature [64]. Therefore, a more appropriate approach is to construct a training dataset of the normal observations, in this case, satellite images that do not contain dumping. Then, a model processes new data, and when it deviates significantly enough from the learned model (an image containing dumping), it can be classified as an anomaly.

Autoencoder networks such as adversarial autoencoders have shown superior performance over other models when given the task of anomaly detection in high-dimensional data such as images. The idea behind AAEs for anomaly detection is that normal data points will have a higher likelihood under the encoded distribution produced by the encoder, compared to anomalous data points. During the training phase, the AAE learns to encode normal data points into latent codes that are similar to the prior distribution. This prior distribution is chosen to be the multivariate Gaussian distribution in this case.

During the detection phase, the AAE is used to encode new data points into latent codes. The likelihood of each data point under the encoded distribution produced by the encoder is then calculated. Data points with lower likelihoods are considered as an anomaly. The threshold for determining the anomaly can be selected based on the distribution of the likelihoods of the normal data points. This way, AAEs can be used to detect anomalous data points that are different to the normal data points in terms of the encoded distribution [60]. This is different from traditional methods such as reconstruction-based methods that rely on the reconstruction error to detect anomalies. AAEs can detect anomalies that are not only dissimilar to the normal data points in terms of reconstruction but also in terms of the encoded distribution.

Additionally, AAEs can also be used for semi-supervised anomaly detection, where a small amount of labelled anomalous data is used to fine-tune the model, making it more robust to detect anomalies. One such model is the GANomaly, which combines features of the AE and GAN very similarly to the AAE [69].

3 Methods and techniques

This chapter will cover the methods and techniques used in the design process of the illegal dumping detection system. First, an overview will be given of the Creative Technology Design Process. Next, the CRISP-DM standard process will be discussed, a more data science-oriented approach to the design process. Both these techniques will serve as an underlying architecture for the project.

3.1 Creative Technology Design Process

To answer the research questions posed in section 1.1, a solid design approach is needed. This project will use the Creative Technology Design Process by Mader & Eggink [53]. This approach is an iterative design process, meaning that a prototype will be built, evaluated and improved upon in a cyclical fashion. The process consists of four phases: the ideation phase, specification phase, realisation phase and evaluation phase.

Firstly, in the ideation phase, a problem statement is formulated, and relevant information or required knowledge is acquired. Inspiration for the ideation phase may come from existing solutions or other related work identified in chapter 2. In this project, this involves looking at relevant or similar existing solutions, examining their methods and results and determining what would be good approaches to this problem. This can involve multiple concepts, as per the Creative Technology Design Process [53].

The specification phase is where the multiple ideas from the ideation phase are built and evaluated. Project requirements are formulated based on the acquired knowledge from the ideation phase [53]. In the realisation phase, a functional model is built according to the requirements set in the specification phase. Lastly, in the evaluation phase, the model built in phase 3 is tested based on users, target audience, or in our case the test dataset. Conclusions are made about the design and potential improvements or adjustments are explored.

This process is a cyclical process, meaning it would be possible to return to previous phases during the design process. For instance, when the feasibility of the design turns out to be questionable, it would be possible to return to the ideation phase. An overview of this design process can be seen in Figure *5*.

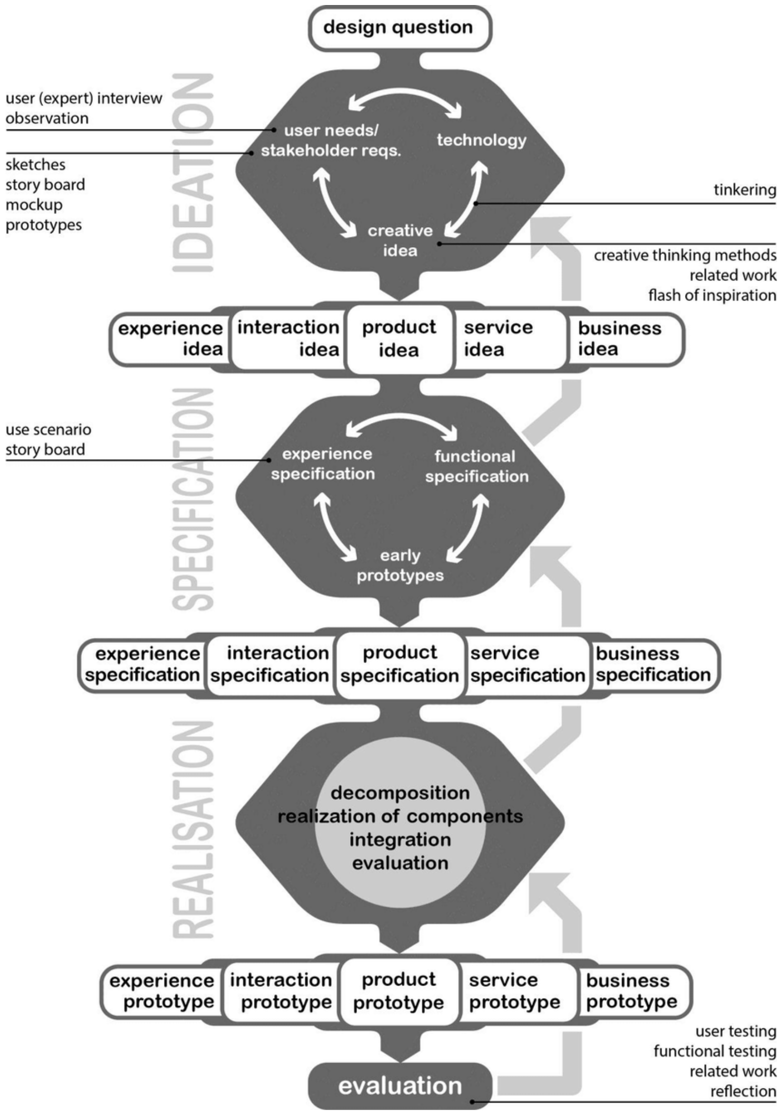

Figure 5 The Creative Technology Design Process [53]

3.2 CRISP-DM

A commonly used standard process in data science is the CRISP-DM model, which stands for Cross-Industry Standard Process for Data Mining. This methodology, introduced by Chapman et al. [54] aims to standardize the data mining process across industries. The CRISP-DM process consists of six sequential phases: business understanding, data understanding, data preparation, modelling, evaluation and deployment. The following section will provide additional information regarding the steps involved in the CRISP-DM process model, as described by [55].

1. **Business understanding** focuses on understanding the objectives and requirements of the project. An overview of the available and required information will be given. This knowledge will lead to a data mining problem definition and a preliminary plan in order to achieve the established objectives.

2. **Data understanding** starts with data collection, followed by activities to get familiar with the data. Data types should be identified, along with the number of rows and columns in the data. In the context of this project, it would mainly contain satellite imagery. Furthermore, data could be visualized and a statistical analysis of the quality of the data could be conducted.

3. **Data preparation** is the third phase in the process which covers the activities necessary to create the final dataset. This includes a selection of a dataset, relying on certain inclusion/exclusion criteria. Then the data is cleaned, constructed and integrated.

4. **Modelling** is when the actual model is built. This entails a choice of algorithm/model, the implementation of said algorithm, a system built for testing the model and lastly, the actual model is developed.

5. **Evaluation** is the fifth phase. In this phase, the model built in phase 5 is evaluated critically against the business objectives and problem definition determined in phase 1. If there as aspects of the model that do not meet certain expected results, in which case the cycle may be repeated to make adjustments or corrections to either the algorithm or the dataset.

6. **Deployment** is the phase a project enters if the results of the evaluation phase are satisfactory in light of the set objectives. In this last phase, a plan is made for the deployment of the model, next to plans for the monitoring and maintenance thereafter. Lastly, a final report should be made concluding the results of the implementation.

A diagram of the CRISP-DM process model and the relations between the phases can be Figure 6. Using the CRISP-DM model as a guideline in the coming steps of this project will be advantageous in providing an overview in organizing the data science project, as well as a clear view of progress for supervisors and stakeholders.

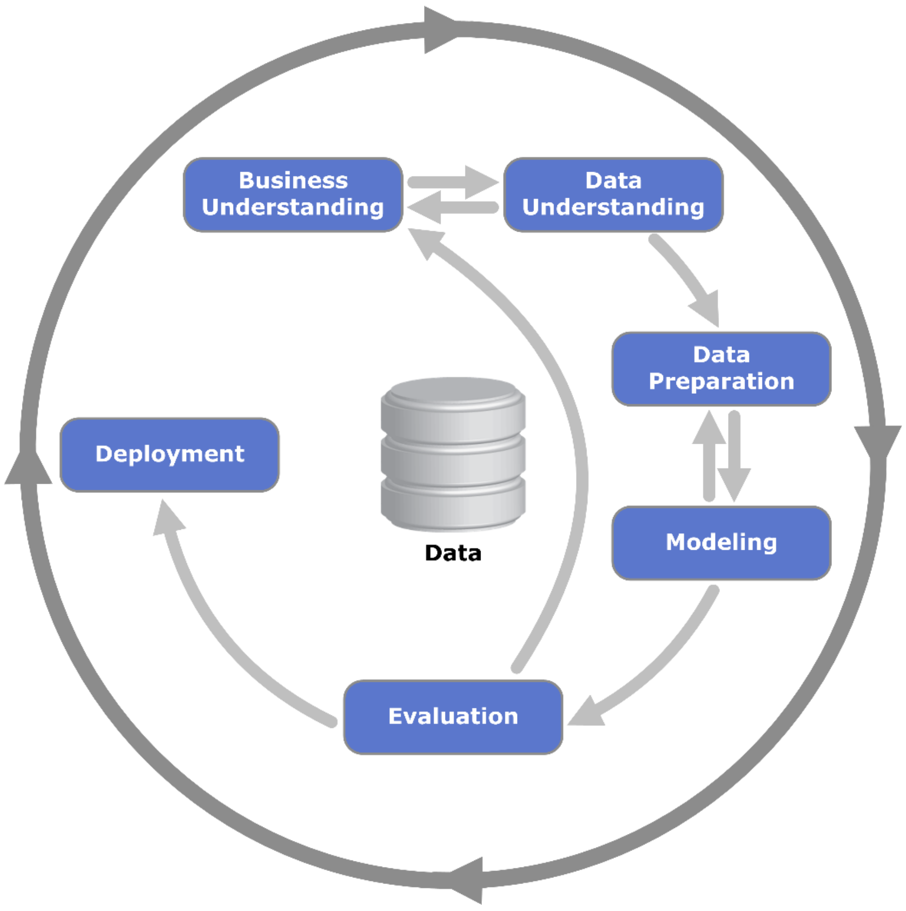

Figure 6 Diagram of CRISP-DM, by Kenneth Jensen

3.3 Approach

In this project, the Creative Technology Design Process and CRISP-DM will be utilized to create a comprehensive approach to detect illegal dumping.

Firstly, the Creative Technology Design Process will be applied to generate novel and creative ideas that will aid in the development of the deep learning model. The ideation will take place, wherein ideas for the model will be generated. These ideas will be analysed based on feasibility and potential effectiveness before being implemented. Here, principles of CRISP-DM will also be applied, keeping the data in mind when making these decisions. Additionally, testing will be conducted at every stage of the process to ensure that the model is performing adequately.

Secondly, the CRISP-DM process will be utilized to guide the data mining process. The process will begin with the initial data collection phase, wherein a dataset will be constructed. The next phase will be data preparation, wherein the data will be pre-processed and transformed for use in the model. This will likely involve manual annotation of the data in order for it to be suitable for a model. The modelling phase will come next, wherein various deep learning models will be evaluated and compared for their effectiveness in detecting illegal dumping sites. As this process is cyclical, changes to the data and/or the model can be made during this phase. As deployment to real use-cases falls outside of the scope of this project, the deployment phase of CRISP-DM is left out.

Overall, utilizing principles of both the design-oriented Creative Technology Design Process and the data-oriented CRISP-DM process will ensure a thorough approach to detecting illegal dumping from satellite imagery using deep learning.

4 Ideation

4.1 Dataset

The dataset that will be used is a dataset provided to use by the project coordinator containing satellite images of known locations of illegal dumping on the island of Cyprus. The data consists of 1016 citizen-reported dumping points across the country, demonstrated in Figure 7. Then, Google Earth Engine was used to create the dataset from the various locations, with 15 acquisition points for each location. Every 6 months, an image was captured of each location, going back 7,5 years in time. As Google Earth Engine allows for satellite imagery in time-series, some of the acquisition points may represent the environment prior to the initiation of waste deposition, at the time when waste material is disposed of or being disposed of, and potentially after clean-up efforts have been made. This entails that not every image in the ~15.000 image dataset necessarily contains dumping. Consequently, this means that this image dataset must be annotated, to allow for reliable training. The images will be sorted into two categories: dumping and non-dumping.

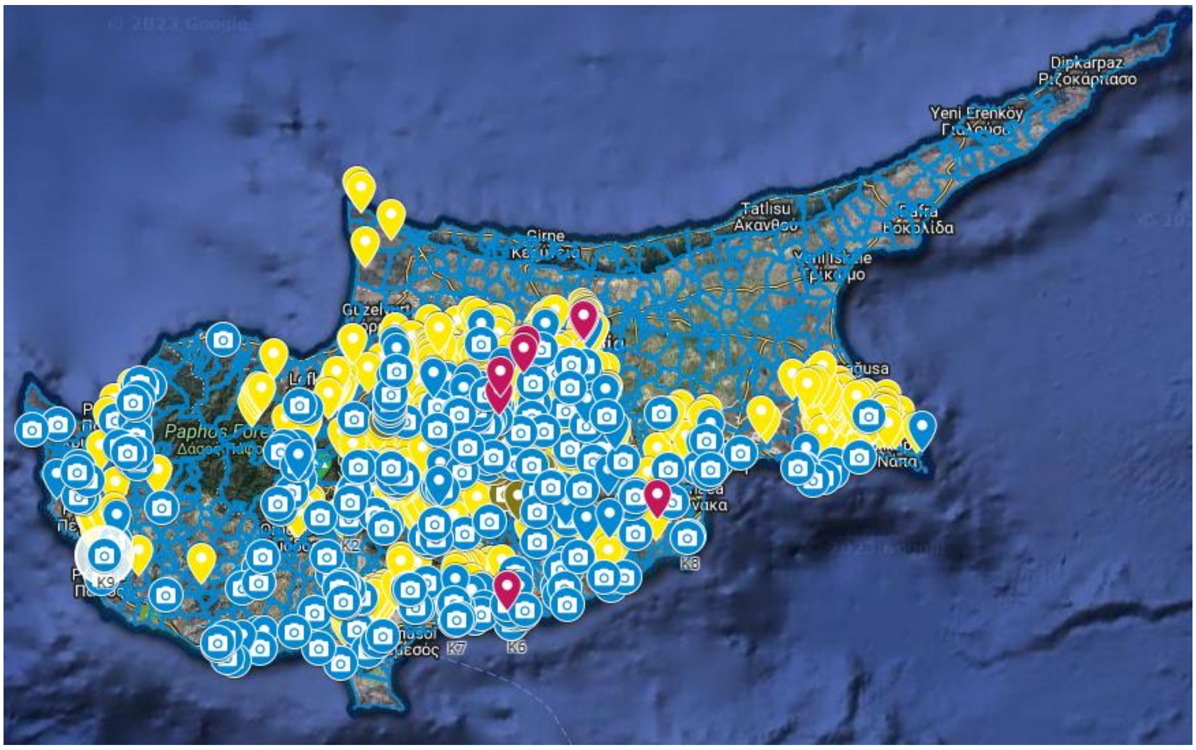

Figure 7 Map of reported dumping sites in Cyprus

4.1.1 Cropping and annotation

The images from the original dataset possess a very high resolution, with dimensions of 6663x8192 pixels. This high resolution exceeds the computational capacity of many machine learning models. Additionally, the manual annotation process for these images may also be affected due to the larger file sizes, potentially leading to a longer annotation process and decreased accuracy in the annotated data. Therefore, the images require cropping to a lower dimension to simplify the annotation process and reduce computational cost in the neural networks. The images are cropped to a 512x512 pixel size via a Python script provided in Appendix B. This 512x512 dimension is consciously chosen as most machine learning models support image sizes up to 512x512. Due to the way the dataset was created in Google Earth Engine, we know that the centre of the image represents the exact dumping location. Therefore, the cropping was done on the centre of the original image, as displayed in Figure 8. This size of patch still covers a large area of land and is fit to visually confirm whether the image contains dumping or not.

