FEDERAL UNIVERSITY OF TECHNOLOGY – PARANÁ
GRADUATE PROGRAM IN ELECTRICAL AND COMPUTER ENGINEERING

MATHEUS GUTOSKI

LEARNING AND TRANSFER OF FEATURE EXTRACTORS FOR AUTOMATIC ANOMALY DETECTION IN SURVEILLANCE VIDEOS

MASTERS DISSERTATION

CURITIBA

2018

MATHEUS GUTOSKI

LEARNING AND TRANSFER OF FEATURE EXTRACTORS FOR AUTOMATIC ANOMALY DETECTION IN SURVEILLANCE VIDEOS

Masters Dissertation presented to the Graduate Program in Electrical and Computer Engineering of the Federal University of Technology – Paraná as partial fulfillment of the requirements for the degree of “Master of Science (M.Sc.) ” – Area of concentration: Computer Engineering.

Advisor: Prof. Dr. Heitor Silvério Lopes

Co-advisor: Prof. Dr. André Eugênio Lazzaretti

CURITIBA

2018

Dados Internacionais de Catalogação na Publicação

G984L Gutoski, Matheus
2018 Learning and transfer of feature extractors for automatic anomaly detection in surveillance videos / Matheus Gutoski.-- 2018.
82 f.: il.; 30 cm.

Disponível também via World Wide Web.
Texto em inglês com resumo em português.
Dissertação (Mestrado) - Universidade Tecnológica Federal do Paraná. Programa de Pós-graduação em Engenharia Elétrica e Informática Industrial. Área de Concentração: Engenharia de Computação, Curitiba, 2018.
Bibliografia: f. 77-82.

1. Detecção de anomalias (Segurança de computadores). 2. Análise de imagem. 3. Aprendizado do computador. 4. Videovigilância. 5. Reconhecimento de padrões. 6. Segmentação de imagens. 7. Visão por computador. 8. Máquinas de vetores de suporte. 9. Métodos de simulação. 10. Engenharia elétrica – Dissertações. I. Lopes, Heitor Silvério, orient. II. Lazzaretti, André Eugênio, coorient. III. Universidade Tecnológica Federal do Paraná. Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial. IV. Título.

CDD: Ed. 23 -- 621.3

Biblioteca Central do Câmpus Curitiba – UTFPR
Bibliotecária: Luiza Aquemi Matsumoto CRB-9/794

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Diretoria de Pesquisa e Pós-Graduação

TERMO DE APROVAÇÃO DE DISSERTAÇÃO Nº 792

A Dissertação de Mestrado intitulada **“Learning and Transfer of Feature Extractors for Automatic Anomaly Detection in Surveillance Videos”** defendida em sessão pública pelo(a) candidato(a) **Matheus Gutoski**, no dia 03 de abril de 2018, foi julgada para a obtenção do título de Mestre em Ciências, área de concentração Engenharia da Computação, e aprovada em sua forma final, pelo Programa de Pós-Graduação em Engenharia Elétrica e Informática Industrial.

BANCA EXAMINADORA:

Prof(a). Dr(a). Heitor Silvério Lopes - Presidente – (UTFPR)
Prof(a). Dr(a). Lúcia Valéria Ramos de Arruda - (UTFPR)
Prof(a). Dr(a). Alceu de Souza Britto Junior - (PUCPR)
Prof(a). Dr(a). Chidambaram Chidambaram – (UDESC)

A via original deste documento encontra-se arquivada na Secretaria do Programa, contendo a assinatura da Coordenação após a entrega da versão corrigida do trabalho.

Curitiba, 03 de abril de 2018.

To my mother Leonira, my father Ewerson,
my sister Rafaela and my brother Felipe.

ACKNOWLEDGEMENTS

First of all, I would like to thank my family for the support during the many years I have dedicated exclusively to my education.

I would also like to thank all my friends from the Laboratory of Bioinformatics and Computational Intelligence for the countless times they have helped me through difficult technical and personal issues.

Special thanks to my advisor Heitor Silvério Lopes and my co-advisor André Eugênio Lazzaretti, who, during the past two years, have pushed me beyond my limits. I have learned so much from you two. Also, a special mention to Chidambaram Chidambaram for encouraging me to pursue a Masters degree and opening the doors to the laboratory.

Last but not least, I would like to thank everyone who was somehow involved in the pursuit of this degree, including friends, colleagues, and teachers.

“The first principle is that you must not fool yourself, and you are the easiest person to fool” - Richard P. Feynman

“If I have seen further, it is by standing upon the shoulders of giants” - Isaac Newton

“Science is a way of thinking much more than it is a body of knowledge” - Carl Sagan

RESUMO

GUTOSKI, Matheus. APRENDIZAGEM E TRANSFERÊNCIA DE EXTRATORES DE CARACTERÍSTICAS PARA DETECÇÃO AUTOMÁTICA DE ANOMALIAS EM VÍDEOS DE SEGURANÇA. 82 f. Masters Dissertation – Graduate Program in Electrical and Computer Engineering, Federal University of Technology – Paraná. Curitiba, 2018.

A vigilância automática de vídeos de segurança está se tornando um tema de grande importância no mundo atual. A quantidade de câmeras de vigilância em locais públicos e privados supera amplamente o número de humanos disponíveis para executar a tarefa de observação. Isto reduz a eficácia das câmeras, uma vez que as imagens de segurança geralmente são utilizadas após o ocorrido, em vez de permitir ações corretivas rápidas com base na detecção de eventos em tempo real. No entanto, a tarefa de conceber sistemas robustos de vigilância automática é bastante árdua. Este alto grau de dificuldade está associado ao problema da construção de modelos capazes de compreender a semântica humana. Os seres humanos têm a capacidade inata de observar um evento em andamento e julgar suas implicações, o que leva à tomada de decisões. Simular este entendimento em uma máquina, mesmo em um nível simplificado, tornou-se um verdadeiro desafio na pesquisa recente. Visão Computacional, Aprendizagem de Máquina e Aprendizagem Profunda são áreas de estudo relacionadas à esta questão. Juntas, estas áreas alcançaram recentemente resultados impressionantes em uma ampla gama de tarefas relacionadas à visão, e fornecem métodos e ferramentas que podem ser usados para o problema de vigilância automática de vídeos de segurança. Neste trabalho, o problema da vigilância automática é abordado a partir de uma perspectiva de detecção de anomalias. Para isto, um modelo de normalidade é aprendido a partir de vídeos previamente rotulados como normais por observadores humanos. Este modelo é então usando para detectar anomalias. Para alcançar este objetivo, a tarefa é dividida em duas subtarefas principais: extração de características e classificação. As contribuições deste trabalho estão principalmente relacionadas ao processo de extração de características, onde foram propostos dois métodos baseados em Aprendizagem Profunda. O primeiro método baseia-se na transferência de conhecimento de uma tarefa completamente independente para a detecção de anomalias em vídeos de segurança. A ideia é investigar a extensão da capacidade de generalização de um modelo, usando-o para executar uma tarefa completamente nova e inesperada. O segundo método é baseado em aprender um extrator de características que extrai representações compactas dos vídeos de segurança. Este método foi investigado sob a hipótese de que a criação de grupos compactos no espaço de características pode levar a um maior desempenho de classificação. A classificação é realizada por Máquinas de Vetores Suporte de uma classe em ambos os métodos. Os resultados mostram que o primeiro método apresentou desempenho semelhante aos métodos considerados estado da arte, o que leva à conclusão de que a capacidade de generalização de alguns modelos de Aprendizagem Profunda pode ser estendida para diferentes tarefas. Os resultados usando o segundo método corroboraram com a hipótese de compacidade, onde um ganho no desempenho da classificação foi obtido após tornar as representações compactas. Em geral, é possível concluir que ambos os métodos se mostram promissores para melhorar o processo de extração de características e podem ser importantes contribuições para sistemas robustos de detecção automática de anomalias em vídeos de segurança.

Palavras-chave: Detecção de anomalias em vídeos; Extração de Características, Aprendizado Profundo

ABSTRACT

GUTOSKI, Matheus. LEARNING AND TRANSFER OF FEATURE EXTRACTORS FOR AUTOMATIC ANOMALY DETECTION IN SURVEILLANCE VIDEOS. 82 p. Masters Dissertation – Graduate Program in Electrical and Computer Engineering, Federal University of Technology – Paraná. Curitiba, 2018.

Automatic video surveillance is becoming a topic of great importance in the current world. Surveillance cameras in private and public spaces greatly outnumber the humans available for performing the observation task. This hinders the effectiveness of the cameras since the footage is often used much after the event has occurred, rather than allowing for quick corrective action based on real-time detection of events. However, the task of devising robust automatic surveillance systems is a rather difficult one. This high degree of difficulty is associated with the problem of building models able to understand human semantics. Humans have the innate ability to observe an ongoing event and judge its implications, which then leads to decision making. Simulating this understanding in a machine, even to the slightest degree, has become a real challenge in recent research. Computer Vision, Machine Learning, and Deep Learning are fields of study deeply connected to this issue. Together, these fields have recently achieved impressive results across a wide array of vision-related tasks, and provide methods and tools that can be used for the automatic video surveillance problem. In this work, the problem of automatic surveillance is approached from an anomaly detection perspective. It consists of learning a model of normality from videos previously labeled as normal by human observers and then using this model for detecting anomalies. To achieve this goal, the task is divided into two main subtasks: feature extraction and classification. The contributions of this work are mainly related to the feature extraction process, in which two methods based on Deep Learning were proposed. The first method is based on transferring knowledge from a completely unrelated task to video anomaly detection. The idea is to investigate the extent of the generalization capacity of a model, by using it for performing a completely new and unexpected task. The second method is based on learning a feature extractor that extracts compact feature representations from surveillance video datasets. This method was investigated under the hypothesis that creating compact clusters in the feature space may improve the classification performance. Classification is performed by One-Class Support Vector Machines in both methods. Results have shown that the first method had a performance similar to state-of-the-art methods, which leads to the conclusion that the generalization capacity of some Deep Learning models can be extended to different tasks. The results using the second method have corroborated to the compactness hypothesis, in which an increase in classification performance was obtained after introducing compactness. In general, it is possible to conclude that both methods show great promise for enhancing the feature extraction process, and can be valuable contributions towards robust automatic video anomaly detection systems.

Keywords: Video Anomaly Detection; Feature Extraction; Deep Learning

LIST OF FIGURES

FIGURE 1 – Example of a One-Class Classification problem. 21
FIGURE 2 – Basic structure of a Stacked Autoencoder. 26
FIGURE 3 – A 2-dimensional example of the SVDD model. 34
FIGURE 4 – Architecture of the LeNet5 CNN. 35
FIGURE 5 – Basic idea of Transfer Learning for image and video anomaly detection. . . 38
FIGURE 6 – Structure of a basic Autoencoder. 40
FIGURE 7 – Structure of a Denoising Autoencoder. 41
FIGURE 8 – Structure of a Fully Convolutional Autoencoder. 42
FIGURE 9 – Example of the representation of normal and anomalous events on the feature space using multiple clusters of normality. 43
FIGURE 10 – Example of the representation of normal and anomalous events on the feature space using multiple clusters of normality with increased compactness. 44
FIGURE 11 – Deep Embedded Clustering optimization process. 46
FIGURE 12 – Overview of the proposed video anomaly detection method. 50
FIGURE 13 – Example of a Receiver Operating Characteristic Curve 52
FIGURE 14 – Overview of the preliminary method of compact feature learning for anomaly detection. 53
FIGURE 15 – Overview of the main method of compact feature extraction for anomaly detection. 55
FIGURE 16 – Normal and anomalous events in the LABIC3 dataset 58
FIGURE 17 – Normal and anomalous events in the LABIC4 dataset 58
FIGURE 18 – Normal and anomalous events in the Avenue dataset 59
FIGURE 19 – Normal and anomalous events in the UCSD Ped1 dataset 60
FIGURE 20 – Normal and anomalous events in the UCSD Ped2 dataset 60
FIGURE 21 – Normal and anomalous events in the UMN dataset 61
FIGURE 22 – Average performance of each CNN model on the anomaly detection task. 62
FIGURE 23 – Relation between ILSVRC validation Top 5 error and anomaly detection average performance. 63
FIGURE 24 – Relation between number of features output by a model and anomaly detection average performance. 64
FIGURE 25 – Relation between the number of layers in a model and anomaly detection average performance. 64
FIGURE 26 – Relation between the number of convolutional layers in a model and anomaly detection average performance. 65
FIGURE 27 – Samples from the STL10 dataset. 66
FIGURE 28 – Samples from the MNIST dataset. 66
FIGURE 29 – Samples from the NOTMNIST dataset. 67
FIGURE 30 – ROC curves using SDAE and DEC (best epoch) for each preliminary experiment. 68
FIGURE 31 – Performance comparison between CAE and CAE-CE on six anomaly detection benchmark datasets in terms of AUC/EER. 70

FIGURE 32 – Performance comparison betweem CAE and CAE-CE on six anomaly detection benchmark datasets in terms of TPR. 71
FIGURE 33 – Average anomaly detection performance of all methods. 73

LIST OF TABLES

TABLE 1 –Overview of the related works 31
TABLE 2 –General information about the models used in this work. 49
TABLE 3 –References of the models used in this work. 49
TABLE 4 –Architecture of the Convolutional Autoencoder. 56
TABLE 5 –Results obtained for each model and dataset in terms of AUC. 61
TABLE 6 –Results obtained for each model and dataset in terms of EER. 61
TABLE 7 – Experimental results (AUC) for the SDAE and DEC in three different experiments. 67
TABLE 8 –AUC/EER results for all datasets using the CAE-CE. 69
TABLE 9 –Confusion matrices using the EER threshold for all datasets. 70
TABLE 10 –Anomaly detection results using the EER threshold for all datasets 70
TABLE 11 – Anomaly detection performance of all methods presented in this work in terms of AUC. 71
TABLE 12 – Anomaly detection performance of all methods presented in this work in terms of EER. 72

LIST OF ACRONYMS AND ABBREVIATIONS

CV	Computer Vision
ML	Machine Learning
DL	Deep Learning
CI	Computational Intelligence
OCC	One-Class Classification
SVDD	Support Vector Data Description
OC-SVM	One-Class Support Vector Machine
OF	Optical Flow
HOG	Histogram of Oriented Gradients
CNN	Convolutional Neural Network
SAE	Stacked Autoencoder
SDAE	Stacked Denoising Autoencoder
CAE	Convolutional Autoencoder
DEC	Deep Embedded Clustering
GD	Gradient Descent
SGD	Stochastic Gradient Descent
ReLU	Rectified Linear Unit
GPUs	Graphics Processing Units
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
DAE	Denoising Autoencoder
MSE	Mean Squared Error
KL	Kullback-Leibler
TL	Transfer Learning
ROC	Receiver Operating Characteristic
AUC	Area Under the Curve
EER	Equal Error Rate
CAE-CE	Convolutional Autoencoder with Compact Embedding

CONTENTS

1 INTRODUCTION **16**
1.1 OBJECTIVES 18
1.1.1 General Objective 18
1.1.2 Specific Objectives 18
1.2 DISSERTATION STRUCTURE 19
2 RELATED WORK **20**
2.1 ONE-CLASS CLASSIFICATION 20
2.2 ANOMALY DETECTION IN IMAGES AND VIDEOS 23
2.2.1 Handcrafted Feature Extraction Methods 24
2.2.2 Learned Feature Extraction Methods 25
2.3 COMPACT FEATURE REPRESENTATION 29
3 THEORETICAL ASPECTS **32**
3.1 ONE-CLASS SUPPORT VECTOR MACHINE 32
3.2 CONVOLUTIONAL NEURAL NETWORK 34
3.3 TRANSFER LEARNING 38
3.4 AUTOENCODERS 39
3.5 COMPACT FEATURE REPRESENTATION 43
4 PROPOSED METHODS **47**
4.1 TRANSFER LEARNING FOR IMAGE AND VIDEO ANOMALY DETECTION 47
4.1.1 Data Preprocessing and Feature Extraction 49
4.1.2 Classification 50
4.1.3 Moving Average Filter 51
4.1.4 Evaluation 51
4.2 COMPACT FEATURE LEARNING USING DEEP EMBEDDED CLUSTERING 52
4.2.1 Preliminary method for compact feature learning using Stacked Denoising Autoencoders 53
4.2.1.1 Data Preprocessing 53
4.2.1.2 Training DEC 53
4.2.1.3 Classification 54
4.2.2 Main method for Compact Feature Learning using Convolutional Autoencoders 54
4.2.2.1 Data Preprocessing 55
4.2.2.2 Training the Convolutional DEC 55
4.2.2.3 Feature extraction, Classification and Evaluation 56
5 EXPERIMENTS AND RESULTS **57**
5.1 TRANSFER LEARNING APPROACH 57
5.1.1 Datasets 57
5.1.2 Experiments and Results 60
5.2 COMPACT REPRESENTATION APPROACH USING STACKED DENOISING AUTOENCODERS 64
5.2.1 Datasets 65
5.2.2 Experiments and Results 67

5.3 MAIN COMPACT REPRESENTATION APPROACH USING CAE-CE 69
5.4 COMPARISON AND DISCUSSION 71
6 CONCLUSION 74
6.1 CONTRIBUTIONS 75
6.2 FUTURE WORKS 75
REFERENCES 77

1 INTRODUCTION

Vision is a powerful tool for understanding the world around us. It is so powerful that, in fact, many species in nature have evolved the ability to see. The visual system is the complex combination of an organ that receives light, such as the eye, and the brain, which interprets the light received. In all species, the eyes and the brain evolve jointly, following the Darwinian principles. Individuals with a better perception of their surroundings have advantages in crucial aspects of their lives, such as locating prey, avoiding predators, or even finding mates.

In comparison to other species, the human eye is not special in any way. In fact, many species possess more intricate eye structures, allowing them to see farther and more clearly than us. What makes the human visual system so remarkable is the innate ability to draw high-level interpretations from the input image. We can successfully recognize objects, people, or even more subtle concepts such as emotions. However, what truly makes us different from other species is the capability of finding semantic meaning in images. That is to say, understanding the interactions between objects or people, the implications associated with them and estimating possible outcomes. Most of the times, this process of understanding is affected by past experiences, which are unique to each individual. For this reason, the interpretation of an image may vary according to the observer.

For many years, researchers have attempted to replicate the human image understanding in machines. This field of study is called Computer Vision (CV). CV often works in combination with other areas such as Machine Learning (ML) and Deep Learning (DL). Together, these areas have already taken the first steps towards computer image understanding, or in a more broad sense, towards a Computational Intelligence (CI). So far, the current state-of-the-art of CI has performed tasks that were previously done exclusively by humans, such as driving cars, controlling robots, and classifying images.

Despite the great performance of the state-of-the-art CI models, many challenges still remain unsolved. One of them is the huge amounts of annotated data required to train such models. Even in the era of Big Data, annotations are a scarce resource and require intensive human labor to be created. Hence, effectively learning from unlabeled data is a problem for future CI researchers. Another problem with current CI is the lack of ability to multi-task. Even the most advanced CI models, despite being good at a specific task, are unable to perform multiple unrelated tasks at once. This fact is the source of many CI criticisms, which state that a true CI should contain some form of general intelligence. This is also a problem for future CI researchers. These challenges indicate that CI is still in its early years, and much work still needs to be done.

The future prospects for the impact of CI on society are huge. Enhancement of human senses, such as vision, can be accomplished using CI. For instance, glasses with augmented reality can automatically identify faces in real time, which can be useful for security workers. Automation is another consequence of CI that will shape future economy. The frontier of what is possible to automate is constantly being pushed. Tasks that require the use of a sense, such as vision, are no longer exclusively accomplished by humans. Robots equipped with cameras and mechanical arms can already identify and manipulate objects automatically. The implications of large-scale automation are a concern for the future and may cause massive unemployment across the world. However, historically, the decline of certain jobs is often followed by the rise of new ones, which means that the adaptation to these new jobs will be a key element for the thrive of future generations.

Nowadays, employing CI for automating some tasks is becoming almost mandatory. This is the case in video surveillance monitoring. The advances in technology bring a massive increase in the number of security cameras scattered around public and private spaces. Another factor that contributes to this increase is the concern for public safety and security in the current world, where crime and terrorism reach alarming rates in some regions. As a result, humans can no longer keep up with analyzing the ever-increasing volume of security footage, hindering the effectiveness of the security system as a whole. Hence, this work intends to tackle the problem of automatic video monitoring, aiming specifically at automatic detection of anomalous events.

The current surveillance video monitoring method, as done by humans, is deeply ineffective on many occasions. Security footage is often used in a reactive, rather than in a proactive way. That is to say, security footage is often analyzed after the fact has occurred in order to identify the author of a prohibited act. In an ideal scenario, the system would detect the misbehavior immediately, alerting the appropriate authorities, and allowing for quick corrective action. In an even more optimistic scenario, the system could probably be able to predict this kind of event moments before it happened.

The main problem with automatic video monitoring is determining what to look for.

Humans have the innate ability to decide whether a certain event presents a danger to the individuals involved or some other implications that are worth noticing. This is, again, the problem of semantics. The term “Semantic Gap” is often used in the literature to describe the gap between an image, which is merely an ensemble of pixels, and the high-level meaning associated with it. In this context, this work addresses this issue as an anomaly detection problem.

In anomaly detection, the goal is to detect patterns that deviate from the normal or expected patterns. Therefore, it is not necessary to define what should be considered an anomaly. It is necessary, however, to clearly define what should be considered normal. Suppose a surveillance scenario in which a camera points towards a pedestrian walkway. The expected behavior is people walking normally in both directions, and occasionally practicing sports such as jogging or riding a bike. Any deviation from these patterns could characterize an anomaly, for instance, a car invading the walkway or a person fainting and staying on the ground for a long time.

In order to represent such events, it is necessary to perform a feature extraction process on the input videos. Constructing or learning proper feature extractors is a challenge in all image related tasks. The problem becomes even more challenging in cases that are different from the traditional multi-class supervised learning task, such as video anomaly detection, in which only one class is known. Methods for surpassing the limitations imposed by this type of problem are therefore necessary in order to build robust video anomaly detection systems.

1.1 OBJECTIVES

1.1.1 GENERAL OBJECTIVE

The general objective of this work is to propose feature extraction methods for automatically performing anomaly detection in surveillance videos.

1.1.2 SPECIFIC OBJECTIVES

The specific objectives of this work are:

- to propose a Transfer Learning approach for feature extraction in the context of video anomaly detection;
- to learn a model for extracting features from videos using Autoencoders;

- to further refine the Autoencoder representations by introducing feature compactness;
- to test the impact of compact representations in the anomaly detection process;
- to compare the methods presented in this work to the state-of-the-art methods, as well as a baseline method.

1.2 DISSERTATION STRUCTURE

This work is structured as follows: Chapter 2 presents the related work found in the anomaly detection literature, Chapter 3 presents the theoretical aspects regarding the methods used in this work, Chapter 4 presents the methods developed for performing video anomaly detection, Chapter 5 presents the experiments and results obtained by each method, as well as a comparison with other methods and a discussion, and finally, Chapter 6 presents the general conclusions of this work and points the directions for future works.

2 RELATED WORK

This chapter presents a literature review related to each substantial topic in the scope of this work. It is organized as follows: Section 2.1 introduces the problem of One-Class classification and the methods that have been developed to address the issue, Section 2.2 describes the problem of anomaly detection in images and videos, as well as the relevant literature review. Section 2.3 presents the problem of compact feature representation and its recent applications.

2.1 ONE-CLASS CLASSIFICATION

One-Class classification (OCC) is a particular class of problem that differs from the traditional multi-class classification problem (TAX, 2001). In multi-class classification, an object is classified as one of the existing classes. This process requires training a classifier using instances of all the known classes. However, suppose a scenario where only one class of objects is known (often referred to as positive or target class), and the goal is to determine whether a new object belongs to this class. Since the new object can be of any nature, it is unfeasible to train a classifier using objects of all other possible classes. It is feasible, however, to build a model that estimates a boundary around the positive class, and rejects objects that fall outside of it. This boundary should be defined in such a way to maximize the coverage of positive examples, and, at the same time, minimize the chance of accepting negative examples (TAX, 2001). Figure 1 shows an illustration of an OCC problem. The positive samples are encompassed by a decision boundary, which separates them from negative samples in the feature space. The positive class is represented by green circles, whilst the negative class is represented by red triangles and diamonds, indicating that different subclasses may be present in the negative class.

An example of OCC problem is detecting a malfunction in a machine. In this case, the positive class is represented by the behavior of the machine when it is operating properly. Supposing that the machine has never malfunctioned before, the only data available for collection belongs to the positive class (KHAN; MADDEN, 2009). This example illustrates

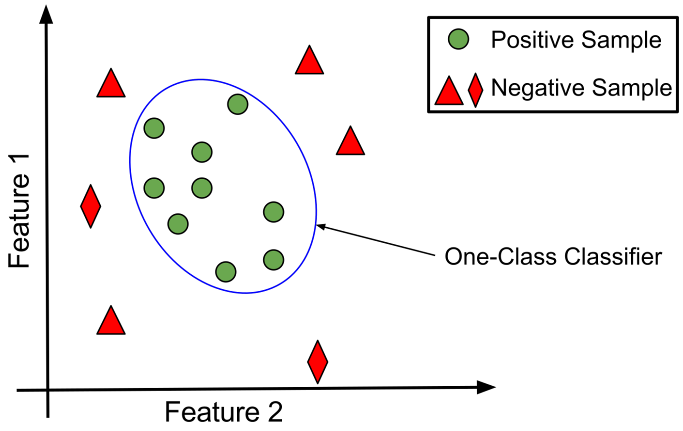


Figure 1: Example of a One-Class Classification problem.

a classic scenario in OCC problems and shows one of the possible reasons behind the lack of data for describing the negative class. Hence, in OCC, the positive class is well defined by a substantial amount of samples. In its turn, the negative class is ill-defined and very few, or in most cases, no instances at all are available during the training phase (KHAN; MADDEN, 2009).

Many classifiers were developed over the years in order to deal with the particularities of OCC problems. Tax (2001) categorizes OCC methods in three groups: density estimation methods, boundary methods and reconstruction methods. For density estimation, the author shows the applicability of the Gaussian model, Mixture of Gaussians and Parzen Density Estimator. Boundary methods include the Support Vector Data Description (SVDD), as well as the K-Centers and Nearest-Neighbor methods. For reconstruction methods, the author presents the applicability of K-Means Clustering, Self Organizing Maps, Principal Component Analysis and Autoencoder Neural Networks. After performing a series of experiments using the above-cited methods, the author has come to the conclusion that density estimation methods work better when the train and test set distributions are identical. However, density methods are greatly compromised when these distributions are different. In this case, boundary methods, such as the SVDD, perform significantly better than the other methods.

The SVDD model was developed by Tax (2001). This method aims to model the positive class by surrounding it with a hypersphere in the feature space. The method attempts to make the volume of the hypersphere as small as possible, whilst covering the maximum number of samples as possible. Hence, the model may reject a portion of the positive instances of the training set. Moreover, kernel functions such as Polynomial and Gaussian can be used to make the hypersphere more flexible (TAX, 2001).

Schölkopf et al. (2001) have developed an extension of the classic Support Vector

Machine algorithm (SVM) (CORTES; VAPNIK, 1995), which was originally developed for two-class classification problems. This method, denoted as One-Class Support Vector Machine (OC-SVM) allows tackling the OCC problem. The OC-SVM model is equivalent to the SVDD model, as discussed in Tax (2001), under the condition that both methods use the same Gaussian kernel.

Similarly to the method proposed by Schölkopf et al. (2001), the Outlier-SVM was proposed by Manevitz and Yousef (2001) in the context of document classification. However, the method performed worse than OC-SVM in most of the experiments presented by the authors. Another modification of the SVM model is presented by Yu (2005), denoted Mapping Convergence. This model allows using a set of unlabeled data along with the positive class instances. Yu (2005) discusses how easily a model can overfit or underfit in OCC problems when using SVM based methods, depending on its parameters. The author also emphasizes the importance of having a sufficiently large amount of samples in order to build a robust model that can accurately describe the data.

Currently, the SVDD is one of the most relevant methods for OCC. Erfani et al. (2016) state that the model can achieve a good generalization capacity once the parameters are properly tuned; delivers a unique solution due to the convex loss function; and can, in principle, model any training set, under the condition that an appropriate kernel function is chosen. Another advantage of this model is its capacity to represent high dimensional data without the need for large additional computational effort (TAX, 2001).

Due to its effectiveness, the model has attracted the attention of many researchers, and it is still being used to tackle OCC problems. In Erfani et al. (2016), anomaly detection is performed using OC-SVMs combined with Deep Belief Networks on several anomaly detection benchmarks. Yan, Ji and Shen (2017) employ a modified version of the OC-SVM, denoted Recursive One-Class Support Vector Machine for fault detection in chiller machines. Kumar and Ravi (2017) present a framework for document classification using Principal Component Analysis and OC-SVMs. Roodposhti, Safarrad and Shahabi (2017) successfully use two OC-SVMs to detect vegetation changes in response to drought anomalies.

Due to its popularity and performance on OCC tasks, this work employs the SVDD model proposed by Tax (2001) as its primary OCC classifier. The next Section presents a review of OCC problems in the context of anomaly detection in images and videos. A more in-depth description of the SVDD model will be presented in Section 3.1.

2.2 ANOMALY DETECTION IN IMAGES AND VIDEOS

Anomaly detection is a class of problems typically modeled under the OCC context. The term is often used as a synonym for novelty or outlier detection. According to Pimentel et al. (2014), the terms came from different domains of application, and there is no universally accepted definition for each of them. For the sake of avoiding confusion regarding these terms, this work considers them to have equal meaning, and employs the term "anomaly detection". In this class of problem, the positive class is referred to as "normal", whilst the negative class is referred to as "anomalous" or "abnormal".

Anomaly detection has practical applications in many real-life scenarios. Pimentel et al. (2014) classify these scenarios in six main domains: electronic IT security; healthcare informatics and medical diagnostics; industrial monitoring and damage detection; image processing and video surveillance; text mining; and sensor networks. This Section will focus on the domain of image processing and video surveillance since the other domains are outside of the scope of this work.

In the context of image processing and video surveillance, anomaly detection is performed using data collected through cameras, which monitor the behavior of surveillance targets (SODEMANN; ROSS; BORGHETTI, 2012). In most applications, the target for anomaly detection is often related to human behavior. However, when it comes to human behavior, it is difficult to establish what should be considered an anomaly. This is due to the fact that, in this case, anomalies are human-defined concepts, and change according to the situation (RIBEIRO; LAZZARETTI; LOPES, 2017). This issue leaves the task of defining anomalies open to interpretation and may lead to different dataset annotations depending on the individual who performed the task. Nevertheless, annotations performed by humans are provided in anomaly detection benchmarks, which, despite having received much criticism by the research community (HASAN et al., 2016; HINAMI; MEI; SATOH, 2017; IONESCU et al., 2018; COLQUE et al., 2017), are still widely used as means to validate image and video anomaly detection models.

Since video cameras generate image data, it is natural to divide the video anomaly detection task into two steps: feature extraction and classification. The feature extraction process is common in Computer Vision research and widely applied to image recognition tasks. The process is important for dealing with image-related problems, such as rotation, occlusion, and changes in illumination (DALAL; TRIGGS, 2005). This work classifies feature extraction methods in two main groups: handcrafted feature extractors and learned feature extractors.

2.2.1 HANDCRAFTED FEATURE EXTRACTION METHODS

Handcrafted feature extraction methods, as the name suggests, are constructed by human experts as an attempt to extract meaningful information from raw data. In images and videos, this information is usually related to motion, color, edges or some other fundamental property (SODEMANN; ROSS; BORGHETTI, 2012).

One of the most common approaches for representing motion in videos is by computing the Optical Flow (OF). Mehran, Oyama and Shah (2009) present a video anomaly detection algorithm based on OF and bag of words, named Social Force Model, which successfully detects abnormal crowd behavior on two anomaly detection benchmark datasets. Wang and Snoussi (2014) present the Histogram of Optical Flow Orientations (HOFO) for extracting motion features and perform classification using an OC-SVM with Gaussian kernel. Results on two benchmark video anomaly detection datasets show the effectiveness of the method. In a recent work, Colque et al. (2017) propose a spatiotemporal descriptor named Histograms of Optical Flow Orientation and Magnitude and Entropy (HOFOME), and perform classification using a simple Nearest-Neighbor search algorithm. Authors compare their approach with several other OF based approaches on two benchmark datasets and achieve similar results. Ponti, Nazare and Kittler (2017) propose a method that uses OF and Empirical Mode Decomposition (EMD), and present results on one benchmark dataset. Wang et al. (2018) propose two novel descriptors based on OF information: Spatially Localized Histogram of Optical Flow (SL-HOF) and Uniform Local Gradient Pattern-based Optical Flow (ULGP-OF). Moreover, a new classifier called One-Class Extreme Learning Machine (OCELM) is proposed. Authors compare their approach to several others and achieve results comparable to the state-of-the-art approaches.