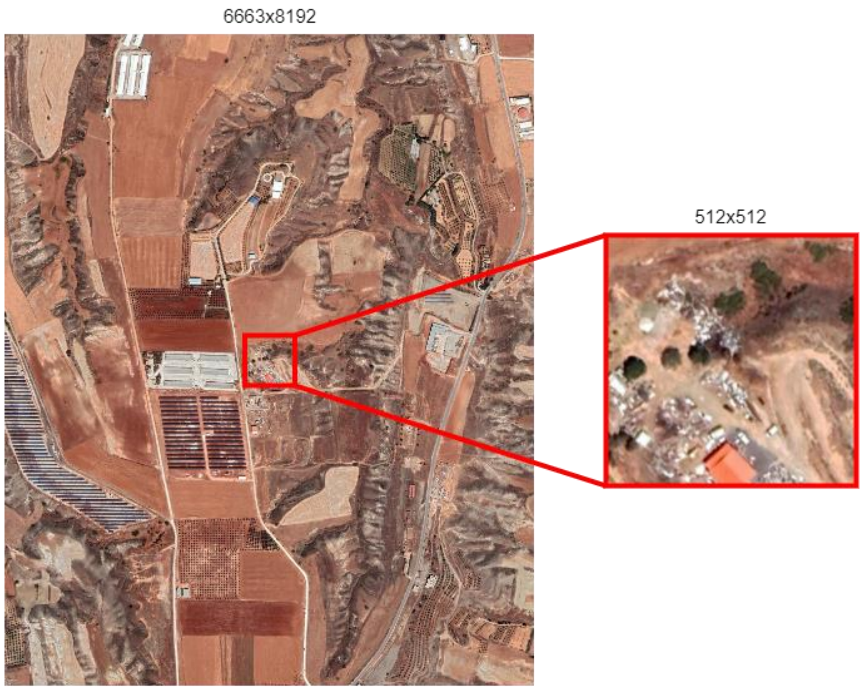

Figure 8 Original and cropped image

After cropping, the images were sorted into the two categories of dumping vs. non-dumping. The sorting process was made conducive by the use of the Image-Sort application [56]. During the sorting process, some images were discarded as unusable due to the land being completely obstructed by a cloud or images with exposure errors. Additionally, some images showed artefacts of the Google Earth Engine UI. Examples of these unusable images can be seen in Figure 9. The resulting final dataset to be used contained 9417 dumping images and 4212 non-dumping images.

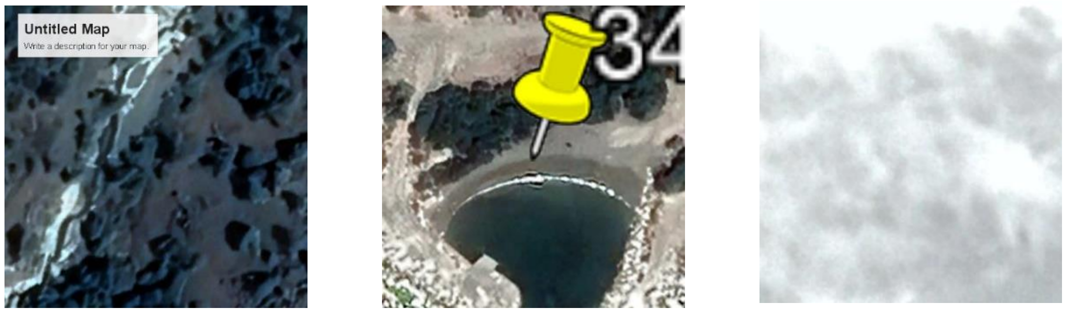

Figure 9 Unusable images

4.2 Possible approaches

During the ideation phase of this project, different approaches to illegal dumping detection were discussed. As mentioned in the state-of-the-art in chapter 2, the predominantly used approach is a CNN with a large dumping dataset and a large non-dumping dataset. The preliminary approach after the background research was to train a CNN model with YOLO architecture for this task. However, the YOLO models require a training dataset with images where the bounding box is already given. The current dataset does not support this, and there are no available YOLO-supported datasets with dumping vs. non-dumping satellite images.

In this project, we therefore ultimately decided to take a different, novel approach to this detection task. As CNNs are the more straightforward strategy, we wanted to explore different approaches and models. A selection of the approaches considered will be covered in the following section.

4.2.1 Time-series LSTM

One approach to address this problem is by utilizing a Long Short-Term Memory (LSTM) network. LSTMs are a type of Recurrent Neural Network (RNN) that are specifically designed to handle sequential data and retain memory of past inputs. The LSTM network is able to capture the temporal dependencies present in the time-series data, as also demonstrated by [57]. In a time-series model, the LSTM network would be trained on the historical data, which would include a series of dumping and non-dumping images. The network would then make predictions based on the current state and the information it has retained from previous inputs. In this case, the prediction would be whether a given image in the time-series is a dumping image or not.

By using an LSTM network, we can determine the point in time when dumping occurs by tracking the predictions made by the network. As the LSTM network processes each image in the time-series data, it would generate a probability score indicating whether the image is a dumping image or not. The point in time when the network's prediction crosses a certain threshold would correspond to the point when dumping occurs.

Given the dataset that we used, this ultimately did not seem a fitting choice for this problem. This is due to the fact that the 15 acquisition points for each location would not prove to be a clear time-series, as assumed initially. For instance, out of the fifteen images captured for each location, oftentimes dumping would occur in some images, disappear again and then reappear in the following images. These small variations may cause unreliability in a trained model. Additionally, there are images in a location where I, as a human interpreter of the images, know that a certain image contains dumping based on the context of the previous image and the next image containing dumping. However, due to strong shadows, clouds, disturbances or insufficient resolution, no dumping can clearly be seen in the image without the context of the surrounding images. These images were classified by me as non-dumping, but this raises concerns about whether this kind of data will be suitable for this approach.

4.2.2 Anomaly detection

Anomaly detection is generally understood to be the identification of rare items, events or observations which deviate significantly from the majority of the data and do not conform to a well-defined notion of normal behaviour. Anomaly detection has applications in many domains including cyber security, medicine, computer vision, statistics, neuroscience, law enforcement and financial fraud, to name only a few [58].

In our case, we can use anomaly detection to see which images contain dumping and which do not. The norm would be the non-dumping images, while the dumping images would be considered anomalous. Through anomaly detection, a model will be given a dataset containing only non-dumping images, leading the model to learn how the norm is represented. When this trained model is consequently presented with a new dataset containing a subset of the non-dumping images and the dumping images, the model will detect the dumping images as anomalies as they fall outside what the model has learned to be the norm [58].

This would be a more fitting approach to the detection problem, thereby now treating it as an anomaly detection problem. Given the data, it would seem that the dataset is better suited for anomaly detection tasks. Furthermore, anomaly detection models are significantly lighter and require less training and data to output meaningful results. Possible approaches to anomaly detection would be statistical techniques, GANs, RNNs or adversarial autoencoders to name a few. Due to the high dimensionality of the data, statistical techniques will not provide sufficient insightful results, and RNNs would require a time-series dataset which is not optimal as discussed in the section above. Therefore, approaches that will be explored feature GANs and adversarial autoencoders.

5 Realisation

Following the Creative Technology Design Process, this chapter represents the realisation phase of the design process. In this section of the report, the process of the implementation is documented, starting with the tools used, dataset creation, the choice of model and any alterations to the initial planning will be discussed.

5.1 Tools

For this project, the high-level programming language Python was used. Among data scientists, Python is popular due to its simplicity, readability and versatility. It has a vast collection of machine learning and data science libraries, such as PyTorch, TensorFlow, Keras and Scikit-learn. These libraries offer a wide range of algorithms, tools and frameworks for developing machine-learning solutions.

As for the choice of machine learning frameworks, PyTorch was chosen because of its flexibility in implementation. It is easier to make code modifications in the model and is more suited to atypical or exotic machine-learning approaches than other frameworks.

Scikit-learn was also used as it supports many useful tools for statistical modelling and other machine learning model. This library was mostly used for data pre-processing and data analysis.

The machine used to run these experiments was an 8th generation Intel Core i5-8250U CPU with 8GB RAM and Intel UHD Graphics 620.

5.2 Generative Adversarial Network

5.2.1 GANs for anomaly detection

As briefly explained in section 4.2.2, GANs are a suitable approach to generating new data samples based on a given dataset. To reiterate, a GAN is comprised of two networks, a generator and a discriminator. The generator receives a dataset, after which its purpose is to generate new data as close to the input data. The discriminator receives either real data, sampled from the actual data distribution or a synthetic, 'fake' sample from the generator. The discriminator then tries to differentiate the generated data from the real data. These two networks are trained in an adversarial manner, where each network becomes more skilled at its respective task over iterations of the training process. For generative purposes, as mentioned in section 2.7.2, the discriminator is discarded, whereafter a generator remains which can create new and truthful images based on its input.

This GAN architecture can also be used for anomaly detection. Once the GAN has been trained on non-dumping images, it can be used for anomaly detection by evaluating the likelihood of an instance being generated from the normal data [70]. Instances with low likelihood scores are considered to be anomalies, as they are considered to be significantly different from the normal data. This approach has been shown to be effective in a number of studies, with results that are on par with or superior to traditional anomaly detection methods in some cases, as discussed by Deecke et al. [70].

5.2.2 Vanilla GAN implementation

The implementation process started with the development of a vanilla GAN, which is the simplest form of the model, using the tools described in section 5.1. The GAN was constructed using sequential dense layers. To test the generative capabilities of the GAN, the model was trained on the MNIST handwritten digits dataset. This is a benchmark dataset in computer vision containing 10.000 scanned handwritten digits, and in this case, will be used to train the model. The GAN will take this dataset as an input and will output new handwritten digits, where it will try to create new data samples as closely as possible to the handwritten digits.

Initially, the vanilla GAN showed poor and noisy results which did not seem to converge over more epochs. A batch normalization layer was added, which improves the stability of the network. This normalizes the inputs for each layer of the network, preventing too small or large value, improving performance in the generator and discriminator. The image below is the result of a run of 200 epochs with a batch size of 32.

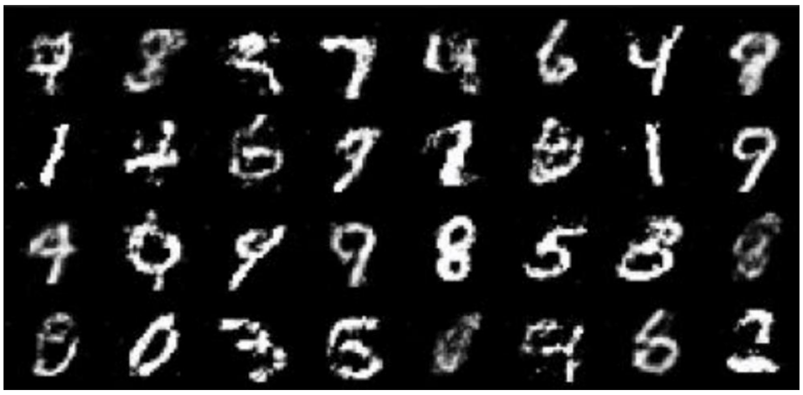

Figure 10 Vanilla GAN, 200 epochs, batch size of 32

These outputs were an improvement upon the first iterations, but the results were not satisfactory for a simple dataset like the MNIST dataset. The batch size was then increased to 64, and below is the result for 400 epochs:

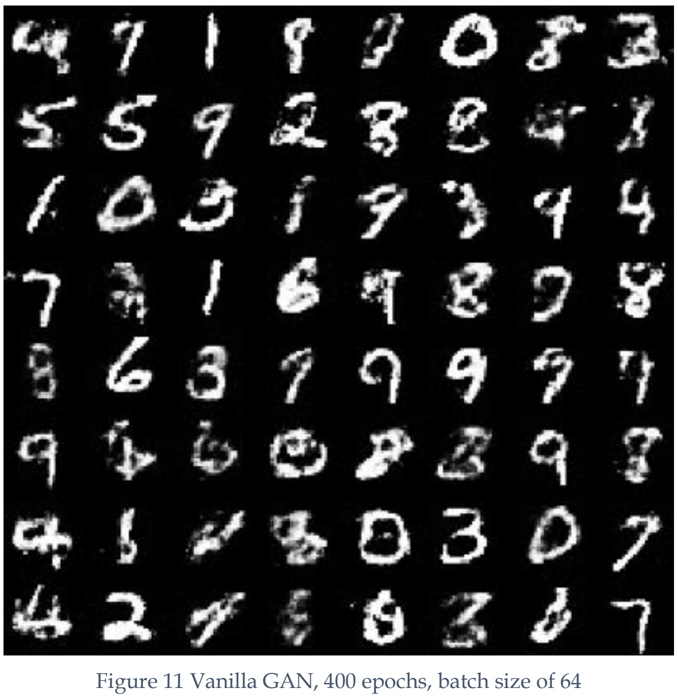

Figure 11 Vanilla GAN, 400 epochs, batch size of 64

The results obtained from the vanilla GAN still display a significant level of noise, which is inadequate for the desired purpose. Furthermore, in other runs of the same model, there was evidence for mode collapse. This is a phenomenon in GAN training where the generator creates a specific subset of samples, as opposed to a variety of unique samples, as it has learned that that specific sample is successful at 'fooling' the discriminator and thus continues only generating that type. This is a common problem in GAN training, and the next section will explore a variation of the vanilla GAN that remedies this problem.