Appearance features, such as colors and edges are also often used in combination with motion features since some anomalies are not related to motion, but to appearance. For instance, anomalous objects entering the scene or being abandoned in a crowded place could characterize such a situation. Amraee et al. (2017) propose a method for video anomaly detection that employs the well-known Histogram of Oriented Gradients (HOG) (DALAL; TRIGGS, 2005) combined with OF features. Video anomaly detection has also been performed using the Local Binary Patterns (LBP) (OJALA; PIETIKAINEN; HARWOOD, 1994) texture descriptor, as presented by Xu et al. (2011a, 2011b). Cheng, Chen and Fang (2015) use a combination of various descriptors, namely Interest Point Response (IPR), 3D Scale Invariant Feature Transform (3DSIFT), as well as 3D variations of HOG and Histogram of Optical Flow (HOF). Authors use a Gaussian process regression to estimate the likelihood of an event being abnormal. The results show that this method outperforms several other approaches in the literature.

Despite being successful at describing many aspects of images, handcrafted feature extractors have limitations. For instance, Xu et al. (2015) point out that such methods require *a priori* knowledge about the scene in which anomaly detection should be performed. This limitation becomes more clear when dealing with complex scenes, in which the *a priori* knowledge is hard to obtain. This issue has also been addressed in other image processing applications, such as text feature extraction (LIANG et al., 2017) and image quality assessment (LI et al., 2016). As an alternative, some works have proposed a feature learning process, in which a feature extractor is learned from the data. This approach softens the need for *a priori* knowledge and creates fine-tuned feature extractors for specific problems.

2.2.2 LEARNED FEATURE EXTRACTION METHODS

The task of learning features from raw data is challenging. Recently, DL models such as the Convolutional Neural Network (CNN) have achieved human-level performance on image recognition tasks (SZEGEDY et al., 2015). The success of CNNs arises from the fact that features are learned directly from the raw image data, unlike in traditional Computer Vision where features are extracted using handcrafted methods. However, this success comes at a cost. The feature learning process in CNNs is guided by the annotations, which make the network learn to differentiate between a predefined number of classes. Notwithstanding, obtaining annotated datasets require intense human labor. This type of learning is defined as supervised learning.

Sodemann, Ross and Borghetti (2012) categorize anomaly detection learning tasks in three main groups: supervised, unsupervised, and *a priori* knowledge application. Anomaly detection in the context of OCC (when only the normal class is known) falls under the supervised learning category. However, there is a fundamental difference between multi-class classification and OCC: the number of classes available during the training phase. Most supervised learning algorithms are not directly applicable to OCC problems since they were devised for dealing with multiple classes. For instance, CNNs require at least two classes in order to learn representative features from data. Hence, supervised OCC and supervised multi-class classification are fundamentally different problems, and should not be treated or understood as similar.

Since CNNs were devised for multi-class problems, different DL models have been used in the context of anomaly detection. Most of the approaches are based on Autoencoders

(AEs) and its variants. AEs are unsupervised learning algorithms since class annotations are not used in the training process. The AE is a Neural Network that attempts to reconstruct the input data at the output layer after going through a hidden layer (RIBEIRO; LAZZARETTI; LOPES, 2017). The hidden layer is commonly referred to as "bottleneck", and holds the latent feature representation of the input data.

Deep AEs can be constructed by stacking many fully connected layers, creating a Stacked Autoencoder (SAE). The first part of the network, which comes before the bottleneck is commonly referred to as the "encoder", whilst the last part is named the "decoder". Figure 2 shows the basic structure of a SAE.

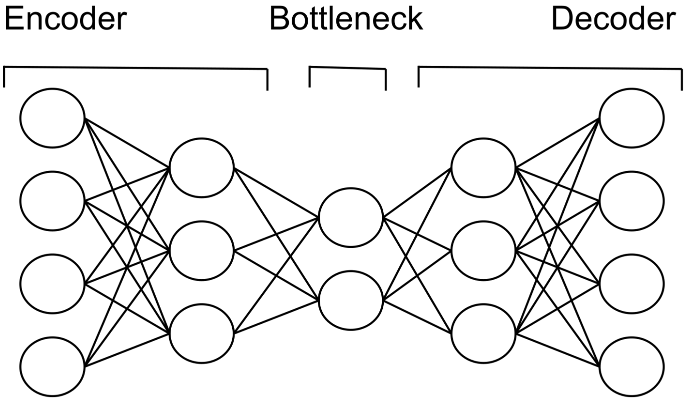


Figure 2: Basic structure of a Stacked Autoencoder.

AEs are trained by using the backpropagation algorithm (RUMELHART; HINTON; WILLIAMS, 1986), with the objective of minimizing the difference between the input and output. This difference is often referred to as the "reconstruction error". Since only normal samples are used during the training phase, it is expected that the AE learns a model of normality. A more in-depth explanation about AEs is presented in Section 3.4. AEs have been used for video anomaly detection in two main ways:

1. Solely as a feature extractor, by extracting features from the bottleneck layer and using a One-Class classifier, such as the OC-SVM;
2. As a feature extractor and classifier, by measuring the reconstruction error of new samples, and defining a threshold above which a sample is considered anomalous. This method works under the assumption that anomalous samples have higher reconstruction errors since they have not been seen during the training phase.

Regarding the first method, Xu et al. (2015) proposed the Appearance and Motion DeepNet (AMDN), which consists of three pipelines of Stacked Denoising Autoencoders

(SDAE), where each one learns a specific type of representation: appearance features, motion features, and a joint representation. These representations are then fed to OC-SVM classifiers, which perform the fusion of the three pipelines for a final classification. The results obtained by the method are still considered state-of-the-art.

Sabokrou et al. (2015) employed Sparse Autoencoders (SPAE) for real-time anomaly detection in videos. The SPAEs learn different aspects of the video from non-overlapping cubic patches, generating local and global features. The classification task is performed by Gaussian classifiers. Authors successfully detect anomalies on two benchmark datasets. A similar approach was presented by Narasimhan and Kamath (2017) using Sparse Denoising Autoencoders (SPDAEs). Authors also put emphasis on the low computational time of the method.

In a more recent work, Sabokrou et al. (2017) presented a method named Deep Cascade. The method consists of cascading Gaussian classifiers in combination with a Deep AE and a CNN in a multi-stage anomaly detection process. Early stages detect normal patches that are considered “simple”, such as background patches. Later stages detect more difficult patches, which contain more complex information. Results on three benchmark datasets demonstrate that the method achieves state-of-the-art performance, and also outperforms other methods in terms of computational time.

Tran and Hogg (2017) presented the Convolutional Winner-Take-All Autoencoder (Conv-WTA), which is a non-symmetric Convolutional Autoencoder (CAE) with three layers on the encoder and one layer on the decoder. Authors use only Optical Flow data as input for the network and assume that anomalies only occur where there is motion. Moreover, a patch extraction strategy is used, and classification is performed by OC-SVMs. Results are comparable to other state-of-the-art approaches.

Instead of dividing the anomaly detection task into feature extraction and classification, some works use the reconstruction error of the AE in an end-to-end classification strategy. Hasan et al. (2016) present a deep CAE for learning temporal regularities in videos by concatenating several frames to form a cubic input. A data augmentation strategy was also used to compensate the large number of parameters of the deep CAE. Authors also present a secondary method, based on the combination of handcrafted appearance and motion features (HOG and HOF) as input to an AE. In both methods, an anomaly score is computed based on the reconstruction error of new samples. Results suggest that the CAE approach outperforms the handcrafted plus AE approach, and performs similarly to the state-of-the-art methods on some benchmark datasets.

Ribeiro, Lazzaretti and Lopes (2017) presented a study on deep CAEs for video anomaly detection by aggregating high-level information to the input data, such as borders and OF information. The classification was performed using the reconstruction error, achieving significant results on three benchmark datasets.

Luo, Liu and Gao (2017) proposed the Temporally-Coherent Sparse Coding (TSC), which forces neighboring frames to have similar reconstruction errors. The method is implemented using a Stacked Recurrent Neural Network (SRNN), which introduces a temporal element into the model. Classification is done by forwarding all patches extracted from an image throughout the SRNN and capturing the highest reconstruction error among these patches. Authors also introduce a novel video anomaly detection dataset, which contains more data than the combination of many commonly used anomaly detection benchmarks. The method achieves results comparable to other state-of-the-art approaches on benchmark datasets and shows that the dataset introduced is very challenging.

Chong and Tay (2017) presented a method for learning spatiotemporal features from videos using a Convolutional Long Short-Term Memory (LSTM) Autoencoder. The model contains convolutional layers on the encoder and decoder parts of the network and convolutional LSTM layers on the bottleneck. Classification is performed by using the reconstruction error as an anomaly score. The method presented results comparable to other state-of-the-art models.

Recent literature regarding the use of AEs and its variants to tackle the video anomaly detection is wide. Kiran, Thomas and Parakkal (2018) present a survey where over seventy recent AE approaches have been categorized based on the type of model and classification strategy. This shows that AE models currently hold many state-of-the-art results on various video anomaly detection benchmark datasets. However, AEs require an intensive computational endeavor during their training phase and require a high amount of data in order to be effective.

Recent works investigate the applicability of a different feature learning paradigm that consists of transferring knowledge between different tasks. This idea has been named Transfer Learning (PAN; YANG, 2010). Up to date, two works have employed this paradigm for video anomaly detection. Smeureanu et al. (2017a) employ a pre-trained CNN on the ImageNet dataset as a feature extractor. Authors forward the anomaly detection frames throughout the CNN and capture their representation at a given convolutional layer. These features are then fed to an OC-SVM for classification. The CNN features have shown to be robust even in a different image classification task, achieving near the state-of-the-art results with no training phase for the feature extractor. Similarly, Hinami, Mei and Satoh (2017) obtained generic knowledge from unrelated annotated image databases in order to detect regions of interest on the anomaly detection video frames. These regions of interest are then categorized and classified as normal or anomalous, achieving state-of-the-art results.

Transfer learning applied to video anomaly detection is still underexplored in the literature, despite achieving very promising results with low computational effort in other tasks, such as multi-class supervised classification. Hence, part of this work aims to further explore the applications of Transfer Learning to video anomaly detection.

2.3 COMPACT FEATURE REPRESENTATION

The quality of the features is critical to the success of most classifiers. Features should be informative, in a way to accurately represent a class, while at the same time being distinctive, to allow easy discrimination between classes. Notwithstanding, devising methods to extract such features is a difficult task.

In past works, several approaches have been proposed in order to reduce the influence of features on the classifiers. A key area of study in this context has been named as Metric Learning. It consists of learning a metric that more accurately measures the similarity (distance) between objects (SHENTAL et al., 2002). For instance, Weinberger and Saul (2009) learn a Mahalanobis distance for a classification problem using the K-Nearest-Neighbor (K-NN) algorithm. Authors emphasize the importance of the distance metric in this particular problem and propose a procedure for learning a distance metric subject to the constraints of minimizing the distance between K-Nearest-Neighbors and maximizing the distance between classes. Results show significant improvement in performance when compared to the Euclidean distance metric. However, this method is not directly applicable to OCC problems, since only the normal class is available during the training phase.

One way to apply Metric Learning to the anomaly detection problem is by approaching it from a clustering perspective. Normal events can be represented by a set of clusters, each one representing a different aspect of normality (XU et al., 2014; BODESHEIM et al., 2013). The within-cluster distance of each cluster of normality can then be reduced by using methods such as the Relevant Component Analysis (RCA) (SHENTAL et al., 2002; BAR-HILLEL; WEINSHALL, 2005), which contains a constraint that prevents all points from shrinking to a single point. Differently, Bodesheim et al. (2013) introduced the Kernel Null-Space method for anomaly detection in both the OCC and multi-class classification contexts. The method ignores the constraint present in RCA and projects all data points that belong to the same class (or cluster) to a single point in the space. This removes a drawback present in RCA, which is the preservation of most pairwise distance characteristics in the mapped space (BODESHEIM et al., 2013). The Null-Space method outperforms other multi-class anomaly detection problems in two benchmark datasets. A more recent approach using the Null-Space method was presented by Zhang, Xiang and Gong (2016). Authors perform person re-identification and achieve state-of-the-art results on some benchmarks.

However, the Kernel Null-Space method also has limitations. Bodesheim et al. (2015) briefly argue that the even with a classifier based on Null-Space representation, results may still be suboptimal due to the feature extraction process not being the most appropriate.

Some works have addressed this problem by attempting to learn compact feature representations from the data, instead of learning a distance metric. Song et al. (2013) presented a clustering method based on AEs. Authors modify the objective function by adding a term that imposes compactness on the feature learning process. As a result, within-cluster distances became small, whilst between-cluster distances increased. This led to improved performance on the clustering task.

A recent work presented by Xie, Girshick and Farhadi (2016), devised a method for automatically learning compact feature representations for the unsupervised clustering problem, named Deep Embedded Clustering (DEC). It is based on a two-phase training process using a SDAE. The first phase learns to reconstruct the input, and the second phase is a fine-tuning process aiming at achieving compact clusters in the bottleneck representation. The method has outperformed state-of-the-art clustering algorithms on multi-class image classification benchmarks.

Inspired by the success of the applications presented by Song et al. (2013), Xie, Girshick and Farhadi (2016), part of this work pursues to explore compact feature learning in the context of anomaly detection in images and videos. Both methods have shown increased performance on the unsupervised clustering task, however, learning compact representations using modified versions of AEs in the context of image and video anomaly detection have not yet been explored in the literature. An overview of the related works is presented in Table 1.

Table 1: Overview of the related works

Authors	Method	Feature Extraction Method			Datasets								
Citation	**Name**	**Handcrafted**	**Learned**	**Compact**	**Avenue**	**UCSD Ped1**	**UCSD Ped2**	**UMN**	**Subway**	**LABIC3**	**LABIC4**	**Other**	**Total**
Mehran, Oyama and Shah (2009)	Social Force Model	x						x				x	2
Wang and Snoussi (2014)	HOFO	x						x				x	2
Colque et al. (2017)	HOFOME	x				x	x		x				3
Ponti, Nazare and Kittler (2017)	EMD	x					x						1
Wang et al. (2018)	SL-HOF, ULGP-OF	x				x	x	x					3
Amraee et al. (2017)	HOG, OF	x					x						1
Xu et al. (2011a, 2011b)	LBP	x				x	x					x	3
Cheng, Chen and Fang (2015)	IPR, 3DSIFT, 3DHOG, 3DOF	x				x			x			x	3
Xu et al. (2015)	AMDN		x			x	x					x	3
Sabokrou et al. (2015)	SPAE		x			x	x	x					3
Narasimhan and Kamath (2017)	SPDAE		x			x	x	x					3
Sabokrou et al. (2017)	Deep Cascade		x			x	x	x					3
Tran and Hogg (2017)	Conv-WTA		x		x	x	x						3
Hasan et al. (2016)	CAE + Handcrafted Features	x	x		x	x	x		x				4
Ribeiro, Lazzaretti and Lopes (2017)	CAE + High Level Information	x	x		x	x	x						3
Luo, Liu and Gao (2017)	TSC		x		x	x	x		x			x	5
Chong and Tay (2017)	Convolutional LSTM AE		x		x	x	x		x				4
Smeureanu et al. (2017a)	Transfer Learning		x		x			x					2
Hinami, Mei and Satoh (2017)	Transfer Learning		x		x		x						2
Xie, Girshick and Farhadi (2016)	DEC		x	x								x	3
This work	Transfer Learning		x		x	x	x	x		x	x		6
This work	DEC		x	x	x	x	x	x		x	x	x	9

3 THEORETICAL ASPECTS

This chapter presents the theoretical foundation for the methods used in this work. It is organized as follows: Section 3.1 presents the OC-SVM and its formulation, Section 3.2 presents the principles of CNNs, Section 3.3 presents the idea of Transfer Learning with CNNs, Section 3.4 presents an explanation about AEs, and finally, Section 3.5 presents the theory for extracting compact feature representations using AEs.