5.2.3 WGAN with gradient penalty

After the insufficient results from the vanilla GAN, a WGAN was implemented. The WGAN, or Wasserstein loss GAN, implemented here uses convolutional layers instead of dense layers as seen in the vanilla GAN. One of the key differences between a WGAN and a traditional GAN is the loss function used to evaluate the performance of the generator and discriminator. In a traditional GAN, the loss function is a binary cross-entropy, which is based on the idea of minimizing the difference between the generated data and real data. In a WGAN, as proposed by Arjovsky et al. [71], the loss function is based on the Wasserstein distance, which measures the difference between the generated and real data distributions.

The use of the Wasserstein distance in WGANs has several advantages over traditional GANs. For example, the Wasserstein distance is a well-defined and continuous metric, which makes it easier to optimize the generator and discriminator during training [71]. Additionally, the Wasserstein distance provides a more stable training process, which may lead to improved results.

In addition to this, gradient penalty was included in the WGAN, a concept introduced by Gulrajani et al. [72]. The main idea behind gradient penalty is to add a regularization term to the loss function that punishes the discriminator for having a high gradient magnitude. During training, it is possible for the discriminator to become too powerful, resulting in a highly fluctuating loss function that is difficult to optimize. This can lead to training instability and slow convergence. The gradient penalty term helps to alleviate this problem by penalizing the discriminator for having a high gradient magnitude [72]. The image below is the output of the model after 1500 epochs with a batch size of 64.

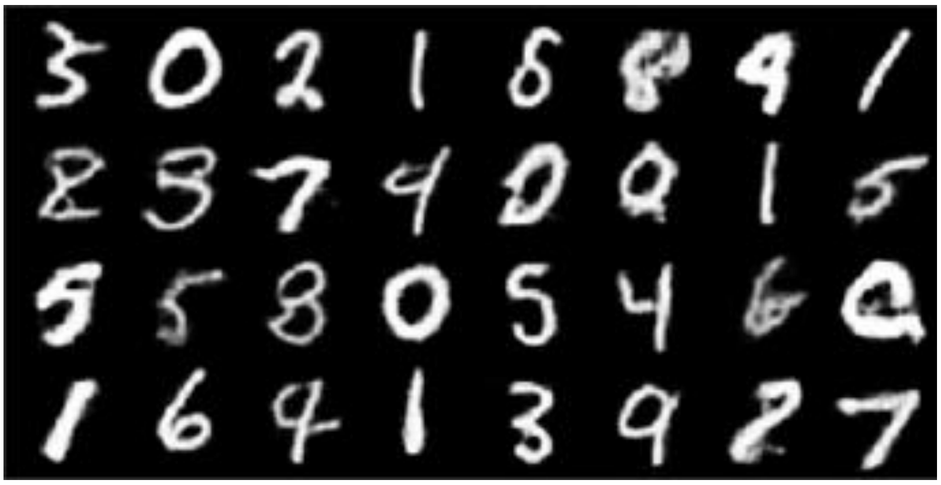

Figure 12 WGAN with gradient penalty, 1500 epochs, batch size 64

Visually, this is an improvement over the experimental results from the vanilla GAN. The synthetic samples inhibit significantly less noise than the previous model, and over larger training periods, some of the samples can clearly be recognized as digits.

However, as the MNIST handwritten digit dataset features 28x28 pixel images and our dataset features 512x512 pixel images, the dumping dataset might pose a serious computational challenge for this model. The results obtained in Figure 12 were obtained after two days of training, and with results that are still unreliable and unrecognizable at times, this approach proved lacking for our purposes. The results raise concerns about whether a dataset with images almost 20 times in size will be an improvement in performance, as well as training times which were already long on this dataset.

5.3 Adversarial Autoencoder

After evaluating the results from the previous models, more research was done on what kind of anomaly detection model could be a better candidate for our purpose. The adversarial autoencoder was a fitting contender, as it allows for more manipulation of latent space for anomaly detection. To achieve this, the AAE must first learn the data distribution of the satellite images. This model can then learn high-level features of the structure of the images or the dimensions that are most informative in an observation out of the multi-dimensional data distribution. Conversely, the model can also learn lower-level features that contribute less to the overall image structure, such as the texture of lands, pastures, etc.. As dumping disturbs the 'normal' land textures, the aim is to detect changes in the satellite imagery texture. Because an AAE outputs the latent space, this allows for great flexibility in detecting anomalous images. Furthermore, in contrast to certain GAN models which also output the latent space, the AAE does so without the need for full GAN training. This immensely speeds up the process of tuning hyperparameters in the model and running it.

Furthermore, the GAN generator would likely map the processed dumping images to the dense areas of the latent space distribution, which would be problematic. The dumping images would then be considered probable images (thus not anomalous), and low probability-based anomaly identification would no longer be possible. On the other hand, this phenomenon would likely happen with an AAE model as well, however, AAE models allow for greater control over the latent space, which could prove beneficial in the process of interpreting and manipulating the output of the model.

5.3.1 New dataset

As the initial planning was to use a CNN or GAN model, large amounts of data were a requirement for reliable and adequate results. For an AAE, the dataset might look different from the dataset created earlier, which features two large categories of dumping vs. non-dumping satellite imagery.

First, a dataset was created featuring ~10.000 satellite images of buildings and ~5.000 images of pastures, which were obtained from the Google Earth Engine. This dataset will be used for training purposes only, letting the model learn about the type of imagery, textures of the land, features of buildings etc.

Then, a subset of the dumping dataset was created, which features only the images that depict this dumping most clearly. This dataset will be used to infer the encodings. From each location in the dumping dataset, one representative image was selected that inhibits the most apparent instance of dumping from among the 15 images at that location. The resulting dataset of the most apparent dumping consisted of ~1.000 images, with each image corresponding to one of the ~1.000 dumping locations. When distinguishing between the dumping and non-dumping images, this dataset helps the model recognize the most prevalent examples of dumping, eliminating most edge cases which could potentially lead to ambiguity or interfere with the model's ability to accurately distinguish between normal and anomalous instances. Due to the nature and architecture of the AAE, ~1.000 anomalous images are sufficient. In the same manner, a subset of non-dumping images was created for the same purpose as the images with the most apparent dumping.

However, this non-dumping dataset must meet certain requirements for the model to be reliable and to avoid overfitting. Firstly, the non-dumping data must come from the same data distribution as the dumping dataset, thus the same type of area or land patch that the dumping images were captured from. Secondly, the non-dumping images must contain a roughly similar landscape as the dumping images. The dumping imagery that was captured features mostly rural or agricultural areas, meaning that the non-dumping image dataset must also display this type of environment. Thirdly, it is imperative to ensure that the images used for training are not taken at the exact same location as the images depicting instances of dumping. Utilizing identical locations and framing for both the dumping and non-dumping images could result in a dataset where the model is exposed to both images of a location with and without evidence of dumping. This could result in overfitting, as the model would become overly familiar with the features present in both the dumping and non-dumping images.

In order to adhere to the specified requirements, I chose to process the original satellite images by re-cropping them. As previously mentioned, the presence of dumping in an image can typically be identified at its centre. Consequently, by extracting 512x512 segments from the corners of the original 6663x8192 pixel image, new non-dumping images can be generated that maintain the same data distribution and approximate landscape as the dumping image, while avoiding the use of the same location. Therefore, using the same bulk cropping script as before, sections from the bottom right and top left were cropped in order to create the new non-dumping dataset, as depicted by the red 512x512 pixel squares in Figure 13. This resulted in a ~4.700 image dataset.

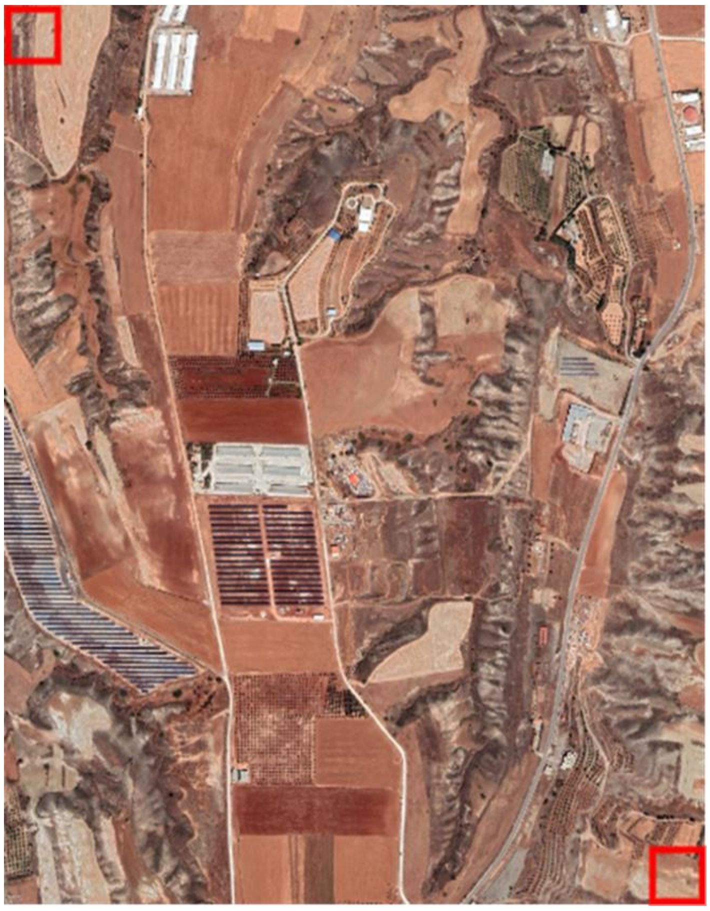
Figure 13 Bottom right and top left re-cropping of the original image

5.3.2 Model implementation

Thereafter, the AAE model was developed according to the architecture displayed in Figure 4 using the tools listed in section 5.1. The model was trained on the ~10.000 building imagery and ~5.000 pasture imagery datasets. After training, the encoder can now be used to infer the encodings, or embeddings, from the new dumping and non-dumping datasets. The trained encoder can now generate truthful and representative embeddings for each image in the dataset.

The embeddings are created for each of the dumping and non-dumping images following a multivariate Gaussian distribution. Each image is expressed in 512 embeddings of size 256 numbers with every embedding group being a multivariate normal distribution. Now that these 512 embeddings are created for each of the ~1.000 dumping images and ~4.700 non-dumping images, the goal is to identify differences between the embeddings and techniques to discriminate between them.

5.4 Processing embeddings in latent space

Having obtained the embeddings for both dumping and non-dumping images, various methodologies can now be used to differentiate between these two classes within the latent space. In this section, Principal Component Analysis was applied to the embeddings, reduced in dimensionality and clustered using the k-means algorithm and the t-SNE technique.

5.4.1 Principal Component Analysis

Principal Component Analysis (PCA) is a widely-used mathematical technique in the field of machine learning and computer vision. It is a dimensionality reduction method that aims to transform high-dimensional data into a low-dimensional representation while preserving as much of the variance in the data as possible [73]. The objective of PCA is to find the most significant features, referred to as "principal components", that capture the greatest variation in the data.

In the context of the embeddings, PCA can be used to reduce the dimensionality of the embeddings, making it easier to visualise and differentiate between the dumping and non-dumping embeddings. By applying PCA to the embeddings, the number of dimensions is reduced, while preserving the relationships between the data points. This can reveal patterns or clusters in the data that correspond to the different categories. The resulting visualization can be used to determine which features or components in the lower-dimensional space are most useful for discrimination between the two categories.

After the PCA has been applied to the embeddings, the plan is to cluster and visualize them in 2-dimensional space using the k-means clustering algorithm. The k-means algorithm itself will be explained in the next section, but the required data input shape for k-means is `(n_samples, n_features)`, where `n_samples` is the number of samples, and `n_features` is the number of features in the input data. The number of features must be consistent across all samples. Therefore, we must process the embeddings such that it adheres to this data shape.

As mentioned before, each image is now represented as 512 embeddings of size 256 numbers. The goal is now to see which of the 512 embeddings correspond to depictions of dumping in an image. We want to apply PCA to each embedding separately, to compress them. To do this, first, we must create embedding groups. This means that, for all of the 512 embeddings, we want to gather the 256-dimensional vector for each embedding over all of the images of one category. The embedding groups (in this case, the dumping embedding groups) are created following the pseudocode below. The code returns 512 groups corresponding to each embedding group.

```
Initialize an empty list called groups
For each of the 512 embeddings:
    Initialize an empty list called observations
    For each image in embeddings_dumping_images:
        Retrieve embedding for the current image and store in a variable called embedding
        Add embedding to the observations list
    Add the observations list to the groups list
Return groups
```

Figure 14 Pseudocode for creating the (dumping) embedding groups

This results in a multi-dimensional array with the following shape:

$$512 \times \#\, of\ images \times 256$$

Next, PCA was applied to each of the embedding groups. The algorithm iterated over the embedding groups and applied PCA with 4 components, whereafter the resulting shape is then:

$$512 \times \#\, of\ images \times 4$$

Then, as we need two dimensions instead of three for the next step (k-means algorithm), the dimensionality is reduced further by flattening the embeddings. Flattening is defined as reducing a multi-dimensional array into a one-dimensional array. In this case, we need to reduce the first (512) and third (4) dimensions into one dimension (512 x 4 = 2048). This was done using the `reshape()` function provided by NumPy as can be seen in Figure 15. Here, the `reduced_embeddings` variables represent the embeddings after PCA.

```
flattened_embeddings_d =
reduced_embeddings_d.reshape(reduced_embeddings_d.shape[1], -1)
flattened_embeddings_nd =
reduced_embeddings_nd.reshape(reduced_embeddings_nd.shape[1], -1)
```

Figure 15 Flattening embeddings code

The resulting shape is now:

$$\#\, of\ images \times 2048$$

This means that each image is expressed by a 2048-dimensional vector. This is done for both the dumping and dumping embeddings and the results are combined using NumPy's `concatenate()` function for n-dimensional arrays. The final shape of the reduced and concatenated embeddings is now:

$$(\#\, of\ dumping\ images + \#\, of\ non\_dumping\ images) \times 2048$$

Now, the data is in the form of `(n_samples, n_features)`, where the number of dumping and non-dumping images combined is the number of samples and each one is represented by a 2048-dimensional vector, the number of features. The data is now significantly reduced in dimension and adheres to the input data shape for the next step, the k-means algorithm.

5.4.2 K-means clustering algorithm

The k-means algorithm is a clustering algorithm commonly used in machine learning and computer vision. The goal of the k-means algorithm is to divide a set of data points into a specified number of clusters (k) in such a way that the points in each cluster are as close to each other as possible, and as far away from the points in other clusters as possible. The k-means algorithm works by initializing k random centroids and then iteratively assigning each data point to the nearest centroid, updating the centroids to be the mean of all the points assigned to them, and repeating these steps until convergence and the data is clustered [74].

The k-means algorithm can be used to group the embeddings into clusters based on their similarity. By consequently observing the resulting cluster, we can identify clusters that correspond to the different dumping and non-dumping categories. The number of clusters is a hyperparameter in this algorithm, initially k = 2, so two clusters were chosen corresponding to the two categories. Scikit-Learn allows us to initialize the centroids using k arrays. The means for the two embedding categories were computed and used for the initial centroid positions. As mentioned before, this is typically done randomly, however, as we can already calculate the means, this allows us to hint the algorithm towards the two clusters.