3.1 ONE-CLASS SUPPORT VECTOR MACHINE

The OC-SVM model described in this Section is identical to the SVDD model (TAX, 2001). For a given input with N multi-dimensional samples $(\mathbf{x}_1, ..., \mathbf{x}_N)$, it is assumed that there is a hypersphere that encompasses all of the samples. The hypersphere contains a center $\mathbf{a}$ and a radius R. When all samples are contained within the hypersphere, the empirical error[1] is zero. However, in order to make the model more robust, a structural risk[2] term is introduced:

$$\varepsilon_{struct}(R, \mathbf{a}) = R^2, \tag{1}$$

which is minimized subject to the constraints:

$$\|\mathbf{x}_i - \mathbf{a}\|_2^2 \leq R^2, \quad \forall i. \tag{2}$$

This term includes the possibility that not all samples will be contained within the hypersphere, i.e. the empirical error does not have to be necessarily zero. With both structural and empirical errors, the model attempts to encompass the maximum number of samples using the smallest possible hypersphere.

By introducing the slack variables, which are subject to $\boldsymbol{\xi}, \xi_i \geq 0, \forall i$, the minimization

[1] Empirical Error: amount of samples incorrectly classified in the training set.

[2] Structural Risk: the risk of a classifier performing poorly when presented with previously unseen samples. In general, simple classifiers are preferred over complex ones (VAPNIK, 1998).

problem can be written as:

$$\varepsilon(R,\mathbf{a},\boldsymbol{\xi}) = R^2 + C\sum_i \xi_i, \tag{3}$$

grouping most patterns within the hypersphere with the constraints:

$$\|\mathbf{x}_i - \mathbf{a}\|_2^2 \leq R^2 + \xi_i, \quad \xi_i \geq 0, \quad \forall i, \tag{4}$$

where C is a trade-off parameter between structural and empirical errors, whilst $\boldsymbol{a}$, R and $\boldsymbol{\xi}$ have to be optimized under the constraints presented in Equation 4.

Similarly to the SVM model, kernel functions can be used to make the model more flexible (VAPNIK, 1998). This flexibility is introduced by allowing the data to be mapped to a different feature space, in which the hypersphere fits the data points more comfortably (TAX, 2001). In this work, the RBF kernel function is used, as defined in Equation 5:

$$K(\mathbf{x}_i,\mathbf{x}_j) = \exp\left(\frac{-\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2}{\upsilon^2}\right), \tag{5}$$

where υ represents the width kernel parameter.

Finally, the Lagrangian dual problem can be constructed and the final error can be written as:

$$L = \sum_i \alpha_i K(\mathbf{x}_i,\mathbf{x}_i) - \sum_{i,j} \alpha_i \alpha_j K(\mathbf{x}_i,\mathbf{x}_j), \tag{6}$$

with respect to α_i, subject to the following constraint:

$$0 \leq \alpha_i \leq C, \ \forall i. \tag{7}$$

Figure 3 displays an example in two dimensions, where one of the objects is outside of the sphere. The center of the sphere is represented by $\boldsymbol{a}$, R represents the radius, and ξ_i is the slack variable that represents the distance between a point and the edge of the sphere. In this illustration, it was advantageous to leave said point out of the sphere in exchange for reducing its radius.

In the OC-SVM, a new pattern $\mathbf{z}$ is classified as an anomaly if:

$$AS = \sum_i \alpha_i \exp\left(\frac{-\left\|\mathbf{z} - \mathbf{x}_i\right\|^2}{\upsilon^2}\right) - \tag{8}$$
$$\frac{1}{2}\left[1 + \sum_{i,j} \alpha_i \alpha_j \exp\left(\frac{-\left\|\mathbf{x}_i - \mathbf{x}_j\right\|^2}{\upsilon^2}\right) - R^2\right] < 0,$$

where AS is the anomaly score of the pattern $\mathbf{z}$. This equation is obtained from Equation 6 using

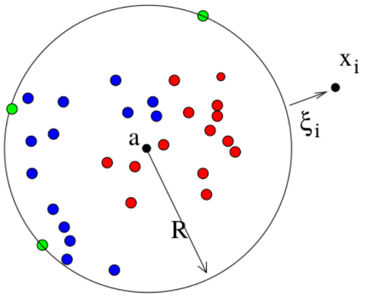


Figure 3: A 2-dimensional example of the SVDD model.

Source: Tax (2001)

the RBF kernel.

3.2 CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Networks (CNNs) are a special type of feed-forward deep Neural Network designed to deal with 2D data, such as, but not restricted to, images. It was made popular by LeCun et al. (1998) in a handwritten digits image classification task, and quickly gained attention from the research community.

CNNs have overcome a great limitation of traditional image classification methods: the hand designed feature extraction process. Handcrafted methods require prior analysis of the target data in order to discover the kind of features that could be relevant to the classification task. This sometimes led to a trial-and-error process of designing the feature extractor and testing its performance. Nevertheless, this method has been successfully applied in many Computer Vision applications (DALAL; TRIGGS, 2005).

CNNs have proposed a new paradigm for supervised image classification. The task no longer required detached feature extraction and classification processes. The network could automatically learn the most suitable feature extractor directly from the raw image data, and simultaneously perform classification, making it an end-to-end classifier. This led to a significant performance boost in image recognition tasks when compared to traditional methods (KRIZHEVSKY; SUTSKEVER; HINTON, 2012), and eventually surpassed human-level performance (SZEGEDY et al., 2015).

The CNN can be divided into two main blocks: the feature extractor and the classifier. The feature extractor is composed of convolutional layers followed by non-linearities and sub-sampling layers, often referred to as pooling layers. Stacking multiple instances of these layers produces a hierarchical feature learning process, where deeper layers learn more complex representations. In its turn, the classifier part is composed of one or more fully-connected layers, each followed by non-linearities. Figure 4 presents a classic CNN architecture used for handwritten digits classification (LECUN et al., 1998). The first convolutional layer generates six feature maps from the original image, followed by a pooling layer that cuts the dimensions of the feature maps by half. This process is repeated once more by another couple of convolutional and pooling layers. Then, the representations from the last pooling layer is fed to fully connected layers and finally to the output layer, which performs classification.

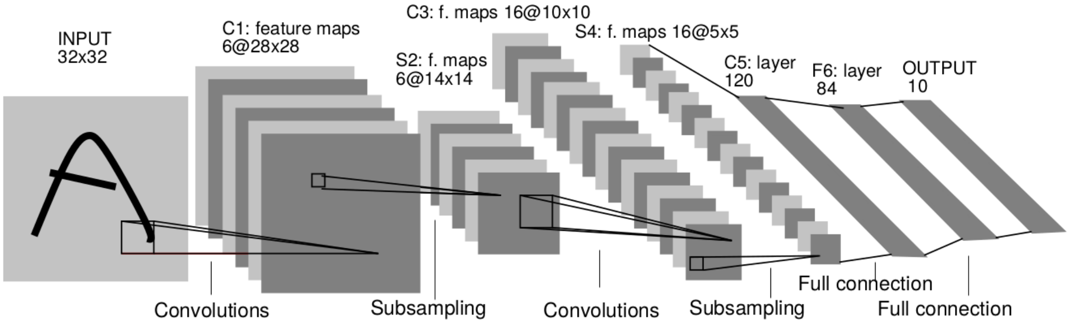


Figure 4: Architecture of the LeNet5 CNN.

Source: LeCun et al. (1998)

A convolutional layer performs a mathematical operation called convolution, which is commonly represented by the $*$ symbol. When dealing with 2-D data, such as images, the convolutional operation outputs a feature map M by performing the operation $M = I * K$, where I is the input image and K is a 2-D kernel. Common parameters of a convolutional layer are the number of output feature maps and the kernel size.

A pooling layer serves two purposes: dimensionality reduction and addition of translation invariance. A $m \times n$ sliding window over the image computes some sort of operation over the current window. The most common are Average Pooling and Max Pooling. Average Pooling outputs the average of values in the current window, whilst Max Pooling outputs the greatest value in the current window.

The output of a neuron y_k is given by the weighted sum of its inputs, which is then fed to an activation function ϕ. Equation 9 describes this process, where x_j is the value of the input j, and w_{kj} is the value of the weight kj:

$$y_k = \phi(\sum_j w_{kj} x_j). \tag{9}$$

The output layer contains the same number of neurons as classes in the classification problem, and is usually combined with an activation function ϕ, such as Softmax (Equation 10), which takes an input $\mathbf{x}$ with k elements and converts it to a vector $\hat{\mathbf{y}}$ that contains the probabilities of the input belonging to each class.

$$\hat{\mathbf{y}}(\mathbf{x}_i) = \frac{e^{\mathbf{x}_i}}{\sum_k e^{\mathbf{x}_k}}. \tag{10}$$

The probabilities $\hat{\mathbf{y}}$ are then used to compute the loss function. One of the most widely used loss functions in CNNs is the Cross Entropy Loss, which computes the divergence between the Softmax predictions $\hat{\mathbf{y}}$ and the true class labels $\mathbf{y}$ (Equation 11).

$$CEL(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_i y_i \log \hat{y}_i. \tag{11}$$

Given the loss of the network, the weight vector of the final layer $\mathbf{w}$ is updated by using the Gradient Descent (GD) method. This is done by computing the partial derivative of the loss function in respect to the weights, as shown in Equation 12, where $\mathbf{w}^t$ represents the weights in the current training epoch, $\mathbf{w}^{t-1}$ represents the weights in the previous epoch, CEL is the loss function, and η is the learning rate parameter, which determines the step size.

$$\mathbf{w}^t = \mathbf{w}^{t-1} - \eta \frac{\partial CEL}{\partial w}. \tag{12}$$

This gradient is used to update the weights on the final layer of the network. In order to propagate the changes to the other layers, the back-propagation algorithm is used along with the chain rule (RUMELHART; HINTON; WILLIAMS, 1986).

The GD method, despite being effective, can be slow when using large datasets. A modification of the method, named Stochastic Gradient Descent (SGD), aims at speeding up the process by using random sampling to update the weights, making the process faster at the cost of unpredictable convergence (PERLIN, 2015).

Over the years, many techniques have been developed to further enhance the performance of CNNs. Jarrett et al. (2009) proposed the Rectified Linear Unit (ReLU) activation function, defined by $ReLU(x) = \max(0, x)$. ReLU has replaced the hyperbolic tangent, which until then was the most used function in Neural Networks. Its faster convergence was demonstrated by Krizhevsky, Sutskever and Hinton (2012).

A common problem in deep CNNs is overfitting. The term is used to describe the lack of generalization capacity of the network. The problem happens when the network gets overly adapted to the training set. An overfit CNN shows good performance on the training phase, however, new images are likely to be misclassified. Recognizing the issue, several researchers have proposed techniques to minimize the effects of overfitting in CNNs. One of the most common technique to tackle the issue is dropout (HINTON et al., 2012). It consists of randomly disabling neurons during the training phase based on an user defined probability. This prevents the co-adaptation of neurons, and consequently, increases generalization.

Another common approach to reduce overfitting is by using L1 or L2 regularization (NG, 2004). This technique modifies the loss function by adding a penalty term that forces small weights in the network. This increases generalization by ensuring all neurons are used in the process, instead of only local groups of neurons. The L1 regularization term adds the sum of the absolute values of the weight vector **w** to the loss function as follows:

$$\mathbf{w}^t = \mathbf{w}^{t-1} - \eta \frac{\partial CEL}{\partial w} + \sum_j |\mathbf{w}_j|. \tag{13}$$

The L2 regularization, in its turn, modifies the loss function as follows:

$$\mathbf{w}^t = \mathbf{w}^{t-1} - \eta \frac{\partial CEL}{\partial w} + \frac{1}{2}||\mathbf{w}||_2^2. \tag{14}$$

Krizhevsky, Sutskever and Hinton (2012) proposed another method to reduce overfitting: data augmentation, which consists of generating new training examples from the already existing ones. This is done by applying random transformations to the input data, for instance, rotation, translation, noise, or even a combination of these transformations. This reveals another weakness of CNNs: the need for large annotated datasets to achieve satisfying performance.

However, training a CNN on large datasets requires expensive computational endeavor. It was not until the widespread use of Graphics Processing Units (GPUs) that CNNs became viable as a solution to real-world problems. GPUs greatly accelerate the training process of Artificial Neural Networks by performing computations in a highly parallel manner. Another factor that contributed to the popularity of CNNs was the development of DL frameworks, such as Caffe (JIA et al., 2014) and TensorFlow (ABADI et al., 2016). These tools removed the complexity barrier in training and deploying CNNs and other DL methods. Moreover, integration with GPUs became transparent to the developer.

3.3 TRANSFER LEARNING

In the context of this work, Transfer Learning (TL) is the process of using pre-trained CNN models solely as feature extractors, as opposed to an end-to-end classifier. The features obtained in this way are then used in the anomaly detection context. Figure 5 illustrates the basic idea behind TL. A CNN model is trained on an unrelated classification task, and the weights are then used for a new purpose, in this case as a feature extractor for anomaly detection frames.

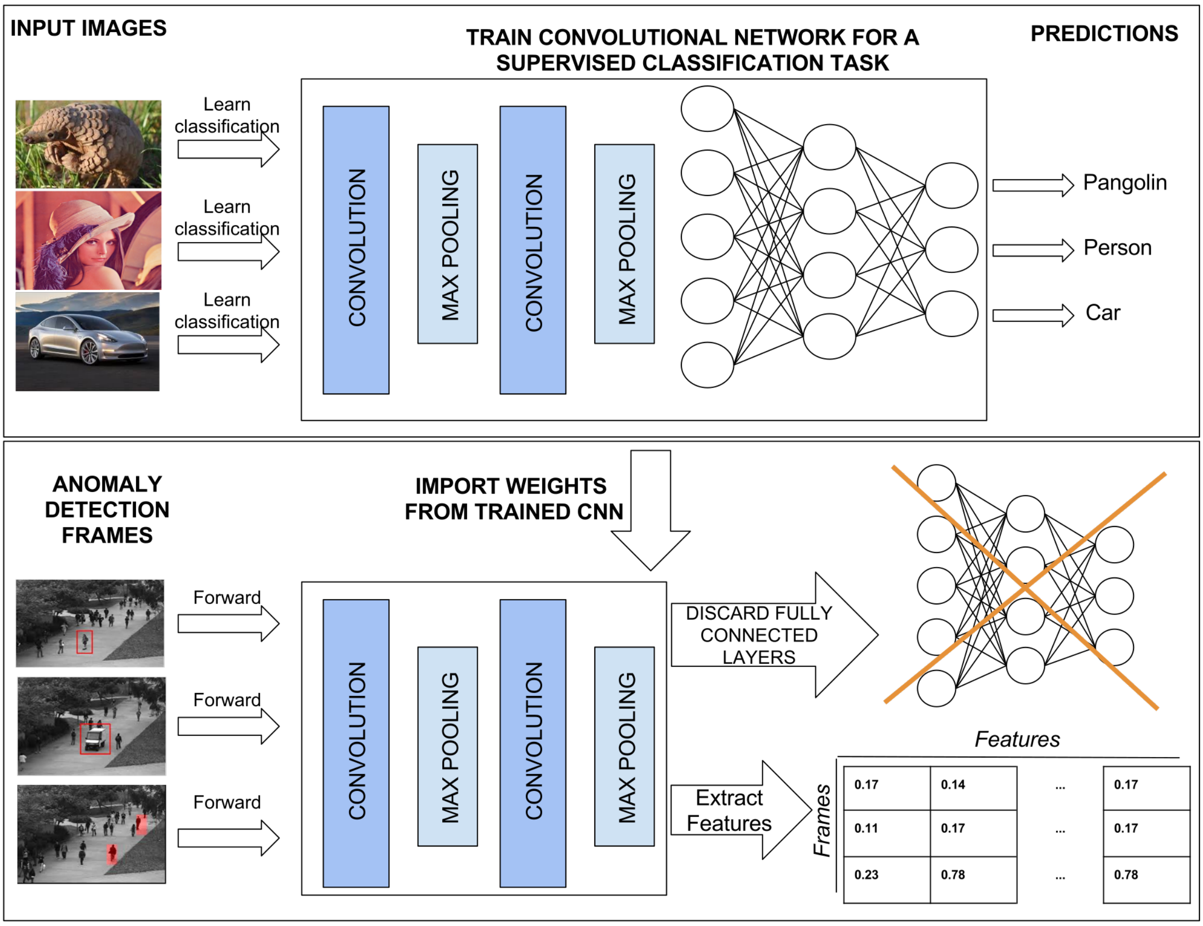


Figure 5: Basic idea of Transfer Learning for image and video anomaly detection.