Next, the k-means was conducted on the concatenated embeddings of the dataset. As the embeddings have been processed such that it is in the form of `(n_samples, n_features)`, the concatenated set of embeddings can be used as is. In Figure 16, the implementation is shown. Scikit-Learn’s `KMeans()` function is used here, the amount of clusters is specified, the means of the dumping and non-dumping clusters are used for the centroid initialization and the k-means is conducted on the concatenated set of embeddings. Furthermore, the variable `y_kmeans` stores the cluster assignments for each data point and `centers` stores the cluster centers, which can now be used for visualization.

```
kmeans = KMeans(n_clusters=2, init=[d_means, nd_means]).fit(embeddings_concat)

y_kmeans = kmeans.predict(embeddings_concat)

centers = kmeans.cluster_centers_
```

Figure 16 k-means clustering implementation

The resulting scatterplot can be seen below. In the graph, the two clusters are displayed in a scatterplot, where the red stars are the two centroids.

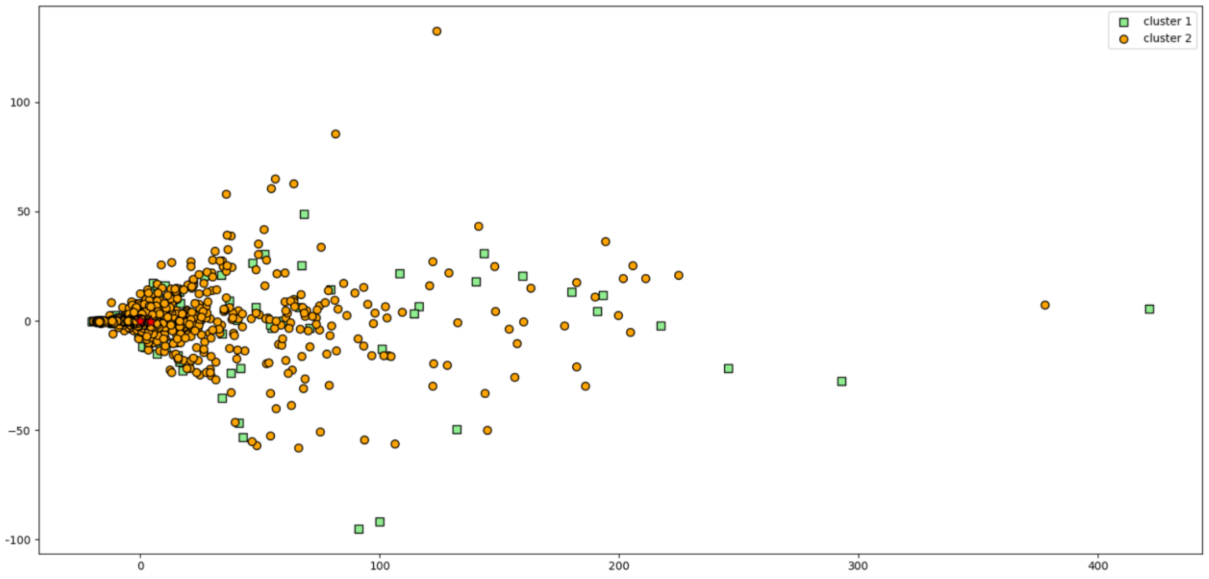

Figure 17 k-means clustering scatterplot with k=2

As can be observed in the plot, there is no clear separation between the two categories. Upon further inspection, it became clear that of the ~1.000 dumping embeddings, 94% of the data points belonged to cluster 2 (yellow), however, from the ~4.700 non-dumping data points, 93% also belonged to cluster 2. Ideally, we want a more clear separation, not two clusters where a significant majority of all data points belong to one cluster.

As mentioned before, the k argument in k-means is a hyperparameter, therefore it can be tweaked to obtain different results. There is a method to determine the optimal number of clusters for the k-means algorithm. A popular technique is the 'elbow method'. This concept uses inertia, which is a measure of the sum of squared distances between each data point and the centroid of its assigned cluster. In the elbow method, the inertia of the clusters is plotted as a function of the number of clusters. Then, the elbow point is selected in the plot as the optimal number of clusters in the k-means. The elbow point is characterized by a notable decrease in the rate of change of inertia, indicating that further increasing the number of clusters beyond that point would not result in a significant improvement in the quality of the clustering. In Figure 18, the elbow plot can be seen, however, there is no apparent elbow or significant drop-off in the inertia. This may be an indicator that the data is not well-separable with this technique.

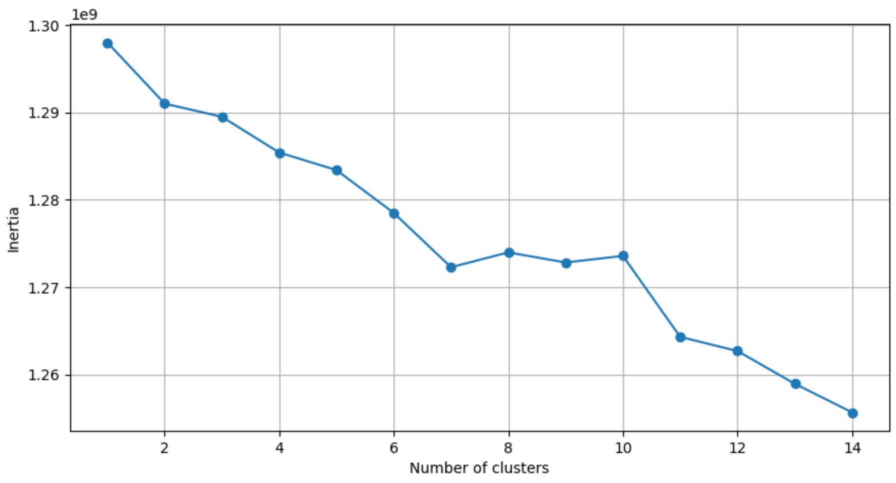

Figure 18 Elbow plot

Another method that we can use to get the optimal number of clusters is by using the silhouette score. This is a measure of the quality of the clustering in k-means using the Euclidean distance. It provides a measure of how well each data point is assigned to its own cluster, compared to other clusters. The silhouette score ranges from -1 to 1, with a score of 1 indicating a strong, well-defined cluster structure, and a score of -1 indicating a poor clustering structure. A higher silhouette score should mean that the data is more well-clustered with the corresponding k-value. A graph showing the silhouette scores for a range of numbers of clusters can be seen in Figure 19.

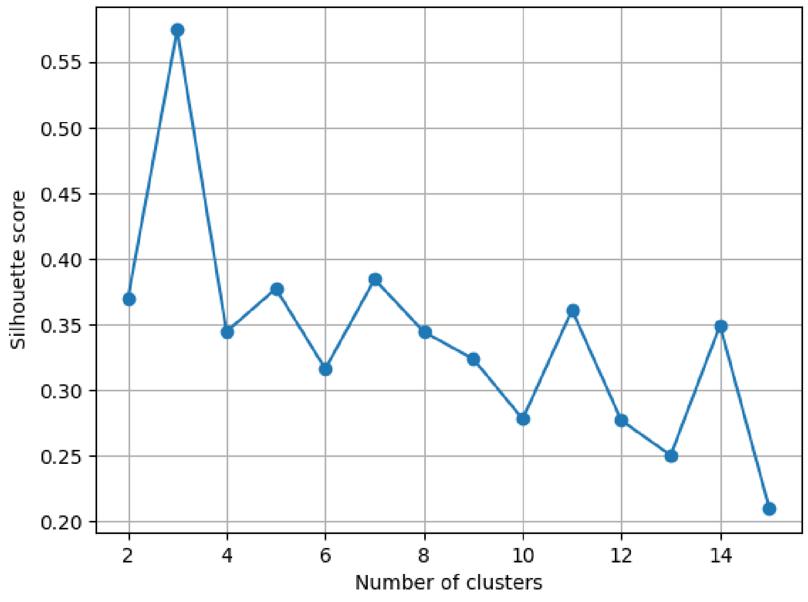

Figure 19 Silhouette score plot

From this plot, we can see that the optimal number of clusters is 3. Previously, the means of the two embedding categories were used for centroid initialization. However, as we now have more than two clusters, this is no longer possible. Instead, the k-means++ algorithm is used, which is an improved technique on centroid initialization over random initialization. k-means++ is also built into the Scikit-Learn module and can simply be implemented like so: `KMeans(n_clusters=3, init='kmeans++')`. The k-means scatterplot with k=3 can be seen in Figure 20 below.

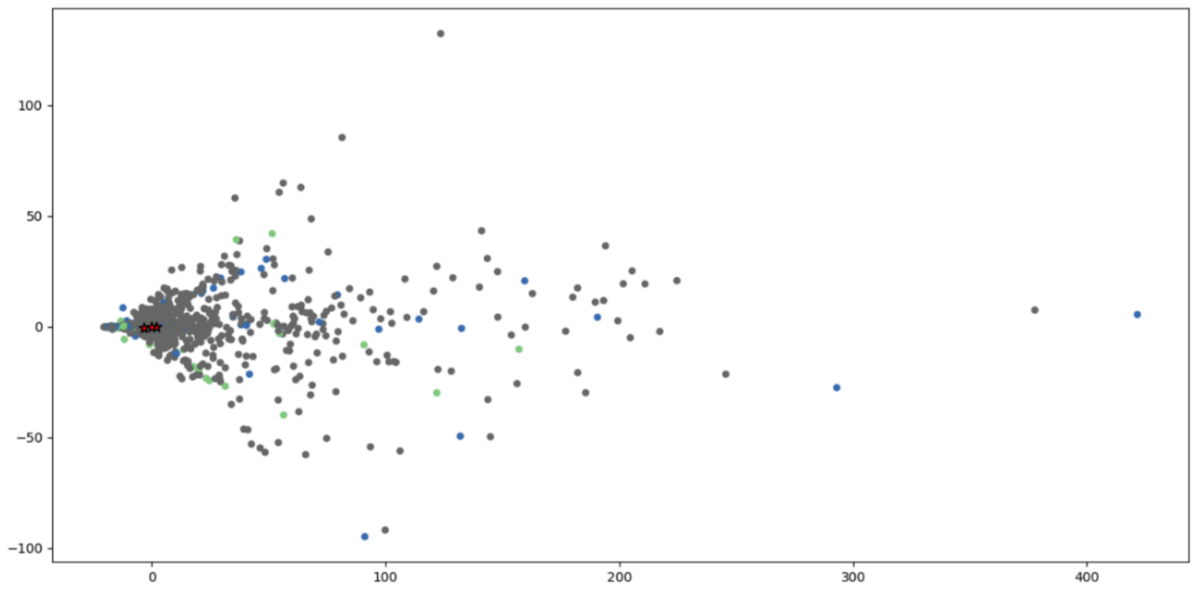

Figure 20 k-means clustering with k=3

It can be observed that the data is still not well-separated, the cluster centres were situated in close vicinity to one another, the data points do not show clear separation and the majority of the data points are assigned to the same single cluster. More experiments were conducted with k-values 4, 6, 8 and 10, and the resulting visualizations can be found in Appendix C. These results did not show significant improvement in the clustering, as hypothesized by the results of the elbow plot.

5.4.3 t-SNE

t-SNE (t-distribution Stochastic Neighbourhood Embedding) is an alternative dimensionality reduction technique used for the visualization of high-dimensional datasets. This method is based on minimizing the divergence between two probability distributions. One distribution represents the high dimensional data points and the other the low dimensional map. The low-dimensional map is randomly initialized, and through an iterative optimization process, point locations are updated to group similar data points and separate different data points [75].

The resulting low-dimensional representation, typically 2 or 3 dimensions, can reveal patterns and relationships in the data that may not be apparent in higher-dimensional space. In particular, t-SNE has proven useful in exploring and visualizing complex datasets where traditional visualization techniques may not work optimally. For instance, PCA is highly affected by outliers in the dataset, whereas t-SNE can manage outliers more robustly [75]. Therefore, experiments were done with t-SNE, as the k-means algorithm did not show well-separated clusters, independent of the number of clusters.

When applying t-SNE on the concatenated sets of embeddings, 2 components were chosen, as we want to map the clusters in 2D space. Scikit-Learn provides a function, `TSNE()`, that was used for this implementation. The exact lines used can be seen in the figure below. The `fit_transform()` function was used on the data as it can scale the data and also learn the scaling parameters of the data.

Line
```
tsne = TSNE(n_components=2)

tsne_result = tsne.fit_transform(embeddings_concat)
```

Figure 21 t-SNE implementation code

A scatterplot is the resulting clusters can be seen in Figure 22. This result is less sporadic in its clustering, in contrast to the k-means plot. There is still overlap, the two clusters both populate the centre of the plot. However, on the outside ring, the blue cluster dominates, perhaps proving effective for a circular classification boundary. The blue cluster represents the dumping embeddings, while the green cluster represents the non-dumping embeddings.

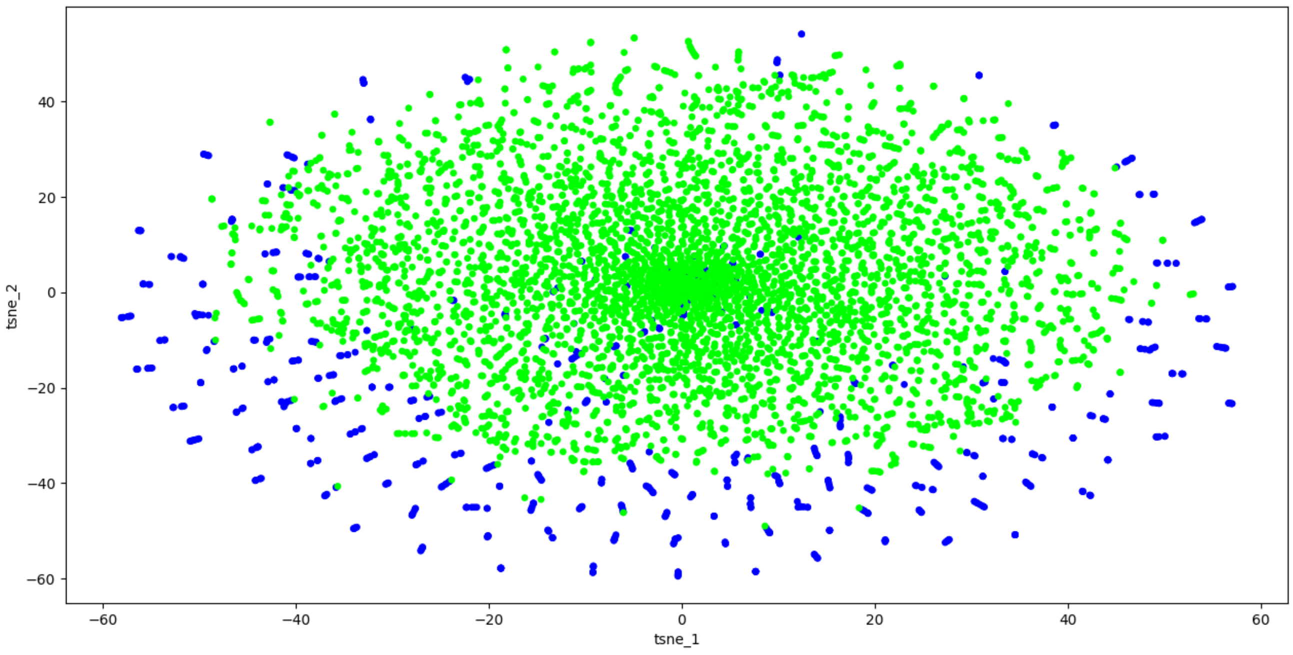

Figure 22 t-SNE scatterplot

5.5 Alterations to initial planning

During the background research phase of the project, the preliminary plan was to train a CNN model with a YOLO architecture. The dataset did, however, not allow for the data requirements for YOLO training, so an alternative model was to be selected. This also meant that the use of synthetic data was no longer relevant. Next, time-series LSTM and anomaly detection models were considered, where anomaly detection seemed to be the better approach to this problem. Experiments were done on a vanilla GAN, WGAN and WGAN-GP, however, the plan was thereafter altered again to go forward with an AAE model.

6 Evaluation

In this chapter, we evaluate the performance of the AAE model, compare it to other approaches, and discuss its generalizability. We also analyse the strengths and weaknesses of the AAE model in order to gain insights into the effectiveness of this approach for detecting dumping. Through this evaluation, the goal is to contribute to the development of more effective and efficient methods for detecting illegal dumping.

6.1 AAE performance

The performance of the AAE was evaluated by visualizing the embedding generated by the encoder in the model, using dimensionality reduction and clustering techniques k-means and t-SNE.

K-means clustering was initially used to separate the embeddings into two clusters corresponding to dumping and non-dumping sites. However, this technique showed poor results, as the clustering did not effectively separate the two classes, with some dumping sites being clustered with the non-dumping sites and vice versa. Thereafter, more experiments were done with a range of k-clusters, with similar performance. This poor separation could be due to the fact that the AAE model generates continuous embeddings, which are difficult to cluster into discrete classes.

In order to visualize the embeddings in a more effective manner, t-SNE was used to reduce the dimensionality of the cluster into two dimensions. t-SNE is a powerful technique that can provide insights where traditional dimensionality reduction techniques fail. The t-SNE visualization showed more promising results, as there was significantly less overlap between the two clusters. Due to the circular mapping of the t-SNE clusters, an oval-shaped boundary can be used as a decision boundary for the classification of dumping vs. non-dumping, with some margin of error due to the overlapping of some data points.

Overall, while the k-means clustering visualization showed poor performance, the t-SNE visualization indicated that the AAE model was effective to some degree in generating embeddings that separate dumping and non-dumping sites. However, the fact that some overlap still existed between the clusters suggests that the AAE model could benefit from further refinement. It is also possible that other methods for visualizing and analysing the embeddings (further discussed in chapter 8), may yield further insights into the performance of the AAE model. Furthermore, it would be interesting to experiment with larger datasets, to see how it affects the performance in the clustering.

6.1.1 Images on decision boundary

Earlier, an oval-shaped decision boundary was proposed, but to test this, it might be useful to inspect some images that are on this boundary. From the scatterplot, random data points were manually selected that are visually in between the dumping and non-dumping clusters. In Figure 23, an image can be seen that was identified by the t-SNE tool to be non-dumping. However, upon inspecting the image, it is clear to see why the t-SNE algorithm was irresolute and mapped this image close to the dumping images. There is a cloud in the top centre of the image, with a colour and texture that could also represent dumping. This cloud can be very similar to some of the dumping sites present in the dataset, however, it correctly classified this as non-dumping, although the image is equivocal to the algorithm to a certain degree.

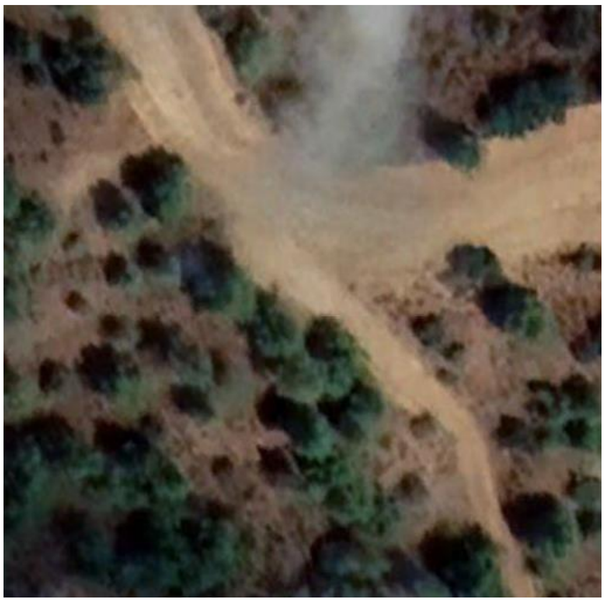
Figure 23 Non-dumping image on decision boundary (cloud)

In Figure 24, we can see another image that was in the decision boundary. Similarly to the previous image, this image also does not contain dumping, however, this time the image was clustered with the dumping images. Likely, the algorithm had difficulty categorizing this image as it contains a body of water, with reflections that look familiar to dumping on the satellite images. The white disturbances make it seem much like scattered waste, but after analysing the other acquisition points for this location, it can be said with certainty that this is a lake and not a dumping site.

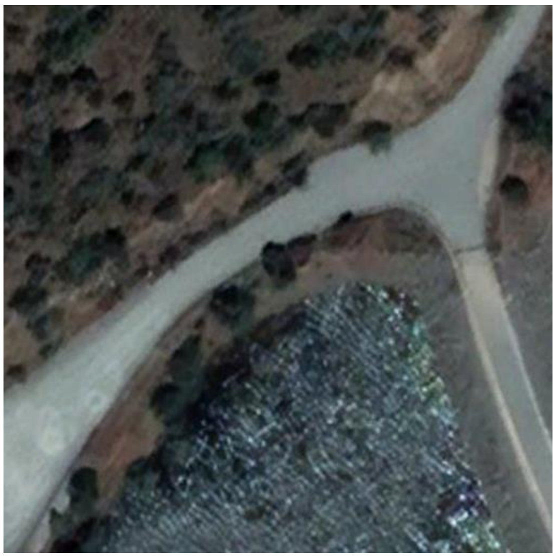

Figure 24 Non-dumping image on decision boundary (body of water)

Lastly, Figure 25 is an image on the decision boundary grouped with the non-dumping images, while it was, in fact, a dumping image. The image is low-resolution and it is not immediately discernible whether there is dumping in the image or not. In the context of the other images of this location in time-series, this image was classified as a dumping image, however, this may not be as unequivocal when viewed in isolation. There are small white dots to be seen in the image which are the dumping instance, but it is an image in which it is difficult to say definitively. The resolution of the image is low, and the quality of the image itself is degraded. The algorithm's classification of the image near the dumping and non-dumping boundary reflects the inherent difficulty in differentiating between these two categories on edge cases like the images discussed here.

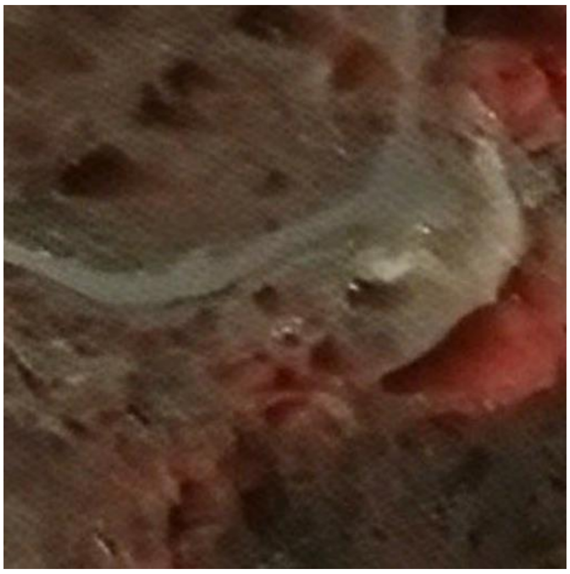

Figure 25 Dumping image on decision boundary (lacking resolution)

6.2 Comparison to other approaches

6.2.1 CNN

CNNs are a popular approach for image classification and object detection tasks and are often used in remote sensing and environmental applications. In contrast, the AAE is a more novel and less researched approach to image classification.

One advantage of CNNs is that they are well-suited to image classification tasks, as they can effectively learn and extract features from images. CNNs have been proven effective in many studies for similar applications and are a well-researched topic. However, CNNs often require large amounts of labelled data for training, which can be a limiting factor in some applications.

An advantage of the AAE model is that it is able to learn a more compact and abstract representation of the images than CNNs, which can be useful in cases where storage and computation resources are limited, for example in less fortunate communities that deal with illegal dumping. The embeddings generated by the AAE model can also be visualized and analysed in a variety of ways, which can provide insights into the performance of the model and the underlying structure of the data. It also allows for more control over the latent space from the model, making the model more flexible.

However, it should be noted that the AAE model also has some limitations. For example, the AAE model may be more difficult to train and optimize than CNNs and may require more specialized knowledge and expertise. This approach requires more time to finetune the model and interpret its results.

6.2.2 GAN

Another popular approach that was discussed is anomaly detection, a research topic where GANs are widely deployed. One advantage of GANs is that they can generate highly realistic images by learning to mimic the distribution of the training data. However, GANs are often more difficult to train and stabilize than AAEs and can suffer from mode collapse (as seen in the vanilla GAN) and other issues.

Comparatively, the AAE is more stable during training and is easier to train than the GAN. AAEs and GANs both similarly allow for control over latent space, however, the AAE does so without the need for full GAN training. GANs are computationally heavy models as compared to AAE. The GAN model would likely have worked as effectively as the AAE, but due to the fact that GANs are heavy models that can show issues in training and the long training phase, an AAE would be the better choice.

7 Conclusion

The goal of this project was to automatically detect illegal dumping from high-resolution satellite imagery. Illegal dumping is a pervasive issue that has a significant negative impact on the environment, wildlife, and human health. Dumping can contaminate the environment, leading to degradation in soil, surface water and air quality. Furthermore, the presence of illegal waste not only disrupts the natural balance of ecosystems but also poses severe health risks to local populations. In addition to ecological consequences, illegal dumping also has significant implications for human health. The toxic substances present in waste materials can contaminate groundwater, which is a primary source of drinking water for many communities. Inhaling or ingesting these toxins can cause serious health problems.

It is important to develop an effective method for detecting illegal dumping to minimize its ecological and public health impacts. Currently, this is mostly done manually, which is slow, expensive and inaccurate. Therefore, an automated solution was needed to detect illegal dumping in near real-time.

7.1 Research findings

The aim of this project was to develop a system for the detection of illegal dumping in high-resolution satellite imagery. Initially, a CNN-based model was considered for this purpose. However, it was later decided to explore alternative approaches, with a focus on anomaly detection. The chosen approach involved the implementation of a vanilla GAN on the MNIST dataset, which yielded poor results. The generated digits displayed a significant level of noise and little resemblance to the input dataset.

The experiment was then repeated with a WGAN, as well as a WGAN with gradient penalty (WGAN-GP), which resulted in significantly improved outcomes. The digits were less noisy and samples could be recognized as digits. However, there were concerns about the performance of these models on larger images from the dataset, prompting the decision to pivot towards an AAE approach.

The AAE model was implemented, trained, and deployed. Subsequently, embeddings were generated for a new dataset, and these embeddings were analysed in the latent space. Several techniques were employed to improve the separation of the embeddings, including PCA and clustering using k-means. However, poor separation was observed, prompting the application of t-SNE, which resulted in a better outcome. There was less overlap in the clusters and the clusters were more clearly defined compared to the k-means clusters. In the t-SNE scatterplot, a circular classification boundary might be useful in determining which embeddings correspond with dumping and no-dumping images.

However, there was still an overlap in the clusters generated by both the k-means and t-SNE clustering algorithms. This could mean that the data is not well-separable or well-suited for clustering. This would also indicate that, superficially, the patterns that define whether an image contains dumping or not is not as clear as initially assumed. Therefore, a more complex analysis would be needed, or a different approach to the pre-processing of the embeddings.

Approach	Hyperparameters
Vanilla GAN	- Batch size 32, 200 epochs - Batch size 64, 400 epochs
WGAN-GP	- Batch size 64, 1500 epochs
AAE k-means	- k-means clustering with 2, 3, 4, 6, 8 and 10 clusters
AAE t-SNE	- t-SNE in 2D space

Figure 26 Brief overview of the experiments performed in chapter 5

7.2 Answer to research questions

In this section, the research questions posed in section 1.1 will be addressed. The sub-questions will be addressed first, followed by an examination of the main research question.

7.2.1 Sub-questions

To offer a concluding answer to the main research question, it is essential to address the sub-questions first. The sub-questions are formulated as follows:

SQ1: *What deep learning framework is optimal for dumping detection from satellite imagery?*

Detection of illegal dumping from satellite images is an important problem that requires an effective model. In this section, I argue that AAEs are a suitable modelling approach for this problem based on experiments. AAEs are trained to learn a low-dimensional representation of input images called embeddings. These embeddings capture the important characteristics of an image, allowing for more effective anomaly detection.

In the case of illegal dumping detection, specific embeddings can be used to determine whether an image contains evidence of illegal dumping. By training the AAE on a dataset of both normal and anomalous satellite images, it can learn to produce embeddings that accurately capture the differences between the two types of images. These embeddings can then be used for classification after processing and analysis of the data.

One advantage of using AAEs for this problem is that they are less prone to overfitting than traditional CNNs. CNNs can sometimes learn to recognize specific visual patterns that are only present in the training data, leading to overfitting and poor generalization to new data. Furthermore, AAEs actually learn subtle features as well, such as land textures. This helps the model, as dumping also disturbs these lower-level features.

Concluding, CNNs have been proven to work well on detection problems such as satellite detection, however, AAEs boast features that may improve the quality of detection and allow for more control over the model and require significantly less computational cost.

SQ2: *What satellite data will be useful for training and as input for the deep learning model?*

There were quite some ambivalent images in the original dumping dataset, creating challenging cases for the model and requiring more pre-processing of the images. For example, images could contain artefacts of the Google Earth Engine UI as a result of exporting. A more suitable dataset could be a dataset containing merely the raw satellite images. Furthermore, there are numerous examples of the image quality being degraded such as in Figure 25, or exposure errors where the distortions in the image were grave enough for it to be unusable. Moreover, many images were discarded as unusable due to a cloud or strong shadows obstructing the dumping site, however, this is of course natural in satellite datasets.

Lastly, the content of the images was also ambiguous at times. If images were difficult to categorize for human interpreters, then this challenge is amplified for a machine-learning model. As mentioned, in many images it was nearly impossible to see whether the image contains dumping or not, which likely impacted the model's performance. It is noteworthy that, if applied to real cases of dumping detection in municipalities, these nuances and vague edge cases will inevitably occur as well.