Some machine learning algorithms work under the premise that the training and testing data belong to the same distribution, which is a strong assumption for real-world problems (SHAO; ZHU; LI, 2015). Contrariwise, TL assumes that these distributions may be different, and a robust model may achieve satisfying results when applied to completely new problems (FUNG et al., 2006; LU et al., 2015).

TL is commonly used when facing two main problems: insufficient computing power to train a large model or the lack of labeled data. The computing power problem is softened by not having to train the model from scratch. For supervised image classification tasks using

CNNs, a fine-tuning process of the last layer is enough to achieve good performance on new problems (YOSINSKI et al., 2014). In case no labeled data is available, the fine-tuning process can be skipped, and the CNN can be used as a feature extractor by forwarding a new image throughout the network and capturing its representation at a given layer. Hence, the lack of labeled data can be relieved by using a model originally trained to solve a similar problem. Such models are easily accessible, given the growing amount of pre-trained models available at the *model zoos*[3] of major DL frameworks.

In this work, we chose to use models trained for classifying images of ImageNet, which is an image dataset used in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (RUSSAKOVSKY et al., 2015). The dataset contains approximately 14 million images and 1000 different classes. Thus, the hypothesis is that the models trained on ImageNet are able to extract robust features, which may also be fit for video anomaly detection problems that include large-scale real-world entities such as vehicles or people.

Whilst the data used to train a model plays an important role in obtaining robust features, the architecture of the model itself also impacts on the features. In general, deeper models have shown better performance than shallow models on image classification tasks, such as the ILSVRC (SZEGEDY et al., 2015, 2016; HE et al., 2015). This performance gain may be attributed to the high-level features, learned at the deeper layers of the network. However, it is not clear if such features are ideal for tasks outside of the scope of the original problem, since they may be overly adapted to it. A possible solution for this problem is to extract features from the middle of the network. Nonetheless, this solution can be computationally costly, since the dimensionality in the middle layers is usually much greater than at the final layers, reaching hundreds of thousands in models such as GoogleNet and ResNet152. Another option is to extract the features from the final layers of shallower networks, which, in theory, produces lower level features whilst keeping dimensionality on a reasonable range.

3.4 AUTOENCODERS

AEs were originally introduced by Rumelhart, Hinton and Williams (1986). In short, AEs are feed-forward Neural Networks with one hidden layer that learn to reconstruct its input at the output. The hidden layer receives a latent representation $\mathbf{h}$, which is a mapping from the input $\mathbf{x}$ using an encoding function $\mathbf{h} = f(\mathbf{x})$. Then, a decoding function $\mathbf{r} = g(\mathbf{h})$ produces the reconstructed input $\mathbf{r}$. In most applications, one is not interested in the reconstructed pattern $\mathbf{r}$, but in the latent representation $\mathbf{h}$. Figure 6 presents the structure of the AE. The input $\mathbf{x}$ is

[3] Model Zoos: Repositories of pre-trained models provided by DL frameworks.

mapped to the latent representation **h** through the encoding function $f(\mathbf{x})$. Then, **h** is used to generate the reconstruction **r** through the decoding function $g(\mathbf{h})$.

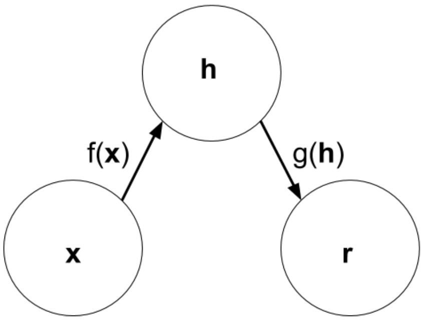


Figure 6: Structure of a basic Autoencoder.
Source: Goodfellow, Bengio and Courville (2016)

AEs are constructed in a way to prevent **x** from being equal to **r**, since such representation would not contain any useful information (GOODFELLOW; BENGIO; COURVILLE, 2016). This is accomplished by imposing constraints to prevent the encoding function from simply copying all values from **x** to **h**. One of the most common approaches to achieve this goal is to have the hidden layer contain fewer neurons than the input layer, forcing the AE to learn only important information for reconstructing the input. This kind of AE architecture is commonly named undercomplete (GOODFELLOW; BENGIO; COURVILLE, 2016).

In contrast to the undercomplete architecture, some approaches employ a greater number of neurons on the hidden layer, defined as overcomplete. In order to prevent the copying problem, overcomplete architectures perform additional tasks besides only reconstructing the input. One of such models is the Denoising Autoencoder (DAE).

DAEs introduce some kind of noise in the input **x**, generating a corrupted input $\widetilde{\mathbf{x}}$ by using a given corruption process , and attempts to reconstruct the original input (prior to the application of noise). In the case of images, this noise can be of many natures, for instance, Gaussian noise, Salt and Pepper noise, or in some cases, applied by using dropout in the input layer (XIE; GIRSHICK; FARHADI, 2016). The process of removing noise whilst learning to reconstruct the input prevents the network from simply learning the identity function, making useful properties arise from the training process (GOODFELLOW; BENGIO; COURVILLE, 2016). Figure 7 shows the basic structure of a DAE.

Similar to deep Neural Networks, it is possible to stack AEs in order to form a Deep Autoencoder, also called Stacked Autoencoder (SAE), or Stacked Denoising Autoencoders

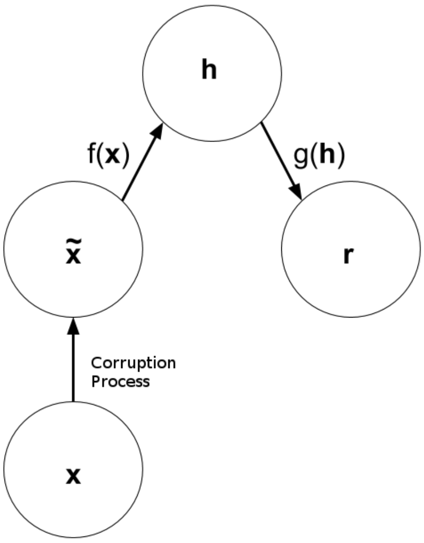


Figure 7: Structure of a Denoising Autoencoder.

Source: Goodfellow, Bengio and Courville (2016)

(SDAE), if stacking DAEs. Stacking AEs or DAEs is beneficial for the same reasons of stacking layers on a Deep Neural Network (GOODFELLOW; BENGIO; COURVILLE, 2016). The universal approximation theorem (HORNIK; STINCHCOMBE; WHITE, 1989) states that a Neural Network with one hidden layer with non-linear activation can approximate any function from a finite dimension to another with any non-zero error, given the hidden layer has enough neurons. However, according to Goodfellow, Bengio and Courville (2016), it is more efficient to add new layers instead of adding neurons to the hidden layer. This way, the same generalization can be achieved with fewer neurons in total.

Deep architectures based on fully connected layers, such as SAEs and SDAEs have some drawbacks when dealing with images (especially large ones). The first one is the loss of spatial correlations, since the images are squeezed into one-dimensional vectors. The second is the high computational effort required when dealing with high-resolution images, since the number of parameters (weights) escalate quickly in fully connected architectures (MASCI et al., 2011).

Following the same principles of the CNN, the Convolutional Autoencoder (CAE) allows to tackle these problems. Like CNNs, CAEs are deep models, containing convolutional and pooling layers. Moreover, two different operations, deconvolution and unpooling, are also used in CAEs. These operations are performed in the decoding step, and perform the reverse operations of convolutional and pooling layers, respectively. Figure 8 displays the structure of a CAE, along with the shape of the data after each layer.

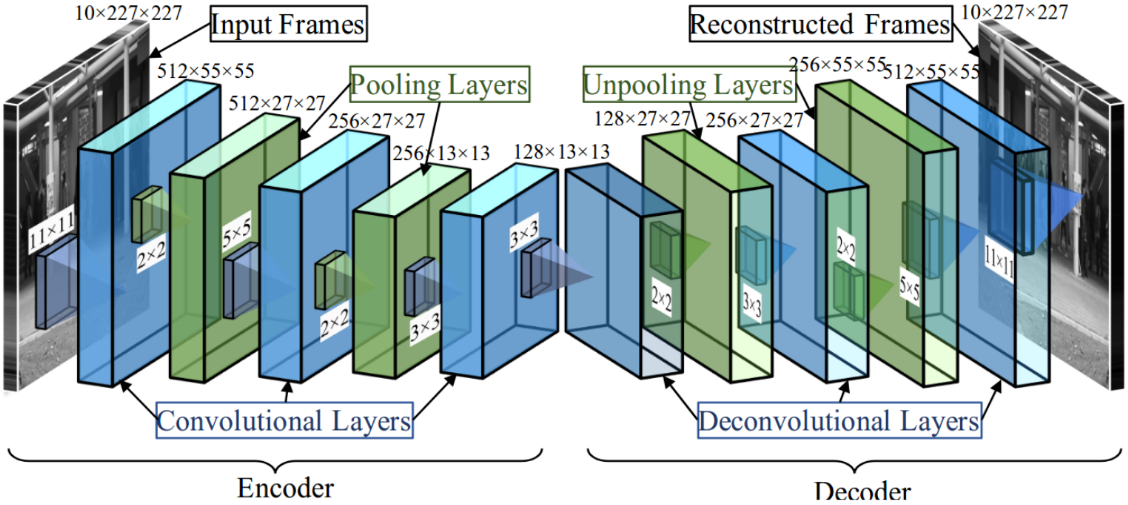


Figure 8: Structure of a Fully Convolutional Autoencoder.

Source: Hasan et al. (2016)

Fully convolutional AEs do not contain a fully connected bottleneck layer, and their uses are primarily related to the reconstructed image, such as measuring the reconstruction error (HASAN et al., 2016). In contrast, hybrid architectures can employ bottleneck layers formed by fully connected neurons, besides the convolutional and pooling layers. This allows to use the network to extract the latent representation of the data, in the same way as AEs and DAEs, whilst dealing with high-resolution images.

AEs and variants are trained in the same way as other feed-forward Neural Network. A loss function measures the dissimilarity between the input and the reconstruction, such as Mean Squared Error (MSE), as presented in Equation 15, where $\hat{\mathbf{y}}$ is the output vector of the decoder, $\mathbf{y}$ is the ground truth vector and n is the number of image samples. Then, the weights of the network are updated through a training process using GD, as described in Section 3.2.

$$L_{MSE}(\mathbf{y},\hat{\mathbf{y}}) = \frac{1}{n}\sum_{i}(y_i - \hat{y}_i)^2. \tag{15}$$

In the convolutional case, the loss function can be computed by measuring the distance between the input and output images. The loss function can be defined as:

$$L_{conv}(\mathbf{X},\hat{\mathbf{X}}) = \frac{1}{n}\sum_{i}||X_i - \hat{X}_i||_2^2, \tag{16}$$

where X_i is the i-th original image, $\hat{X}_i$ is the i-th reconstructed image, and n is the number of samples.

3.5 COMPACT FEATURE REPRESENTATION

Building a model of normality from images and videos requires representing the scene in a feature space. As discussed before, this is done by using some sort of feature extraction process, which attempts to describe different aspects of the target data. In image and video anomaly detection, multiple traits of normality need to be represented in the feature space. For instance, a pedestrian walking on a sidewalk and a cyclist riding a bike could both be considered normal under a certain context. However, their representations may occupy different zones on the feature space, due to differences in appearance and motion patterns.

This example illustrates one important aspect of anomaly detection: different aspects of normality may form separate clusters in the feature space (XU et al., 2014; BODESHEIM et al., 2013). Hence, it may be desirable to approach the problem from a multiple clusters perspective, instead of trying to group all traits of normality on a single cluster. Figure 9 shows a scenario where multiple clusters of normality were formed by the One-Class classifier. In the illustration, it is possible to notice that anomalies may be located in between the regions of normality. Anomalous events are represented by red triangles and diamonds. It is also possible to see some classification errors.

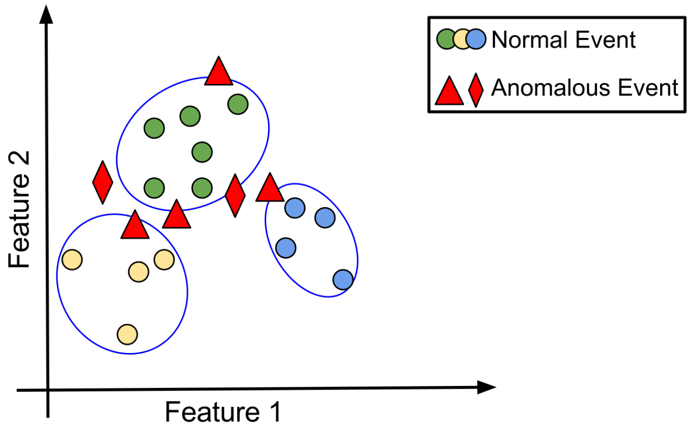


Figure 9: Example of the representation of normal and anomalous events on the feature space using multiple clusters of normality.

From a clustering perspective, it is often desirable that data points within the same cluster are as close as possible to each other, whilst the distance between centroids (cluster centers) should be as great as possible. Following this idea, compact feature representations aim at improving these two aspects of clustering. In the context of anomaly detection, the objective is to make the normal clusters as compact and separated as possible. The hypothesis is that compactness increases classification performance by making anomalous events more distinguishable from normal events in the feature space. Figure 10 presents the same scenario as seen before, in Figure 9, with increased compactness. Note that compact feature representation refers to the between and within cluster distances, and should not be confused with compact representations in the sense of a small feature vector.

The idea behind the compact feature representation is simple. However, devising a method for extracting such features is not a trivial task. Many efforts have been made to achieve compactness in the representations (WEINBERGER; SAUL, 2009; BODESHEIM et al., 2013; ZHANG; XIANG; GONG, 2016). Recently, a DL approach named Deep Embedded Clustering (DEC) (XIE; GIRSHICK; FARHADI, 2016) was devised for extracting compact features, and has shown promising results on unsupervised image classification tasks. However, its application in the anomaly detection context has not been explored in the literature. Hence, this work studies the application of DEC for the image and video anomaly detection problem.

DEC is a SDAE based method, which learns compact feature representations from raw data in a two-steps procedure. The first step is the same as in regular AEs: learning to reconstruct the input at the output. The second step is a fine tuning process, which incrementally introduces compactness in the bottleneck feature representation.

After the first step is completed, i.e., the weights are correctly adjusted for reconstructing the input, the decoder part of the network is discarded. Then, the algorithm simultaneously tries to learn a set of K cluster centers $\{\boldsymbol{\mu}_k\}_{k=1}^{K}$ and the weights of the SDAE that perform the mapping into the bottleneck. In the first iteration, cluster centers are initialized using the K-Means algorithm.

The optimization is performed by iteratively alternating between computing a soft assignment q_{ij} between each point in the bottleneck and the cluster centers, and then updating

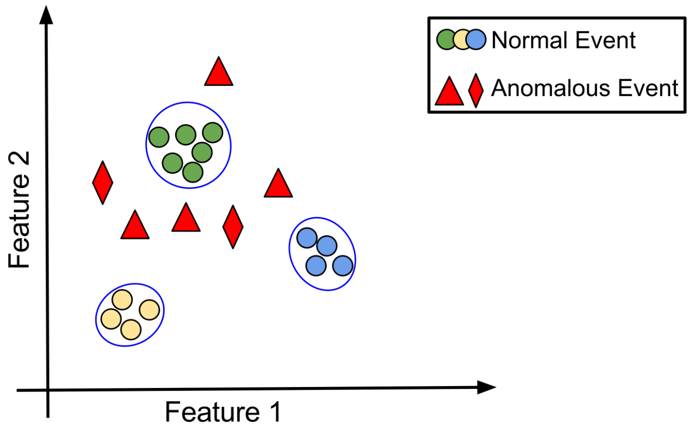


Figure 10: Example of the representation of normal and anomalous events on the feature space using multiple clusters of normality with increased compactness.

the weights and cluster centers $\{\boldsymbol{\mu}_k\}_{k=1}^{K}$ using an auxiliary target distribution p_{ij}, which is based on high confidence cluster assignments. The loss function L_{KL} to be minimized is the Kullback-Leibler (KL) divergence KL between q_{ij} and p_{ij} (van der Maaten, 2009), as shown in Equation 17.

$$L_{KL} = KL(P\|Q) = \sum_i \sum_j p_{ij} \log \frac{p_{ij}}{q_{ij}}. \tag{17}$$

The soft assignment q_{ij} is the probability of a sample $\mathbf{z}_i$ belonging to cluster $\boldsymbol{\mu}_k$, as shown in Equation 18.

$$q_{ij} = \frac{\sum_k \left(1 + \|\mathbf{z}_i - \boldsymbol{\mu}_k\|_2^2\right)}{1 + \left\|\mathbf{z}_i - \boldsymbol{\mu}_j\right\|_2^2}. \tag{18}$$

In its turn, the auxiliary distribution p_{ij} is computed from the soft assignments, as shown in Equation 19.

$$p_{ij} = \frac{q_{ij}^2 / \sum_m q_{mj}^2}{\sum_k \left(q_{ik}^2 / \sum_m q_{mk}^2\right)}. \tag{19}$$

The auxiliary distribution is used with the intent of improving cluster purity, making points assigned with high confidence have greater impact in the optimization. Also, it normalizes the loss contribution of each centroid, preventing large clusters from distorting the bottleneck representation (XIE; GIRSHICK; FARHADI, 2016).