Therefore, the used dataset from Google Earth Engine is deemed insufficient for the purpose of exploring whether the AAE model can be used for illegal dumping detection. The images used produced embeddings which were meaningfully interpretable to a certain degree, but using a different higher-quality or paid satellite imagery illegal dumping dataset is recommended. This can minimize sensor errors, artefacts and ambiguous images and improve the reliability of the model.

7.2.2 Main question

With the sub-questions answered, the main research question of this thesis can now be answered.

> **RQ:** *How can a deep learning model be developed for the purpose of detecting illegal dumping sites from satellite imagery?*

Although experiments with CNNs for the purpose of detecting illegal dumping from satellite imagery have been successful, in this project it is also demonstrated how an AAE can be utilized for this purpose. Compared to CNNs, the proposed AAE-based approach is characterized by its capacity to learn a more efficient and interpretable latent space that can capture meaningful concepts relevant to the task at hand. In addition to its ability to learn effective embeddings, the proposed approach exhibits unique features that distinguish it from traditional CNNs and enhance its overall performance.

To approach this problem, a dataset must be created, pre-processed and annotated based on the desired target categories. From this dataset, a subset must be created displaying only the most apparent instances of dumping. The AAE must be developed and trained and given two datasets, one anomalous and one normal dataset, from which it infers embeddings corresponding to each image in the dataset. These embeddings can then be processed using techniques such as PCA, k-means and t-SNE to expose patterns in the data that can be used to differentiate between dumping and non-dumping satellite imagery.

Concluding, AAEs are a new and innovative approach to anomaly detection that combine concepts of multiple models in a novel manner. As dumping detection from satellite imagery using an AAE model is not yet researched, this state-of-the-art research proves the viability of this approach in this field. While further research in this area is needed, the current study provides evidence to support the effectiveness of utilizing AAE models for detecting anomalies in satellite imagery.

8 Discussion

In this chapter, reflection on the limitations of the research and exploring future directions for this work will be discussed. The limitations of the methodology and models used will be critically assessed. We will also discuss potential opportunities for future research and the implications of this work for the field of deep learning for dumping detection from satellite images. Ultimately, this discussion will aim to provide a transparent and truthful assessment of the research, while also highlighting the potential for future innovation and improvement.

8.1 Limitations

In this section, limitations encountered in the design process and implementation of the illegal dumping detection model will be discussed. Specifically, potential shortcomings in the research process will be identified and provide a discussion of the impact of these limitations on the reliability and generalizability of our findings. By acknowledging these limitations, the goal is to provide a more comprehensive understanding of the research.

The first limitation was the importance of iterative model selection in the project. I initially used a GAN model but found it difficult to train, computationally expensive, and produced poorly generated samples. By exploring other models, including the AAE, I found that the AAE was better suited for this task, producing more meaningful embeddings and better results than the GAN model. In retrospect, I should have pivoted towards the AAE sooner, emphasizing the importance of iterative model selection and experimentation in these kinds of machine learning projects. Moreover, in the end, the GAN models were only trained and tested on the MNIST images, rather than the actual dumping and non-dumping images from the dataset.

One limitation in the annotation process was the bias in the categorization of dumping vs. non-dumping. As the annotators had a time-series of all acquisition points of a location, the context may have given an advantage that the model did not have. For example, when presented with a series of clear dumping images, I was more inclined to classify a less clear dumping image as dumping as I was subconsciously aware that there is, in fact, dumping in that location. I have tried to negate this by being objective in the annotation for a single isolated image, but this bias has undoubtedly played a part in the annotation process.

Hardware limitations also affected parts of this project. In section 5.1, the tools and hardware used throughout this project were specified, however, especially when experimenting with GANs, I noticed that the training process was computationally costly. In addition to GANs being computationally heavy models, this could perhaps be solved by running the experiments on a faster CPU. Ultimately, this partly affected the realisation process as GANs were also deemed to be computationally too costly for this purpose (near real-time detection).

Time constraint was a limitation in this project. This led to a limited analysis of the produced embeddings by the AAE. Only nearing the end stages of this project, the embeddings were properly processed and visualized to potentially observe patterns. Had time not been a factor, further analysis of the embedding could have been done, potentially exposing patterns and specific embeddings leading to a meaningful classification of dumping vs. non-dumping.

Lastly, due to the fact that an AAE model has never been deployed for the specific task of dumping detection, in particular not with high-resolution satellite imagery, there were few references in literature as AAEs are a relatively new concept. At times, it was difficult acquiring relevant resources to develop this model and process its embeddings, as there is no state-of-the-art for this specific task.

8.2 Strengths and weaknesses

To neatly summarize certain strengths and weaknesses of the AAE approach I encountered, the following list is created:

Strengths

- *Unique approach*

  The use of an AAE model for this specific purpose has not been researched yet. This is a unique and novel approach, that will hopefully spark more research into AAE models.

- *Data efficiency*

  The AAE model can work with relatively small datasets, as it can learn to generate representative and truthful embeddings to be used for anomaly detection from a small set of examples. Furthermore, the AAE requires significantly less training time as compared to similar models.

- *Control over latent space*

  This approach allows for much control over the latent space from the model. This allows for more interpretability of the results in a more compressed and simplified representation of the data while preserving the most important features.

Weaknesses

- *Accessibility*

  The interpretation of the embeddings generated by the model requires experts with domain knowledge of deep learning and autoencoder models and an understanding of how to interpret latent space. This may limit the accessibility when deploying the model.

- *Sensitivity to hyperparameters*

  The model has a number of hyperparameters that must be chosen carefully in order to achieve good performance. These hyperparameters include the size and structure of the encoder and decoder networks, the length of the training phase, and the learning rates for the generator and discriminator. Tuning these hyperparameters can be a time-consuming and computationally expensive process.

- *Novelty*

  The fact that this is a relatively new approach may also be a limiting factor for the use of such models. There are few studies available in literature that explore the possibility of similar domains, meaning that there are few reference points when developing these models.

8.3 Future work

In this section, opportunities for improvement and future research are given. Given the scope of the project, along with the abovementioned limitations, parts of the AAE model and the resulting embeddings could be improved upon in the future.

8.3.1 Modelling

One promising area for exploration is the refinement and optimization of the AAE model. For instance, modifications to the AAE architecture, such as incorporating attention mechanisms [76] or other neural network modules, could potentially improve the performance of the model. This could lead to better embeddings being generated which capture the key features of the data more representatively.

Furthermore, an additional feature to potentially implement in the model would be to give a (limited) time-series for each image that it has to classify as an anomaly or not. As seen in the edge cases discussed in section 6.1.1, context is sometimes needed in images where it is difficult to categorize them. Providing the model with one or two images surrounding the image in the time-series may improve the accuracy of the anomaly detection.

8.3.2 Hyperparameters

The hyperparameters of the model could also be tuned more, potentially leading to different results. For example, in the AAE network, hyperparameters such as the batch size, learning rate, latent space size, adversarial loss weight, activation functions and the number of encoder and decoder layers could be finetuned for this specific purpose. Moreover, as the AAE learns to map data to a prior distribution, experiments can be done using different distributions for the data.

8.3.3 Generalizability

The generalizability of the model is a crucial aspect to be evaluated. One potential concern would be whether the AAE can effectively learn and generalize the hidden patterns of dumping in the context of different regions, land types and dumping given the limited size and diversity of the training dataset that was used.

The generalizability of a model can be affected by several factors. For instance, the complexity of the data, variability of the image features, dataset size and configuration of the model can all influence how well the model generalizes to other scenarios.

In this case, the model was trained on a specific dataset of satellite images displaying buildings and pastures. The model was then given the task to infer embeddings from a new dataset, namely the dumping dataset. This dataset displays different landscapes and features but is from the same data source (Google Earth) with the same data distribution. The results of the visualization are optimistic about a certain degree of generalizability of the model, but more research must be done using entirely different datasets with other data distributions. Furthermore, different model architectures and other configurations of hyperparameters must be explored to observe their impact on the performance of the model.

After the development of this model and analysing its generated embeddings for this specific dataset, it could be useful to test on different datasets. This could test the generalizability of the model further. For example, the model could be trained and deployed on more diverse ranges of (anomaly) datasets, such as:

- Medical imagery
- Abnormalities in network traffic
- X-ray security screening
- Manufacturing inspection
- Environmental monitoring (oil spills, deforestation)
- Autonomous vehicle vision (accidents, damage to roads)

8.3.4 Satellite imagery type

More research involving different satellite data such as rear-infrared and other hyperspectral satellite data must be done. This is due to the fact that different spectral bands can capture different types of information about the environment being imaged. For instance, some spectral bands may be sensitive to specific features that dumping inhibits in the satellite data, which can expose more hidden details of dumping. By including a wider range of spectral bands in the imagery, the model may be better able to detect anomalous features or patterns that might be missed with a more limited set of bands.

8.3.5 Alternative visualisation techniques

The visualization experiments conducted in this research mainly feature dimensionality reduction and clustering techniques, however, other visualization techniques could bring different insights into the hidden structures of the embeddings.

For example, saliency maps could be used, which are maps that highlight the areas of an image that are most important for the classification decision. This way, regions of the image most relevant the detecting anomalies can be visualized. Similarly, Grad-CAM (Gradient-weighted Class Activation Mapping) generates heatmaps over regions of the image that emphasize what the model is most sensitive to. These maps also visualize the most relevant areas for anomaly detection.

Additionally, Self-organizing maps (SOMs) are a type of neural network that can be used to create low-dimensional maps of high-dimensional data, which can be used to identify clusters and anomalies in the data. This method is optimized for high-dimensional data and can learn a mapping from the input data to the output map. This can be utilized to visualize the embeddings in the context of the original data, allowing for a better understanding of the underlying structure of the data. Likewise, Uniform Manifold Approximation and Projection for Dimension Reduction (UMAP), could be a useful tool in exposing relationships in the embeddings.

Lastly, in the process of dimensionality reduction, all techniques inevitably encounter some degree of information loss. Therefore, instead of mapping the data into two-dimensional space, three-dimensional space visualization should also be explored. This could preserve more of the original data, and lead to insights previously unobtainable in the lower dimensionality. For example, the t-SNE method allows for visualization in three-dimensional space, which could potentially reveal more of the underlying structure of the normal and anomalous data.

Appendix A

Title	Authors	Purpose	Description	Model used	Imagery type	Spatial resolution	Performance metric	Scores	Datasets used for training	Synthetic data used	Auxiliary data
Mapping illegal waste dumping sites with neural-network classification of satellite imagery	Maria Roberta Devesa & Antonio Vazquez Brust	Create comprehensive map of potential locations of illegal waste dumping sites	Identify dumping site locations and track evolutions over time using neural network classification of satellite imagery of Argentina	U-net CNN	Satellite images (RGB, SWIR-1, SWIR-2, NIR, normalized diff SWIR bands)	10m	Intersection over Union	0.675 (using RGB-NIR-SWIR-NDSW)	Georeferenced locations of known illegal waste dumps	🟥	🟥
Learning to Identify Illegal Landfills through Scene Classification in Aerial Images	Rocio Nahime Torres & Piero Fraternali	Prove feasibility of applying convolutional neural networks for scene classification in this scenario to optimize the process of waste dumps detection	Study the application of convolutional neural network (CNN) scene classification models for landfill detection in aerial images	Binary CNN classifier, ResNet50 + FPN model	Orthophotos (RGB aerophotogram metry survey)	20cm	Average precision	0.94	≈3000 images with ≈33% positive samples (AGEA)	🟥	🟥
Detecting landfills using multi-spectral satellite images and deep learning methods	Anupama Rajkumar, Tamas Sziranyi & Andras Majdik	High resolution Landfill dataset created from satellite images and applying suitable deep learning methods to detect landfills	Using satellite imagery along with modern deep learning methods to detect landfills using semantic segmentation	Fully Convolutional Network and U-Net combination	WorldView and GeoEye satellite images grayscale panchromatic image, rasterized multispectral image with high spectral resolution (upto 8 bands)	30cm	Accuracy	0.830 (using UNet-ResNet34)	High resolution multi-spectral satellite images from WorldView-3, WorldView-2 and GeoEye-1 satellite missions	🟥	🟥
Deep Learning and Remote Sensing: Detection of Dumping Waste Using UAV	Ousmane Youme, Theophile Bayet, Jean Marie Dembele & Christophe Cambier	Train a detection model for finally set up a monitoring and planning tool that can help municipality to control the problem of clandestine waste dumps	Automatic solution for the detection of clandestine waste dumps using unmanned aerial vehicle (UAV) images in the Saint Louis area of Senegal, West Africa	CNN SSD	Drone with L1D-20c RGB color camera (5472 X 3648)	A few centimeters	Intersection over Union	0.64	UAV images split up intro training and test sets	🟥	🟥
Accurate Detection of Illegal Dumping Sites Using High Resolution Aerial Photography and Deep Learning	Andreas Kamilaris, Chirag Padubidri & Savvas Karatsiolis	Develop a method for detection and reporting illegal dumping sites from high-resolution airborne images based on deep learning	Automatically detect sources of illegal waste as fast as possible using deep learning and synthetic training data	CNN classification model with residual block classification model	Multi-resolution and multi-modal optical remote-sensing dataset (high resolution RGB)	11920 × 12020	Precision and recall	0.98 and 0.90	2.000 synthetic dumping images	🟩	🟥
A method for the remote sensing identification of uncontrolled landfills: formulation and validation	S. Sivestri & M. Omri	Introduce and validate a method that uses remotely sensed information and a geographic information system (GIS) to identify unknown landfills over large areas	Remote sensing is for the first time applied to explore an area of more than 10 000 km2 with the aim of identifying possible contaminated sites	Maximum Likelihood Estimation	IKONOS satellite data	1m	N/A	N/A	Satellite data	🟥	🟩 Distributed geographical information and NIRGB
From Illegal Waste Dumps to Beneficial Resources Using Drone Technology and Advanced Data Analysis Tools: AFeasibility Study	Adi Mager & Vered Blass	Demonstrate the feasibility of mapping and analyzing the contentsof illegal waste dumpsites using drones and remote sensing techniques in order to estimatetheir circular economy.	The pilot results suggest that it is feasible to identify valuablematerials left on the ground in the form of unattended, illegally disposed waste.	N/A	Drone (GNSS RTK) remote imaging	2cm	N/A	N/A	Michnaf Company aerial mapping, Google Maps	🟥	🟥
Garbage localization based on weakly supervised learning in Deep Convolutional Neural Network	Mohd Anjum & M. Sarosh Umar	A garbage detection and localization system is proposed based on Convolutional Neural Network, which is trained on images labeled as garbage or non-garbage	This article introduces an automated method for detecting the illegal dumps of garbage using deep convolution neural network.	Deep CNN model	Scene images	Not mentioned	Custom overlapping metric	4.1 (on scale of 0-5)	Garbage Image Dataset (GIDset). Contains large number of images for garbage and non-garbage classes.	🟥	🟥

Figure 27 Table summarizing related works

Appendix B

```python
import os.path
from PIL import Image

PATH = r"D:\uncropped"
OUTPUT_PATH = r"D:\cropped\\"
dirs = os.listdir(PATH)

old_width, new_width = 6663, 8192   # Raw image size:     6663 x 8192px
new_dim = 512                       # Desired image size: 512 x 512px


def crop():
    for index, item in enumerate(dirs):
        full_path = os.path.join(PATH, item)
        if os.path.isfile(full_path):
            img = Image.open(full_path)

            # Calculate new dimensions
            left = int((old_width - new_dim) / 2)
            top = int((new_width - new_dim) / 2)
            right = int((old_width + new_dim) / 2)
            bottom = int((new_width + new_dim) / 2)

            # Crop center of the image and save to output directory
            im_crop = img.crop((left, top, right, bottom))
            im_crop.save(OUTPUT_PATH + item, quality=100)


if __name__ == '__main__':
    crop()
```

Figure 28 Python centre-cropping script

Appendix C

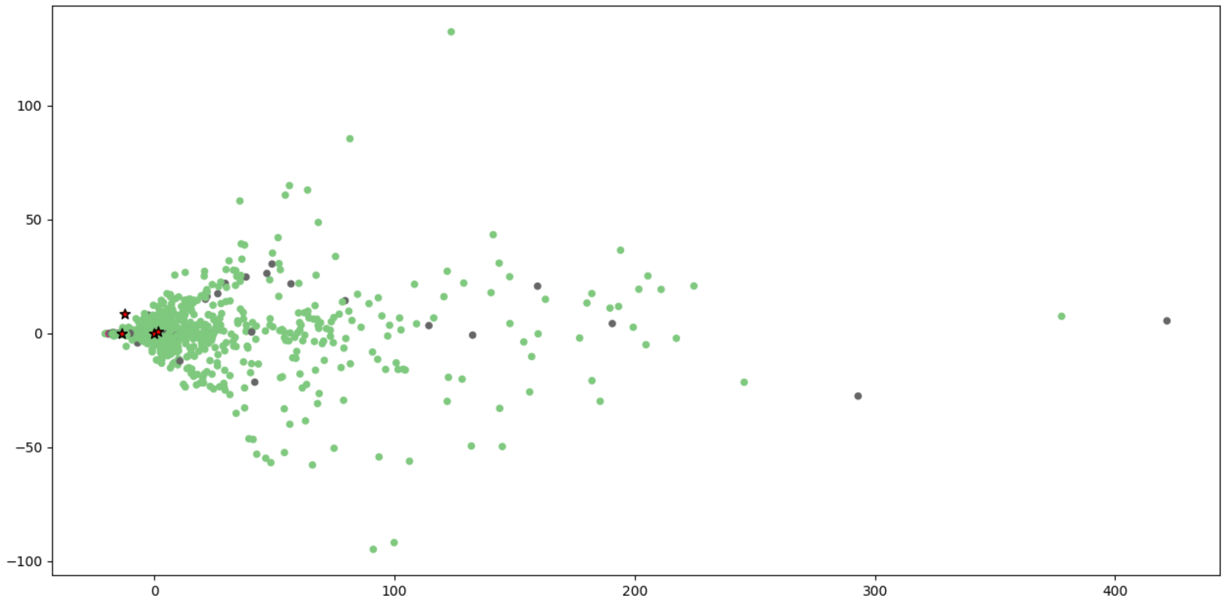

Figure 29 k-means clustering with k=4

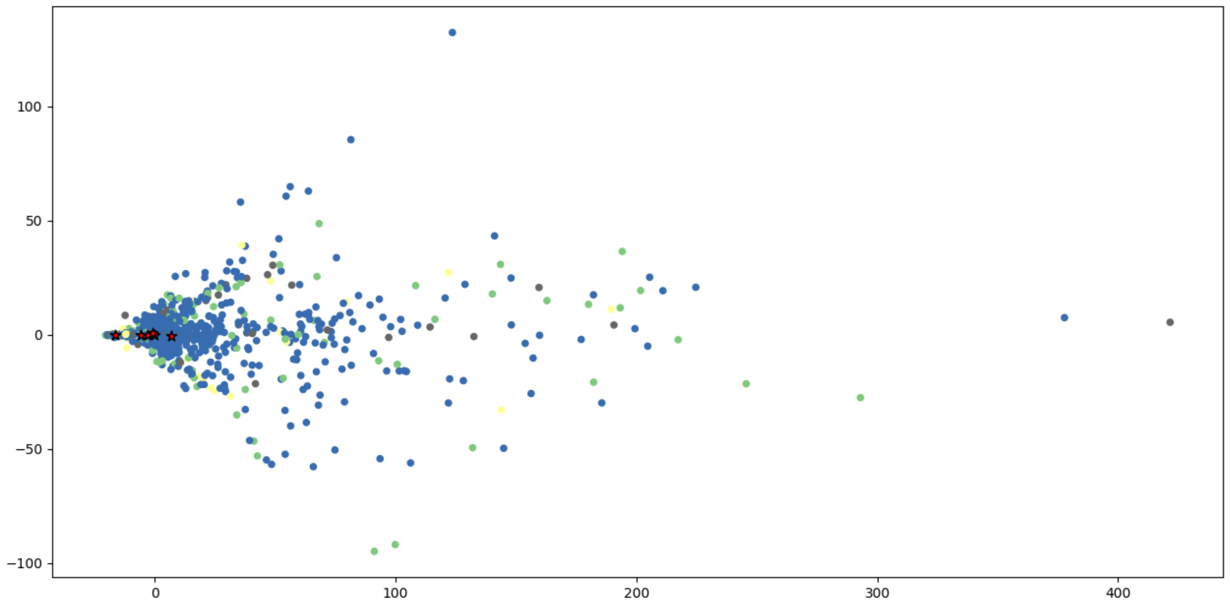

Figure 30 k-means clustering with k=6

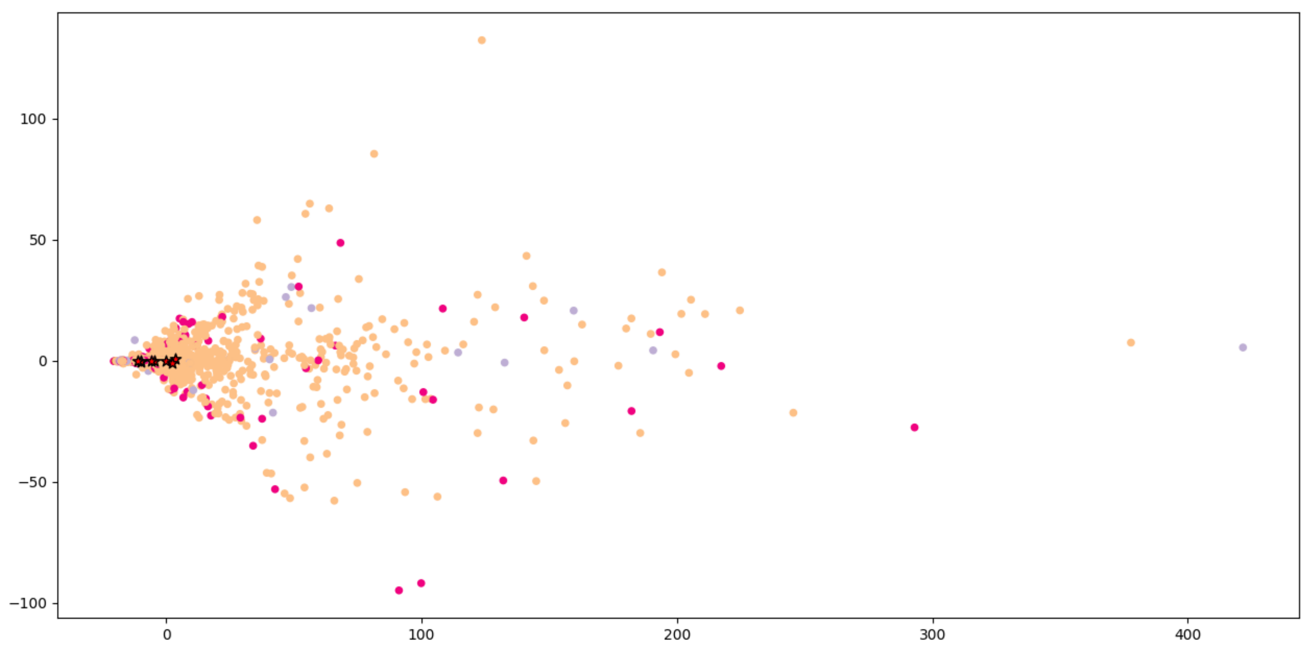

Figure 31 k-means clustering with k=8

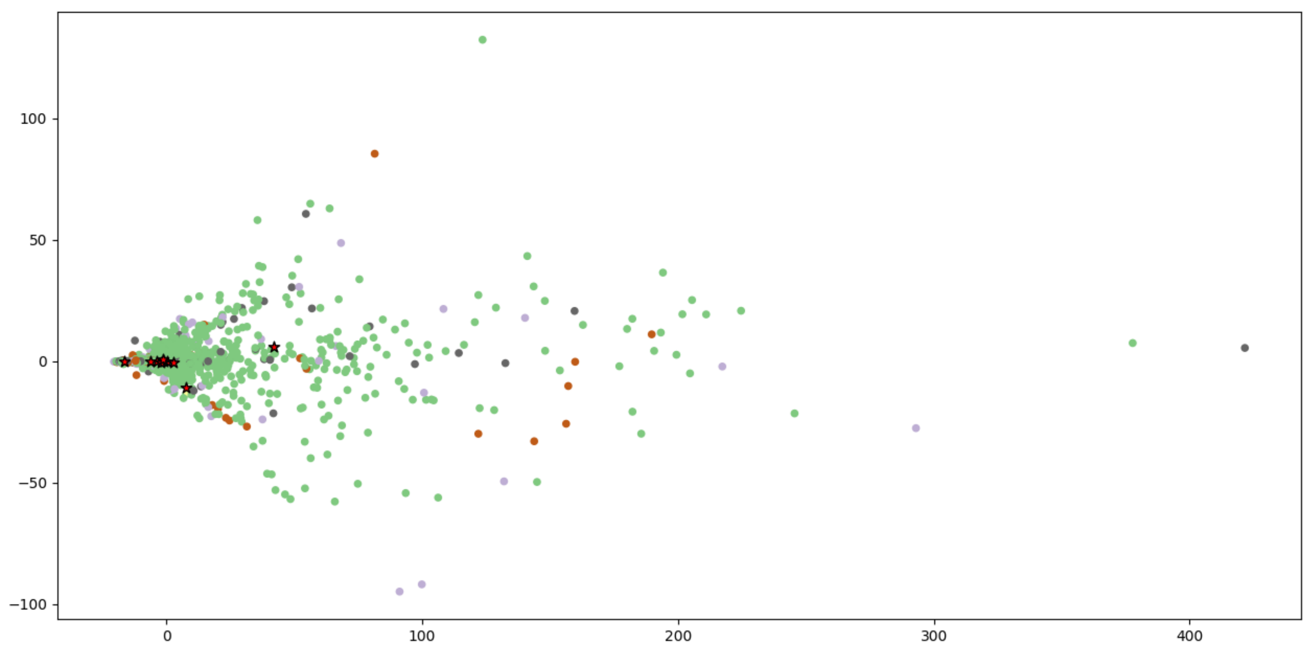

Figure 32 k-means clustering with k=10

References

[1] "World Cities Report 2022." Accessed: Oct. 05, 2022. [Online]. Available: https://unhabitat.org/

[2] M. Triassi, R. Alfano, M. Illario, A. Nardone, O. Caporale, and P. Montuori, "Environmental Pollution from Illegal Waste Disposal and Health Effects: A Review on the 'Triangle of Death,'" *International Journal of Environmental Research and Public Health 2015, Vol. 12, Pages 1216-1236*, vol. 12, no. 2, pp. 1216–1236, Jan. 2015, doi: 10.3390/IJERPH120201216.

[3] "Waste Disposal & Recycling | WM." https://www.wm.com/ca/en (accessed Oct. 05, 2022).

[4] M. Vrijheid, "Health effects of residence near hazardous waste landfill sites: A review of epidemiologic literature," *Environ Health Perspect*, vol. 108, no. SUPPL. 1, pp. 101–112, 2000, doi: 10.1289/EHP.00108S1101.

[5] P. Altavista *et al.*, "[Cause-specific mortality in an area of Campania with numerous waste disposal sites].," *Epidemiol Prev*, vol. 28, no. 6, pp. 311–321, Nov. 2004, Accessed: Oct. 05, 2022. [Online]. Available: https://europepmc.org/article/med/15792153

[6] "Goal 12 | Department of Economic and Social Affairs." https://sdgs.un.org/goals/goal12 (accessed Oct. 05, 2022).

[7] D. C. Wilson and C. A. Velis, "Waste management - Still a global challenge in the 21st century: An evidence-based call for action," *Waste Management and Research*, vol. 33, no. 12, pp. 1049–1051, Dec. 2015, doi: 10.1177/0734242X15616055/ASSET/IMAGES/10.1177_0734242X15616055-IMG1.PNG.

[8] S. Majchrowska *et al.*, "Deep learning-based waste detection in natural and urban environments," *Waste Management*, vol. 138, pp. 274–284, Feb. 2022, doi: 10.1016/J.WASMAN.2021.12.001.

[9] G. Marcus *et al.*, "Deep Learning: A Critical Appraisal," Jan. 2018, doi: 10.48550/arxiv.1801.00631.

[10] M. Cullell-Dalmau, M. Otero-Viñas, and C. Manzo, "Research Techniques Made Simple: Deep Learning for the Classification of Dermatological Images," *J Invest Dermatol*, vol. 140, no. 3, pp. 507-514.e1, Mar. 2020, doi: 10.1016/J.JID.2019.12.029.

[11] I. H. Sarker, "Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions," *SN Comput Sci*, vol. 2, no. 6, pp. 1–20, Nov. 2021, doi: 10.1007/S42979-021-00815-1/FIGURES/13.

[12] L. Deng and D. Yu, "Deep Learning: Methods and Applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3–4, pp. 197–387, 2014, doi: 10.1561/2000000039.

[13] M. Pak and S. Kim, "A review of deep learning in image recognition," *2017 4th International Conference on Computer Applications and Information Processing Technology (CAIPT)*, pp. 1–3, Aug. 2017, doi: 10.1109/CAIPT.2017.8320684.

[14] G. E. Hinton, S. Osindero, and Y. W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput*, vol. 18, no. 7, pp. 1527–1554, 2006, doi: 10.1162/NECO.2006.18.7.1527.

[15] I. H. Sarker, "Deep Cybersecurity: A Comprehensive Overview from Neural Network and Deep Learning Perspective," *SN Comput Sci*, vol. 2, no. 3, pp. 1–16, May 2021, doi: 10.1007/S42979-021-00535-6/FIGURES/11.