The parameters, as well as the cluster centers, are updated using SGD. The gradient regarding the weights can be written as shown in Equation 20:

$$\frac{\partial L_{KL}}{\partial \mathbf{z}_i} = 2 \sum_k \frac{\left(p_{ij} - p_{ij}\right)\left(\mathbf{z}_i - \boldsymbol{\mu}_k\right)}{\left(1 + \|\mathbf{z}_i - \boldsymbol{\mu}_k\|_2^2\right)}. \tag{20}$$

In its turn, the gradient regarding the cluster centers can be written as shown in Equation 21:

$$\frac{\partial L_{KL}}{\partial \boldsymbol{\mu}_k} = -2 \sum_i \frac{\left(p_{ij} - p_{ij}\right)\left(\mathbf{z}_i - \boldsymbol{\mu}_k\right)}{\left(1 + \|\mathbf{z}_i - \boldsymbol{\mu}_k\|_2^2\right)}. \tag{21}$$

Figure 11 illustrates the process of optimization. The top part of the figure describes a regular SAE optimization process by minimizing the reconstruction error. The bottom part of the figure illustrates the DEC optimization by clustering the data points and minimizing the Kullback-Leibler divergence KL between the distributions q and p.

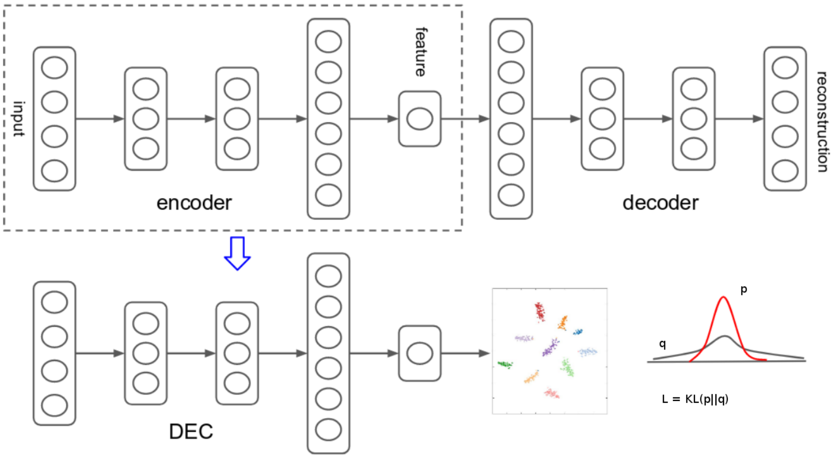


Figure 11: Deep Embedded Clustering optimization process.

Source: Xie, Girshick and Farhadi (2016)

4 PROPOSED METHODS

This Chapter proposes two methods for performing anomaly detection in images and videos. The first method is a Transfer Learning approach, in which a CNN trained for a completely different task is used for extracting features from images and videos related to anomaly detection. The second method is feature learning with a compact representation using DEC, also applied to image and video anomaly detection.

Both methods follow the same basic steps: feature extraction followed by classification using OC-SVM. Hence, the feature learning and extraction process is the main focus of this work. A detailed explanation regarding each method is presented in the following Sections.

4.1 TRANSFER LEARNING FOR IMAGE AND VIDEO ANOMALY DETECTION

As briefly discussed in Chapters 2 and 3, transferring knowledge between tasks is becoming ever more popular. This is commonly referred to as Transfer Learning (TL). Some of the factors that contribute to the growing usage of TL are:

- Models can be easily used, due to the high-level abstraction provided by DL frameworks;
- A wide variety of models trained for different tasks are available at repositories provided by DL frameworks (model zoos);
- TL requires low computational power, since the model's weights are already optimized, and, at most, requires a fine-tuning of the last layers;
- Features extracted in this way have shown to be robust across many different tasks (SMEUREANU et al., 2017b).

TL also contains some limiting aspects. For instance, the input shape of a pre-trained network has already been defined. Hence, it is not possible to adapt the network to the new data, forcing the user to do the inverse: adapt the data to match the specific input shape of the network. In images, this shape is usually composed of three dimensions: width, height, and a number of channels. The first two dimensions (width and height) are usually corrected by employing an interpolation algorithm to resize the images, as done by Smeureanu et al. (2017b). The last dimension mismatch (number of channels) is easily solvable when the new data contains, for instance, three channels (RGB colored image) and the network requires an input with one channel (grayscale image). In this case, a simple conversion from RGB to grayscale on the new images solves the problem. However, the opposite problem may be more troublesome. For instance, the network requires images with three-channels, and the new data contains only grayscale images. In this case, one can replicate the grayscale pixels to all three channels, creating a gray image with three color dimensions. This work utilizes both of the presented strategies for dealing with the input shape mismatches.

An important question that arises when using TL is: among all available pre-trained models, which one is the most suitable for my application? In fact, there is still no definitive answer to this question. One of the ways to approach the problem is by choosing a model that has been trained for a similar task or using similar data. In the particular case of image recognition tasks, a good choice may be to use CNNs trained for the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), which, as stated in Chapter 3, contains a very large amount of images divided into a thousand classes. In this work, it is hypothesized that features extracted from ILSVRC CNNs may be robust across a wide variety of tasks, including the image and video anomaly detection task.

However, one problem remains. Within the array of models trained for the ILSVRC, there are dozens of different CNN architectures, which vary in size and complexity. Moreover, these models achieved different classification accuracies. A trend that can be seen in the ILSVRC is that, historically, deeper models have achieved higher classification accuracies than shallower models (KRIZHEVSKY; SUTSKEVER; HINTON, 2012; SZEGEDY et al., 2015). Nevertheless, it is not clear if that trend actually applies to TL in the context of image and video anomaly detection.

Hence, in this work, feature extraction is performed with eleven CNNs, all of which were pre-trained on the ILSVRC. These models contain a varied amount of layers, convolution operations, output dimensions and classification errors. Table 2 shows information about the models. “Top 5 error” refers to the classification error on the ILSVRC. It is calculated based on the top 5 class predictions of the network. If one of these predictions matches the actual image label, it is considered a hit. “Convolutions” refers to the number of weighted convolutional layers on the network. “N Layers” refers to the total number of weighted layers, and “N

Features" refers to the number of features the network outputs in the final pooling layer. This layer has been chosen to extract features for reasons discussed in Section 3.3. The references to the models are shown in Table 3.

Table 2: General information about the models used in this work.

	GoogleNet	InceptionV3	ResNet10	ResNet50	ResNet152	VGG-S	VGG-M	VGG-F	VGG-16	VGG-19	AlexNet
Top 5 error	11,1	6,1	14,8	7,6	5,7	13,1	13,7	16,7	8,1	8,8	18,1
Convolutions	57	106	12	53	155	5	10	5	13	16	5
N Layers	22	48	10	50	152	8	16	8	16	19	8
N Features	1024	2048	512	2048	2048	18432	18432	9216	25088	25088	9216

Table 3: References of the models used in this work.

Model	Reference
GoogleNet	Szegedy et al. (2015)
InceptionV3	Szegedy et al. (2016)
ResNet10	He et al. (2015), Simon, Rodner and Denzler (2016)
ResNet50	He et al. (2015), Simon, Rodner and Denzler (2016)
ResNet152	He et al. (2015)
VGG-S	Chatfield et al. (2014)
VGG-M	Chatfield et al. (2014)
VGG-F	Chatfield et al. (2014)
VGG-16	Chatfield et al. (2014)
VGG-19	Chatfield et al. (2014), Simon, Rodner and Denzler (2016)
AlexNet	Krizhevsky, Sutskever and Hinton (2012), Simon, Rodner and Denzler (2016)

An overview of the proposed method is presented in Figure 12. Each block represents a step for detecting video anomalies and will be detailed in the next Sections. First, frames are preprocessed aiming at adapting to the shape required by the pre-trained CNN model. Then, frames are forwarded throughout the model for capturing the representations at a given layer. The representations extracted from the training dataset are then used to optimize an OC-SVM. The optimized One-Class SVM is then used for calculating an anomaly score for each test frame. The anomaly scores are then smoothed using a moving average filter and are finally used for classification.

4.1.1 DATA PREPROCESSING AND FEATURE EXTRACTION

All videos were discretized into a sequence of frames covering the full video duration. Each frame is previously labeled as normal or anomalous by a human expert. The benchmark

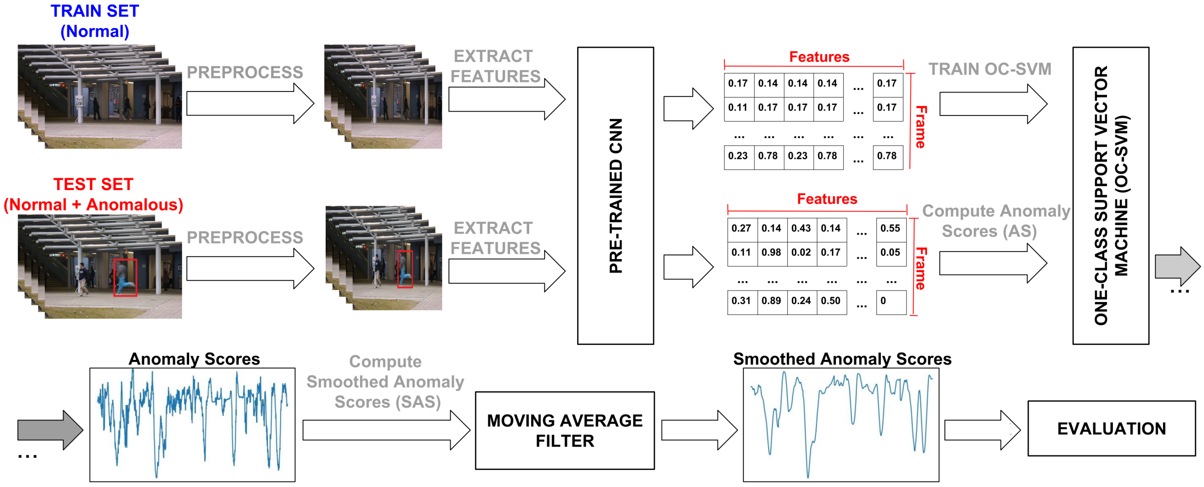


Figure 12: Overview of the proposed video anomaly detection method.

datasets used in this work will be detailed in Chapter 5. Due to the input size limitations discussed earlier in this Section, each frame has its size and channels adapted to the required shape.

Feature extraction was done by forwarding each frame throughout each network and capturing the information at the last pooling layer. No fine tuning process was done before the feature extraction, i.e. the original weights were used without any further adjustment. It is important to note that the dimensionality at the last pooling layer is significantly smaller when compared to any other convolutional or pooling layer in the network, making the process computationally feasible.

4.1.2 CLASSIFICATION

Once the features are extracted, the classification process is performed by the OC-SVM. Only normal examples are used in the training phase. Since the OC-SVM is dependent on the choice of appropriate parameters, a factorial experiment was used to define the best values for the trade-off between empirical and structural risks parameter C (Section 3.1, Equation 3) and the RBF kernel width parameter υ (Section 3.1, Equation 5). The search values for both parameters were $\{0.001, 0.01, 0.05, 0.1, 0.2, 0.5, 1.0\}$ and $\{0.001, 0.01, 0.05, 0.1, 0.2, 0.5, 0.9\}$, respectively. The results produced by the best combination of parameters will be presented in Chapter 5.

Once the OC-SVM optimization is finished, an anomaly score AS_i is computed for each test sample by using Equation 8 (Section 3.1). The AS_i is a distance measured from a test data point to the border of the hypersphere, i.e. how far the sample is from the decision border in the kernel hyperspace.

4.1.3 MOVING AVERAGE FILTER

In videos, anomalies generally occur over time, which means that temporal factors are relevant for the classification task. Hence, a simple moving average filter is used to remove noise in the classification process.

For each frame on a continuous video sequence, a Smoothed Anomaly Score SAS_i is proposed, which is the smoothed AS_i. The SAS_i is computed by using a moving average filter, according to Equation 22, where s is the size of the moving average mask and i is the current frame.

$$SAS_i = \frac{1}{s}\sum_{j=0}^{s-1} AS_{i+j}. \tag{22}$$

This strategy ensures that the current frame SAS_i is influenced by forthcoming frames. Since s is a user-defined parameter, experiments with a set of different values were performed, and the one that performs best was selected. The set of parameters includes the arbitrarily chosen values $\{2,3,5,7,10,15,20,30,50,75,100,150,200\}$. Since most of the datasets used in this work provide a sequence of different videos that have time gaps between them (cuts and changes of scenario), the SAS_i is calculated over continuous (uncut) video sequences only.

In a real-life application, this filter has the downside of forcing the anomaly detection system to operate s frames behind the real-time footage. However, for most applications, this small time gap would not be critical.

4.1.4 EVALUATION

For evaluation of results, the Area Under the Receiver Operating Characteristic (ROC) Curve (AUC) is used. This measure ensures that the classifier is evaluated at many different thresholds, i.e. distance cutting points above which a sample is considered anomalous. The ROC curve is a graphical plot that describes the performance of a binary classifier at different decision thresholds. In the plot, the y-axis describes the True Positive Rate (sensitivity), whilst the x-axis describes the False Positive Rate (1-specificity). The AUC measures the area under this curve, where the larger the area the better the classifier. Figure 13 shows an example of ROC curve and the AUC calculated.

Moreover, the Equal Error Rate (EER) is computed from the ROC curve, which indicates the best balance between sensitivity and specificity. The use of EER as evaluation

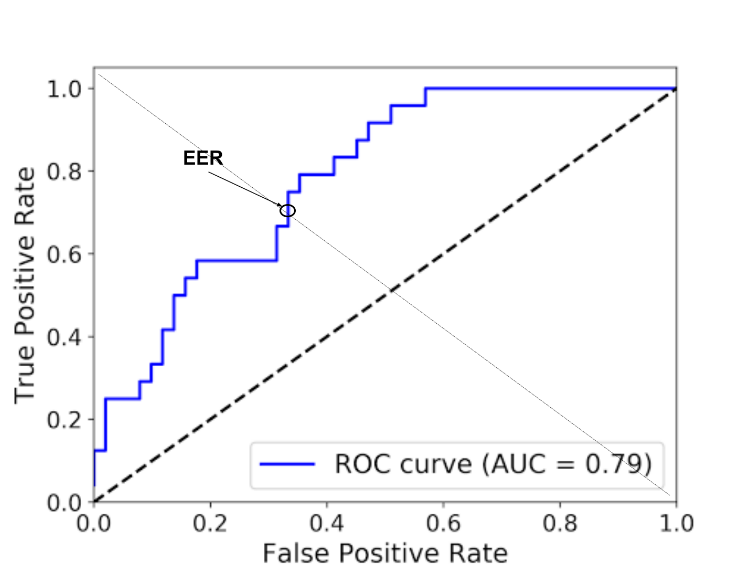


Figure 13: Example of a Receiver Operating Characteristic Curve

measure is a topic of debate in current video anomaly detection literature (SMEUREANU et al., 2017b). Whilst it is true that anomalies are not expected to happen as often as normal events, and, therefore, the appropriate threshold should be application specific, the EER provides a standard threshold for comparing classifiers, which is one of the objectives of this work. Moreover, the EER has been used in many works in the video anomaly detection literature (RIBEIRO; LAZZARETTI; LOPES, 2017)(SALIGRAMA; CHEN, 2012)(XU et al., 2015).