[16] S. Srivastava, A. V. Divekar, C. Anilkumar, I. Naik, V. Kulkarni, and V. Pattabiraman, "Comparative analysis of deep learning image detection algorithms," *J Big Data*, vol. 8, no. 1, pp. 1–27, Dec. 2021, doi: 10.1186/S40537-021-00434-W/TABLES/2.

[17] P. P. Shinde and S. Shah, "A Review of Machine Learning and Deep Learning Applications," *Proceedings - 2018 4th International Conference on Computing, Communication Control and Automation, ICCUBEA 2018*, Jul. 2018, doi: 10.1109/ICCUBEA.2018.8697857.

[18] A. R. Pathak, M. Pandey, and S. Rautaray, "Application of Deep Learning for Object Detection," *Procedia Comput Sci*, vol. 132, pp. 1706–1717, Jan. 2018, doi: 10.1016/J.PROCS.2018.05.144.

[19] D. Wang, A. Khosla, R. Gargeya, H. Irshad, A. H. Beck, and B. Israel, "Deep Learning for Identifying Metastatic Breast Cancer," Jun. 2016, doi: 10.48550/arxiv.1606.05718.

[20] S. Liu, S. Liu, W. Cai, S. Pujol, R. Kikinis, and D. Feng, "Early diagnosis of Alzheimer's disease with deep learning," *2014 IEEE 11th International Symposium on Biomedical Imaging, ISBI 2014*, pp. 1015–1018, Jul. 2014, doi: 10.1109/ISBI.2014.6868045.

[21] T. Tran, T. D. Nguyen, D. Phung, and S. Venkatesh, "Learning vector representation of medical objects via EMR-driven nonnegative restricted Boltzmann machines (eNRBM)," *J Biomed Inform*, vol. 54, pp. 96–105, Apr. 2015, doi: 10.1016/J.JBI.2015.01.012.

[22] R. Miotto, L. Li, and J. T. Dudley, "Deep learning to predict patient future diseases from the electronic health records," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9626, pp. 768–774, 2016, doi: 10.1007/978-3-319-30671-1_66/TABLES/2.

[23] A. Esteva *et al.*, "Dermatologist-level classification of skin cancer with deep neural networks," *Nature*, vol. 542, no. 7639, pp. 115–118, Feb. 2017, doi: 10.1038/NATURE21056.

[24] N. Chaudhuri, G. Gupta, V. Vamsi, and I. Bose, "On the platform but will they buy? Predicting customers' purchase behavior using deep learning," *Decis Support Syst*, vol. 149, p. 113622, Oct. 2021, doi: 10.1016/J.DSS.2021.113622.

[25] A. Da'u, N. Salim, A. Da', A. Da'u, and N. Salim, "Recommendation system based on deep learning methods: a systematic review and new directions," *Artificial Intelligence Review 2019 53:4*, vol. 53, no. 4, pp. 2709–2748, Aug. 2019, doi: 10.1007/S10462-019-09744-1.

[26] A. Singhal, P. Sinha, and R. Pant, "Use of Deep Learning in Modern Recommendation System: A Summary of Recent Works," *Int J Comput Appl*, vol. 180, no. 7, pp. 17–22, Dec. 2017, doi: 10.5120/ijca2017916055.

[27] M. Karna, D. S. Juliet, and R. C. Joy, "Deep learning based Text Emotion Recognition for Chatbot applications," *Proceedings of the 4th International Conference on Trends in Electronics and Informatics, ICOEI 2020*, pp. 988–993, Jun. 2020, doi: 10.1109/ICOEI48184.2020.9142879.

[28] M. Nuruzzaman and O. K. Hussain, "A Survey on Chatbot Implementation in Customer Service Industry through Deep Neural Networks," *Proceedings - 2018 IEEE 15th International Conference on e-Business Engineering, ICEBE 2018*, pp. 54–61, Dec. 2018, doi: 10.1109/ICEBE.2018.00019.

[29] L. Cui, S. Huang, F. Wei, C. Tan, C. Duan, and M. Zhou, "SuperAgent: A Customer Service Chatbot for E-commerce Websites," pp. 97–102, doi: 10.18653/v1/P17-4017.

[30] "Feature Extraction Definition | DeepAI." https://deepai.org/machine-learning-glossary-and-terms/feature-extraction (accessed Oct. 14, 2022).

[31] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *Journal of Big Data 2021 8:1*, vol. 8, no. 1, pp. 1–74, Mar. 2021, doi: 10.1186/S40537-021-00444-8.

[32] "What are Convolutional Neural Networks? | IBM." https://www.ibm.com/cloud/learn/convolutional-neural-networks (accessed Nov. 07, 2022).

[33] J. Song, S. Gao, Y. Zhu, and C. Ma, "Big Earth Data A survey of remote sensing image classification based on CNNs A survey of remote sensing image classification based on CNNs," 2019, doi: 10.1080/20964471.2019.1657720.

[34] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", Accessed: Oct. 27, 2022. [Online]. Available: https://github.com/

[35] A. van Etten, "Satellite Imagery Multiscale Rapid Detection with Windowed Networks", Accessed: Oct. 26, 2022. [Online]. Available: https://aws.amazon.com/public-datasets/spacenet/

[36] M. Li, Z. Zhang, L. Lei, X. Wang, and X. Guo, "Agricultural Greenhouses Detection in High-Resolution Satellite Images Based on Convolutional Neural Networks: Comparison of Faster R-CNN, YOLO v3 and SSD," *Sensors 2020, Vol. 20, Page 4938*, vol. 20, no. 17, p. 4938, Aug. 2020, doi: 10.3390/S20174938.

[37] L. Cheng, J. Li, P. Duan, and M. Wang, "A small attentional YOLO model for landslide detection from satellite remote sensing images," *Landslides 2021 18:8*, vol. 18, no. 8, pp. 2751–2765, May 2021, doi: 10.1007/S10346-021-01694-6.

[38] L. Liu, B. Zhou, G. Liu, D. Lian, and R. Zhang, "Yolo-Based Multi-Model Ensemble for Plastic Waste Detection Along Railway Lines," pp. 7658–7661, Sep. 2022, doi: 10.1109/IGARSS46834.2022.9883308.

[39] L. Zhu *et al.*, "A Review: Remote Sensing Sensors," *Multi-purposeful Application of Geospatial Data*, Dec. 2017, doi: 10.5772/INTECHOPEN.71049.

[40] B. P. Luyendyk, E. J. Hajic, and D. S. Simonett, "Side-scan sonar mapping and computer-aided interpretation in the Santa Barbara Channel, California," *Marine Geophysical Researches 1983 5:4*, vol. 5, no. 4, pp. 365–388, Dec. 1983, doi: 10.1007/BF00305389.

[41] C. Padubidri, A. Kamilaris, and S. Karatsiolis, "Accurate Detection of Illegal Dumping Sites Using High Resolution Aerial Photography and Deep Learning," *2022 IEEE International Conference on Pervasive Computing and Communications Workshops and other Affiliated Events, PerCom Workshops 2022*, pp. 451–456, 2022, doi: 10.1109/PERCOMWORKSHOPS53856.2022.9767451.

[42] Y. Long, Y. Gong, Z. Xiao, and Q. Liu, "Accurate object localization in remote sensing images based on convolutional neural networks," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 5, pp. 2486–2498, May 2017, doi: 10.1109/TGRS.2016.2645610.

[43] Y. ben Youssef, M. Merrouchi, E. Abdelmounim, and T. Gadi, "Aircraft type classification in remote sensing images using deep learning," *2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science, ICECOCS 2020*, Dec. 2020, doi: 10.1109/ICECOCS50124.2020.9314611.

[44] "Combining Synthetic Data with Real Data to Improve Detection Results in Satellite Imagery: Case Study - Oneview." https://one-view.ai/combining-synthetic-data-with-real-data-to-improve-detection-results-in-satellite-imagery-case-study/ (accessed Nov. 14, 2022).

[45] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez, "Convolutional Neural Networks for Large-Scale Remote-Sensing Image Classification," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 55, no. 2, pp. 645–657, Feb. 2017, doi: 10.1109/TGRS.2016.2612821.

[46] A. Rajkumar, T. Sziranyi, and A. Majdik, "DETECTING LANDFILLS USING MULTI-SPECTRAL SATELLITE IMAGES AND DEEP LEARNING METHODS".

[47] M. R. Devesa and A. V. Brust, "Mapping illegal waste dumping sites with neural-network classification of satellite imagery," Oct. 2021, doi: 10.48550/arxiv.2110.08599.

[48] S. Silvestri and M. Omri, "A method for the remote sensing identification of uncontrolled landfills: formulation and validation," *http://dx.doi.org/10.1080/01431160701311317*, vol. 29, no. 4, pp. 975–989, Feb. 2007, doi: 10.1080/01431160701311317.

[49] O. Youme, T. Bayet, J. M. Dembele, and C. Cambier, "Deep Learning and Remote Sensing: Detection of Dumping Waste Using UAV," *Procedia Comput Sci*, vol. 185, pp. 361–369, Jan. 2021, doi: 10.1016/J.PROCS.2021.05.037.

[50] A. Mager and V. Blass, "From Illegal Waste Dumps to Beneficial Resources Using Drone Technology and Advanced Data Analysis Tools: A Feasibility Study," *Remote Sens (Basel)*, vol. 14, no. 16, Aug. 2022, doi: 10.3390/RS14163923.

[51] A. Dabholkar, B. Muthiyan, S. Srinivasan, S. Ravi, H. Jeon, and J. Gao, "Smart illegal dumping detection," *Proceedings - 3rd IEEE International Conference on Big Data Computing Service and Applications, BigDataService 2017*, pp. 255–260, Jun. 2017, doi: 10.1109/BIGDATASERVICE.2017.51.

[52] M. Anjum and M. S. Umar, "Garbage localization based on weakly supervised learning in Deep Convolutional Neural Network," *Proceedings - IEEE 2018 International Conference on Advances in Computing, Communication Control and Networking, ICACCCN 2018*, pp. 1108–1113, Oct. 2018, doi: 10.1109/ICACCCN.2018.8748568.

[53] A. H. Mader and W. Eggink, "A Design Process for Creative Technology." The Design Society, pp. 568–573, 2014. Accessed: Nov. 15, 2022. [Online]. Available: https://research.utwente.nl/en/publications/a-design-process-for-creative-technology

[54] P. Chapman *et al.*, "CRISP-DM 1.0: Step-by-step data mining guide," *undefined*, 2000.

[55] "CRISP-DM: Stappenplan voor data gedreven werken – CBI Analytics | Power BI | Web development | Tutorials | Django | React." https://cbi-analytics.nl/een-stappenplan-richting-data-gedreven-werken-crisp-dm/ (accessed Nov. 15, 2022).

[56] "Lolle2000la/Image-Sort: Sorts your image at high speed." https://github.com/Lolle2000la/Image-Sort (accessed Feb. 08, 2023).

[57] Y. L. Kong, Q. Huang, C. Wang, J. Chen, J. Chen, and D. He, "Long Short-Term Memory Neural Networks for Online Disturbance Detection in Satellite Image Time Series," *Remote Sensing 2018, Vol. 10, Page 452*, vol. 10, no. 3, p. 452, Mar. 2018, doi: 10.3390/RS10030452.

[58] R. Chalapathy and S. Chawla, "Deep Learning for Anomaly Detection: A Survey," Jan. 2019, Accessed: Jan. 20, 2023. [Online]. Available: http://arxiv.org/abs/1901.03407

[59] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science (1979)*, vol. 313, no. 5786, pp. 504–507, Jul. 2006, doi: 10.1126/SCIENCE.1127647/SUPPL_FILE/HINTON.SOM.PDF.

[60] A. Dimokranitou, "Adversarial Autoencoders for Anomalous Event Detection in Images," 2017, doi: 10.7912/C2TS97.

[61] X. Wang, H. Guo, S. Hu, M.-C. Chang, and S. Lyu, "GAN-generated Faces Detection: A Survey and New Perspectives", Accessed: Feb. 08, 2023. [Online]. Available: https://thispersondoesnotexist.com

[62] H. Zhang *et al.*, "StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks," *IEEE Trans Pattern Anal Mach Intell*, vol. 41, no. 8, pp. 1947–1962, Oct. 2017, doi: 10.48550/arxiv.1710.10916.

[63] C. Ledig *et al.*, "Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network".

[64] L. Beggel, M. Pfeiffer, and B. Bischl, "Robust Anomaly Detection in Images using Adversarial Autoencoders".

[65] V. Leveau, A. Joly, and A. J. Adversarial, "Adversarial autoencoders for novelty detection ADVERSARIAL AUTOENCODERS FOR NOVELTY DE-TECTION", Accessed: Jan. 20, 2023. [Online]. Available: https://hal.inria.fr/hal-01636617

[66] "A Short Introduction to Generative Adversarial Networks - Thalles' blog." https://sthalles.github.io/intro-to-gans/ (accessed Feb. 07, 2023).

[67] A. Makhzani, J. Shlens, N. Jaitly, G. Brain, I. G. Openai, and B. Frey, "Adversarial Autoencoders".

[68] D. M. Hawkins, "Identification of Outliers," *Identification of Outliers*, 1980, doi: 10.1007/978-94-015-3994-4/COVER.

[69] S. Akcay, A. Atapour-Abarghouei, and T. P. Breckon, "GANomaly: Semi-Supervised Anomaly Detection via Adversarial Training," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11363 LNCS, pp. 622–637, May 2018, doi: 10.48550/arxiv.1805.06725.

[70] L. Deecke, R. Vandermeulen, L. Ruff, S. Mandt, and M. Kloft, "Image anomaly detection with generative adversarial networks," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 11051 LNAI, pp. 3–17, 2019, doi: 10.1007/978-3-030-10925-7_1/FIGURES/6.

[71] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein GAN," Jan. 2017, doi: 10.48550/arxiv.1701.07875.

[72] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved Training of Wasserstein GANs," *Adv Neural Inf Process Syst*, vol. 2017-December, pp. 5768–5778, Mar. 2017, doi: 10.48550/arxiv.1704.00028.

[73] "A One-Stop Shop for Principal Component Analysis | by Matt Brems | Towards Data Science." https://towardsdatascience.com/a-one-stop-shop-for-principal-component-analysis-5582fb7e0a9c (accessed Feb. 10, 2023).

[74] S. Mannor *et al.*, "K-Means Clustering," *Encyclopedia of Machine Learning*, pp. 563–564, 2011, doi: 10.1007/978-0-387-30164-8_425.

[75] J. Pareek and J. Jacob, "Data compression and visualization using pca and t-sne," *Lecture Notes in Networks and Systems*, vol. 135, pp. 327–337, 2021, doi: 10.1007/978-981-15-5421-6_34.

[76] F. Zhang, S. Kan, D. Zhang, Y.-G. Cen, L. Zhang, and V. Mladenovic, "Self-Attention-Based Adversarial Autoencoder Network for Surface Defect Detection," *SSRN Electronic Journal*, Mar. 2022, doi: 10.2139/SSRN.4062541.