4.2 COMPACT FEATURE LEARNING USING DEEP EMBEDDED CLUSTERING

The second method proposed in this work is compact feature learning using DEC applied to image and video anomaly detection. This section is divided into two parts. The first part describes the method used in a preliminary experiment using well-known image classification benchmark datasets that are unrelated to anomaly detection. In order to make the datasets fit the OCC context, labels were modified such that certain classes are considered normal and some are considered anomalous. The second part of this section describes the method used for the main experiments, in which six anomaly detection benchmark video datasets were used for evaluation.

4.2.1 PRELIMINARY METHOD FOR COMPACT FEATURE LEARNING USING STACKED DENOISING AUTOENCODERS

The method described in this Section was devised as a proof of concept that, by learning compact representations from images, a higher classification performance may be achieved on anomaly detection problems. Figure 14 shows an overview of the proposed method, to be described next.

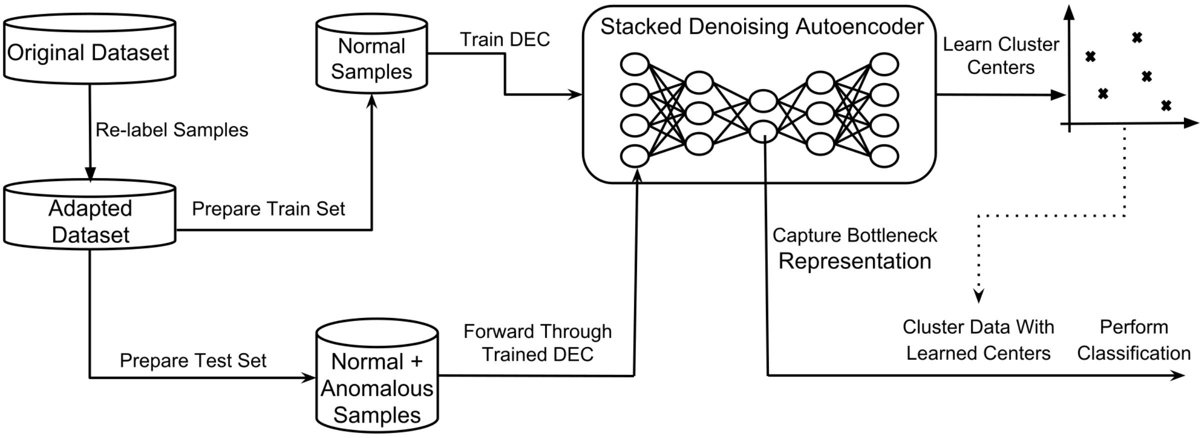

Original Dataset
Re-label Samples
Adapted Dataset
Prepare Train Set
Normal Samples
Train DEC
Stacked Denoising Autoencoder
Learn Cluster Centers
Capture Bottleneck Representation
Prepare Test Set
Normal + Anomalous Samples
Forward Through Trained DEC
Cluster Data With Learned Centers
Perform Classification

Figure 14: Overview of the preliminary method of compact feature learning for anomaly detection.

4.2.1.1 DATA PREPROCESSING

The preprocessing step requires adjusting annotations of the image data to fit in the anomaly detection context. To do so, this work selects a subset of the existing classes to be considered normal. This subset is chosen according to some structural or semantic similarity. For instance, all classes that contain animals were considered normal, whilst all classes that contain something else were considered anomalous. A more detailed explanation of the relabeling process specific for each dataset will be discussed in Section 5.2.1.

The next step is to generate the train and test sets. The train set consists exclusively of normal samples. This set contains a randomly selected fraction of the total normal image pool. The remaining normal samples are included in the test set, along with all other anomalous images. Therefore, each image appears exclusively on one set, i.e. images used for training do not appear in the test set. Finally, the train set is shuffled.

4.2.1.2 TRAINING DEC

The DEC presented in this method employs a SDAE network. The architecture of the model is the same as proposed by Xie, Girshick and Farhadi (2016). The input layer size varies according to the dataset since images have different sizes. The hidden layers contain 500-500-2000 neurons, respectively, followed by the bottleneck layer, which also has a variable size of either 10 or 100, depending on the experiment. The decoder is composed of the exact reverse structure of the encoder. The ReLU activation function was used for all neurons.

As discussed in Chapter 3, the DEC training consists of two steps. In the first step, the SDAE is trained layer by layer aiming at minimizing the reconstruction error. In order to insert noise into the input of the SDAE, the dropout method was employed after the encoder layers with a drop probability of 20%. In this step, the SDAE is trained for 1000 epochs using SGD with a learning rate of 0.1. These parameters were chosen according to the work presented in Hasan et al. (2016).

Once the first part of the optimization finishes, the decoder part of the network is discarded, and the second part takes place. This step aims at improving feature compactness in the bottleneck, as discussed in Chapter 3. With the same set of parameters as the first part, DEC is trained for 150 epochs, which was chosen empirically. However, the weights learned and cluster centers were stored after every epoch in order to evaluate the model at different time steps.

4.2.1.3 CLASSIFICATION

Once DEC is fully optimized, the test set is forwarded throughout the network in order to capture the latent bottleneck representation and cluster centers. Then, the probability q_{ij} of each sample belonging to each cluster center is computed by using Equation 18 (Section 3.5). Each sample is then assigned to the cluster with the highest probability, which was also used as an anomaly score. The method is evaluated at different thresholds by using the AUC, as described in Section 4.1.

4.2.2 MAIN METHOD FOR COMPACT FEATURE LEARNING USING CONVOLUTIONAL AUTOENCODERS

This Section describes the method used in the main video anomaly detection experiment using compact feature representations. Figure 15 presents an overview of the method. It is similar to that presented in Section 4.2.1, in which the main difference is the use of a CAE, instead of a SDAE. Each step in the process is described in the following Sections.

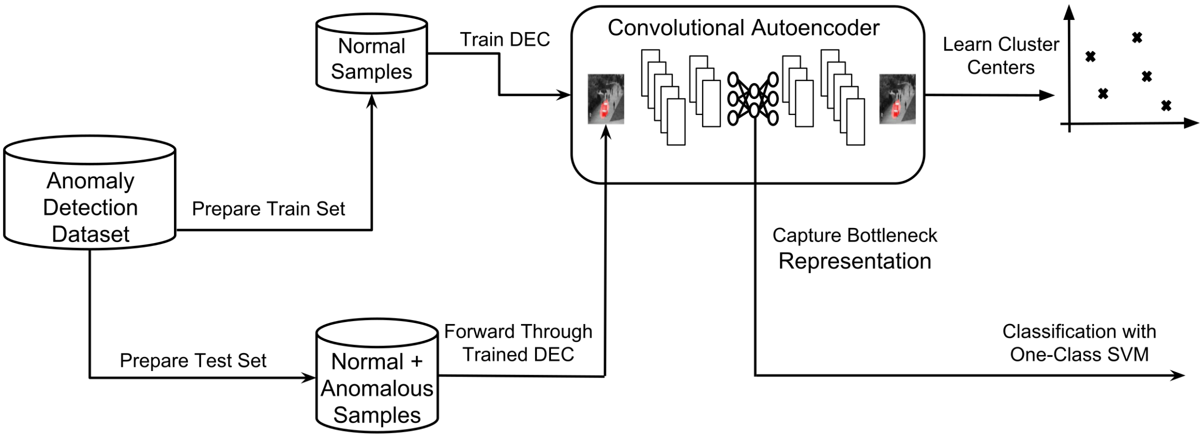

Normal Samples
Train DEC
Convolutional Autoencoder
Learn Cluster Centers
Anomaly Detection Dataset
Prepare Train Set
Capture Bottleneck Representation
Prepare Test Set
Normal + Anomalous Samples
Forward Through Trained DEC
Classification with One-Class SVM

Figure 15: Overview of the main method of compact feature extraction for anomaly detection.

4.2.2.1 DATA PREPROCESSING

All video anomaly detection benchmark datasets were first discretized into frames and converted to grayscale. Moreover, all datasets were resized to 235 × 155 pixels, following the idea presented in Hasan et al. (2016). The train set contains only instances of the normal class, whilst the test set contains instances of both classes.

4.2.2.2 TRAINING THE CONVOLUTIONAL DEC

Since fully connected architectures are not suitable for large images, as discussed in Section 3.4, a Convolutional architecture for DEC is proposed in this work, named Convolutional Autoencoder with Compact Embedding (CAE-CE). The convolutional model has some advantages over the standard fully connected model, such as the preservation of spatial relations and smaller computational effort.

The SDAE is replaced with a CAE, whose architecture is presented in Table 4.

The architecture contains three convolutional and two pooling layers on the encoder side, followed by three fully connected layers and the bottleneck. The decoder follows the exact inverse structure. The architecture was based on that presented by Hasan et al. (2016), which was also successfully used by Ribeiro, Lazzaretti and Lopes (2017).

The procedure for optimizing the CAE-CE is the same as presented in Section 4.2.1. In the first step, the CAE is optimized through SGD and backpropagation for reconstructing the input. This step lasts until the optimization has stagnated.

The second step is the joint optimization of the cluster centers and the parameters of the network, aiming at achieving compact representations. Cluster centers are initialized with the

Table 4: Architecture of the Convolutional Autoencoder.

Encoder

Layer	Size	Dimension
Input	-	1 x 235 x 155
Conv. 1	11x11	256 x 57 x 37
Pool. 1	3x3	256 x 28 x 18
Conv. 2	5x5	128 x 28 x 18
Pool. 2	3x3	128 x 14 x 9
Conv. 3	3x3	64 x 14 x 9
Fully 4	-	2016
Fully 5	-	504
Fully 6	-	168
Bottleneck	-	50

K-Means algorithm, as done in Section 4.2.1. The set of $\{2,3,4,5,10\}$ cluster centers were used during the optimization, and the parameter that presented the best performance was selected.

4.2.2.3 FEATURE EXTRACTION, CLASSIFICATION AND EVALUATION

Feature extraction is done by forwarding the train and test frames through the network and capturing the latent bottleneck representation. The features of the train set are then used to train an OC-SVM. The choice of parameters, classification and evaluation are identical to those presented before, in Section 4.1.2 and Section 4.1.4.

5 EXPERIMENTS AND RESULTS

All experiments done in this work were run on a computer with an Intel Core i7 processor at 3.30GHz, two Nvidia Titan Xp GPUs and a minimal installation of Ubuntu 14.04 LTS. All software was developed using the Python programming language. Moreover, the TensorFlow (ABADI et al., 2016), Caffe (JIA et al., 2014), Theano (Theano Development Team, 2016), and Lasagne (DIELEMAN et al., 2015) Deep Learning frameworks were used to implement the DL models.

This Chapter is organized as follows: Section 5.1 presents the experiments and results obtained using the TL approach. Section 5.2 presents the experimental results on the preliminary study of compact feature learning applied to static image anomaly detection using a SDAE. Section 5.3 presents the experimental results using the CAE-CE architecture for anomaly detection in videos. Section 5.4 presents a comparison of the results obtained by each method, as well as a discussion. Each experiment was performed following the method presented in Chapter 4.

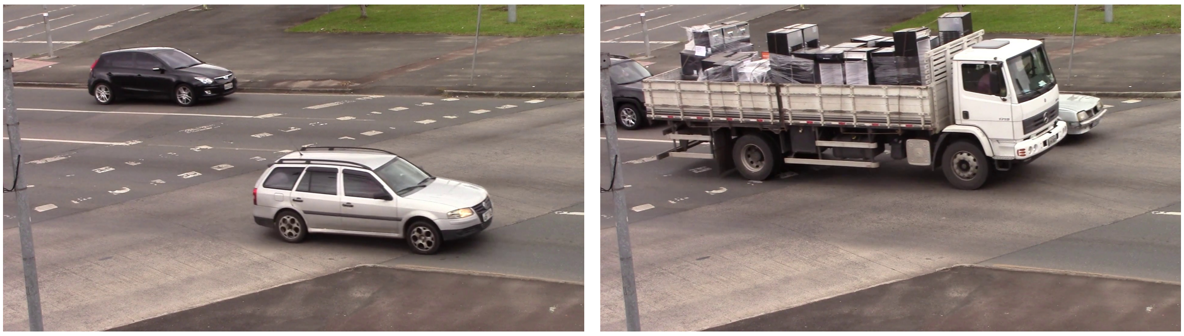
5.1 TRANSFER LEARNING APPROACH

This Section presents the experimental results obtained by using pretrained CNNs on ImageNet as feature extractors for video anomaly detection. It is organized as follows: Section 5.1.1 presents a description of the datasets used to evaluate the method, and Section 5.1.2 presents the results obtained with multiple CNNs as well as an analysis and a discussion.

5.1.1 DATASETS

Six video anomaly detection benchmark datasets were used for evaluating the performance of the proposed method:

LABIC3 and **LABIC4**[1] are video anomaly detection datasets aimed at detecting traffic anomalies on busy highways. They were shot at Curitiba, Brazil, by the Laboratory of Bioinformatics and Computational Intelligence (LABIC) of the Federal University of Technology, Paraná. In order to avoid traffic jams, trucks and other large vehicles are restricted at certain times of the day. Therefore, scenes containing only small vehicles are considered normal, whereas scenes containing at least one large vehicle are considered anomalous. LABIC3 and LABIC4 differ due to the differences in camera angle, elevation, and location. The LABIC3 dataset is composed of 6602 frames in the training set, and 1660 frames in the test set. In its turn, LABIC4 contains 5640 frames in the training set and 1986 frames in the test set. Samples representing the normal and anomalous situations of LABIC3 are presented in Figure 16.

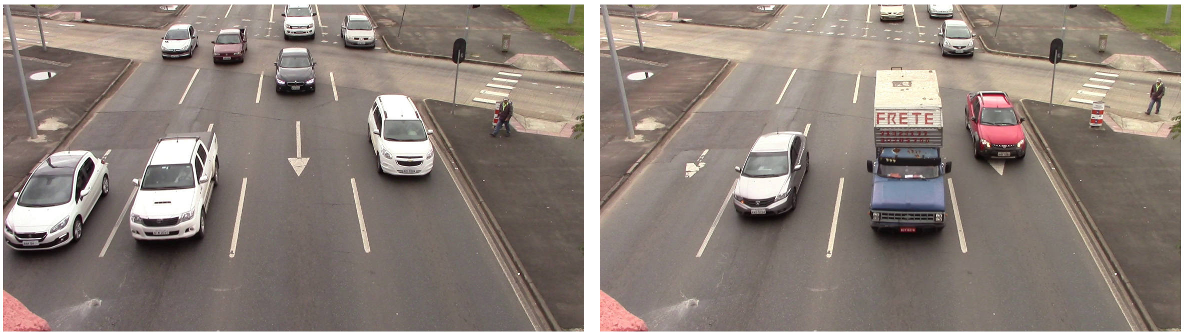


(a) Normal

(b) Anomalous

Figure 16: Normal and anomalous events in the LABIC3 dataset

In this scenario, (b) was considered anomalous because it contains a large vehicle on the scene. Similarly, LABIC4 samples are presented in Figure 17. The anomaly is also characterized by a truck.



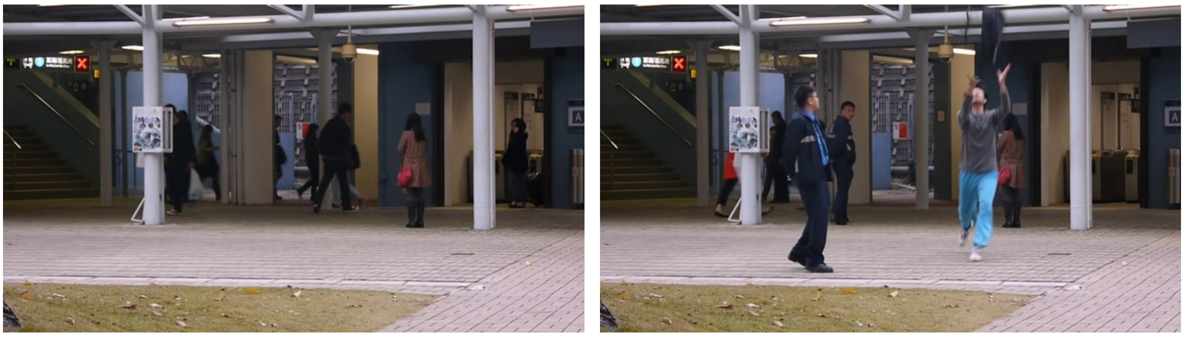
(a) Normal

(b) Anomalous

Figure 17: Normal and anomalous events in the LABIC4 dataset

[1] LABIC3 and LABIC4 Datasets: found at `http://bioinfo.cpgei.ct.utfpr.edu.br/wordpress2/en/softwares/`

The **Avenue** dataset[2] (LU; SHI; JIA, 2013) was created at the Chinese University of Hong Kong and currently serves as a benchmark for video anomaly detection. Normal events are defined by people walking in different directions. Anomalous events occur when people run, throw objects or loiter. The training dataset contains about 15328 frames, whilst the testing dataset contains about 15324 frames. Figure 18 presents normal and anomalous situations in the Avenue dataset. In this specific case, the anomaly is characterized by a person throwing a bag into the air.



(a) Normal (b) Anomalous

Figure 18: Normal and anomalous events in the Avenue dataset

UCSD Ped1 and **Ped2**[3] (MAHADEVAN et al., 2010) are video anomaly detection datasets captured by stationary cameras in a pedestrian walkway. Walking pedestrians are considered normal events, and anomalies occur when vehicles, skateboarders, bicycles, and wheelchairs pass throughout the scene. The dataset is split into Ped 1 and Ped 2, and their main differences are the camera angles with respect to the walkway and the location itself. Ped 1 contains about 5500 normal frames and 3400 anomalous frames. In its turn, Ped 2 contains 364 normal frames and 1652 anomalous frames. Figure 19 displays normal and anomalous events in the UCSD Ped1 dataset.

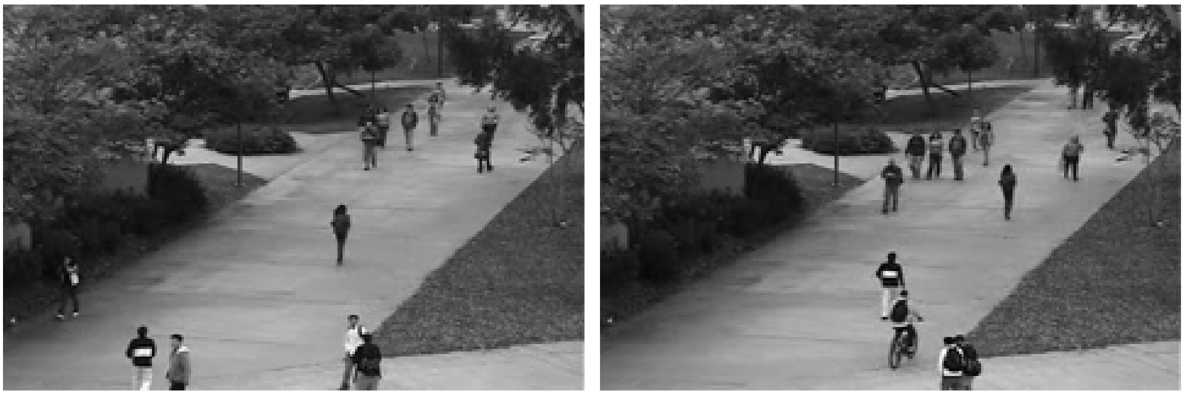
The anomaly in this case is due to the biker entering the scene in the botton of the image. Similarly, scenes of the UCSD Ped2 dataset are shown in Figure 20. In this case, the anomaly is characterized by both the biker and the skater going through the scene.

The **UMN** dataset[4] (MEHRAN; OYAMA; SHAH, 2009) contains footage of crowd behavior in three different locations. Normal events are defined by people randomly walking, whilst anomalies occur when the crowd evades the scene by running radially outwards. The dataset is split in the following way: 5122 frames belong to the training set and 1382 frames

[2] Avenue Dataset: found at `http://www.cse.cuhk.edu.hk/leojia/projects/detectabnormal/dataset.html`

[3] UCSD Ped1 and UCSD Ped2 Datasets: found at `http://www.svcl.ucsd.edu/projects/anomaly/dataset.html`

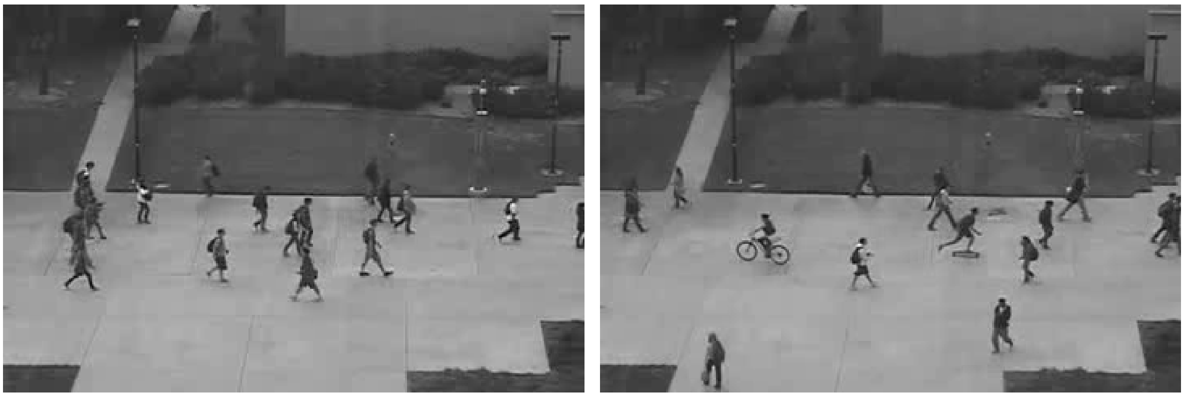
[4] UMN Dataset: found at `http://mha.cs.umn.edu/proj_events.shtml`



(a) Normal

(b) Anomalous

Figure 19: Normal and anomalous events in the UCSD Ped1 dataset



(a) Normal

(b) Anomalous

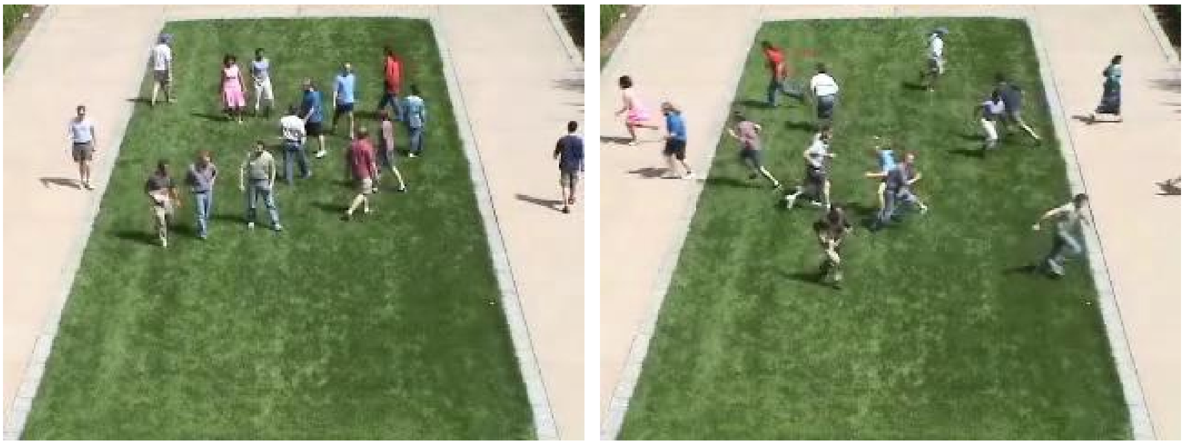
Figure 20: Normal and anomalous events in the UCSD Ped2 dataset

belong the test set, 436 of which are anomalies. It is important to note that this dataset does not have predefined train and test sets, therefore this split may be different from those used in other works. Figure 21 presents normal and anomalous situations in this dataset. The anomalous situation is caused by people evading the scene by running.

5.1.2 EXPERIMENTS AND RESULTS

The experiments presented in this Section aims at answering important questions that arise in TL applications:

- which model should be chosen for the video anomaly detection problem?
- does a model that performs better on its original task also performs similarly on the new task?
- what is the performance of TL in contrast with other methods in the literature?



(a) Normal

(b) Anomalous

Figure 21: Normal and anomalous events in the UMN dataset

As an attempt to answer these questions, a comparison was made, regarding the performance of eleven different pretrained CNN models that were used as feature extractors for the video anomaly detection problem. Moreover, this experiment investigated if there is a relationship between the performance on the ImageNet classification task (as well as other aspects of the models) and the TL performance on the video anomaly detection task.

The experimental results for all datasets are presented in Tables 5 and 6, for AUC and EER respectively.

Table 5: Results obtained for each model and dataset in terms of AUC.

Dataset	AlexNet	GoogleNet	Inception v3	ResNet10	ResNet50	ResNet152	VGG-S	VGG-F	VGG-M	VGG-16	VGG-19
LABIC3	0.918	**0.962**	0.936	0.892	0.947	0.913	0.948	0.885	0.942	0.891	0.936
LABIC4	0.775	0.898	0.936	0.842	**0.950**	0.872	0.907	0.921	0.914	0.922	0.783
Avenue	0.813	0.829	0.762	0.819	0.732	0.821	0.803	0.836	0.820	**0.847**	0.827
Ped1	**0.719**	0.674	0.629	0.651	0.686	0.680	0.636	0.642	0.677	0.688	0.664
Ped2	0.746	0.731	0.759	0.566	0.513	0.617	0.786	0.784	**0.893**	0.662	0.612
UMN	0.911	0.960	0.983	0.946	0.983	0.981	0.981	0.981	**0.988**	0.776	0.950
Average	0.814	0.842	0.834	0.786	0.802	0.814	0.844	0.842	**0.872**	0.798	0.795

Table 6: Results obtained for each model and dataset in terms of EER.

Dataset	AlexNet	GoogleNet	Inception v3	ResNet10	ResNet50	ResNet152	VGG-S	VGG-F	VGG-M	VGG-16	VGG-19
LABIC3	0.136	0.097	0.134	0.216	0.130	0.152	**0.086**	0.213	0.115	0.181	0.143
LABIC4	0.281	0.119	0.179	0.277	**0.102**	0.174	0.148	0.168	0.197	0.211	0.295
Avenue	0.282	0.242	0.321	0.247	0.373	0.245	0.255	0.242	0.247	**0.214**	0.239
Ped1	**0.317**	0.392	0.437	0.376	0.372	0.348	0.356	0.444	0.403	0.386	0.378
Ped2	0.364	0.322	0.32	0.421	0.451	0.41	0.298	0.304	**0.161**	0.366	0.374
UMN	0.172	0.067	0.059	0.136	0.071	0.066	0.057	**0.047**	**0.047**	0.371	0.13
Average	0.259	0.207	0.242	0.279	0.250	0.233	0.200	0.236	**0.195**	0.288	0.260

Whilst many of the models present similar results for a given dataset, some of them stand out. AlexNet, for instance, outperformed all other models on the UCSD Ped1 dataset, which is probably the most challenging among all datasets used in this work. This is surprising, due to the simplicity of AlexNet when compared to some of the deeper models (see Table 2 in Section 4.1). Another case is the VGG-M network applied to the UCSD Ped2 dataset, which achieved significantly higher performance than the other models. VGG-M also performed best on the UMN dataset and achieved the highest average performance. Figure 22 shows the average AUC achieved by each network in a more comprehensible manner. In the chart, the difference in performance becomes clear, and the superiority of the VGG-M model is highlighted.

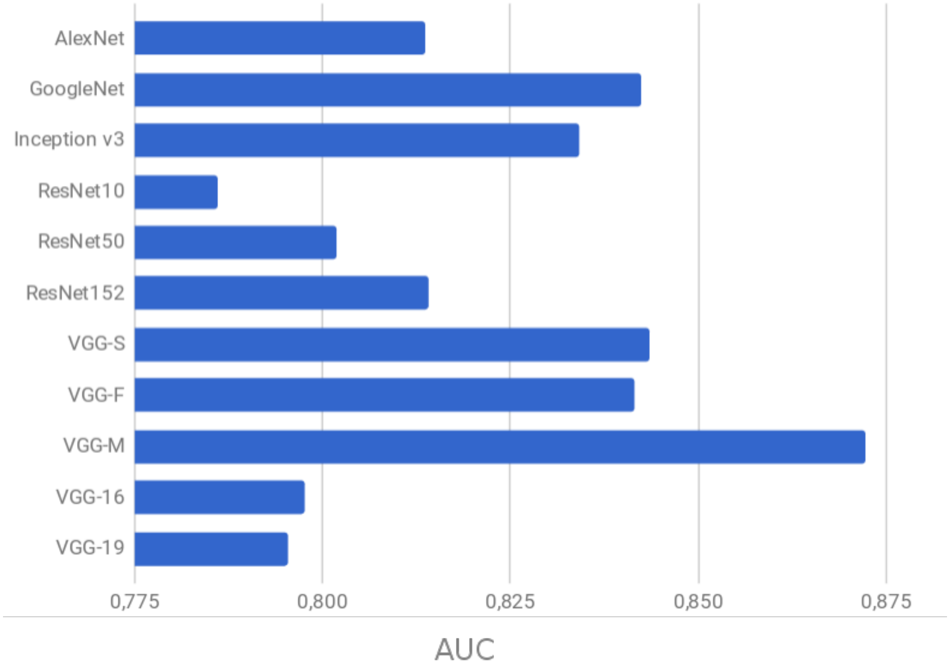


Figure 22: Average performance of each CNN model on the anomaly detection task.

However, VGG-M does not outperform the other models on all datasets. This suggests that some parameters of each model may be more suitable for specific traits of abnormality. Since all models were trained using the same data (ImageNet), it may be reasonable to assume that the variations in performance are due to architectural differences, such as number of layers, number of output features, classification performance, or even smaller details, such as number of feature maps, kernel sizes or regularization methods. This makes the problem of finding the source of the variations in performance very hard since the number of factors that influence the results is too high. Moreover, it may be a combination of these factors that lead a model to perform better than others on a specific dataset.

Despite the difficulty of the task, the relationship between the parameters of the models and the TL performance was investigated. The parameters investigated were presented in Table 2 (Section 4.1). Considering the average anomaly detection performance, scatter plots are presented, where the x-axis represents a parameter of the network, for instance, the Top-5 error rate in ImageNet classification, and the y-axis represents the AUC obtained after using the model as a feature extractor for video anomaly detection. Then, a linear regression trend line is computed for each of the plots. Figure 23 shows the scatter plot considering the average anomaly detection performance. The linear equation and the R^2 measure indicate that the linear model does not fit well to the data, which possibly means that no strong relationship exists between the two variables. The same procedure is done for other three parameters of the models: number of features output in the last pooling layer, total number of layers, and the number of convolutional layers. Figures 24, 25, and 26 display the linear regression lines for these three parameters, respectively.

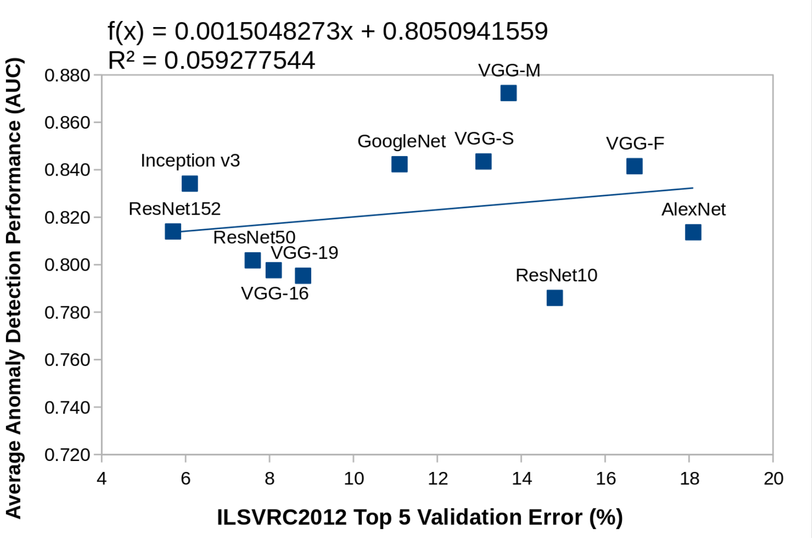


Figure 23: Relation between ILSVRC validation Top 5 error and anomaly detection average performance.

For the datasets used in this work, it is possible to conclude that these parameters of the model have no significant influence on the TL performance for anomaly detection. To answer the questions posed at the beginning of this Section, in order to choose a model for a TL application, it is necessary to evaluate models experimentally. Another alternative is to simply pick a simple model, since they require less computational power and may even achieve high performance, as is the case of VGG-M. Regarding the performance of TL in contrast with other methods, a comparison will be presented later, in Section 5.4.

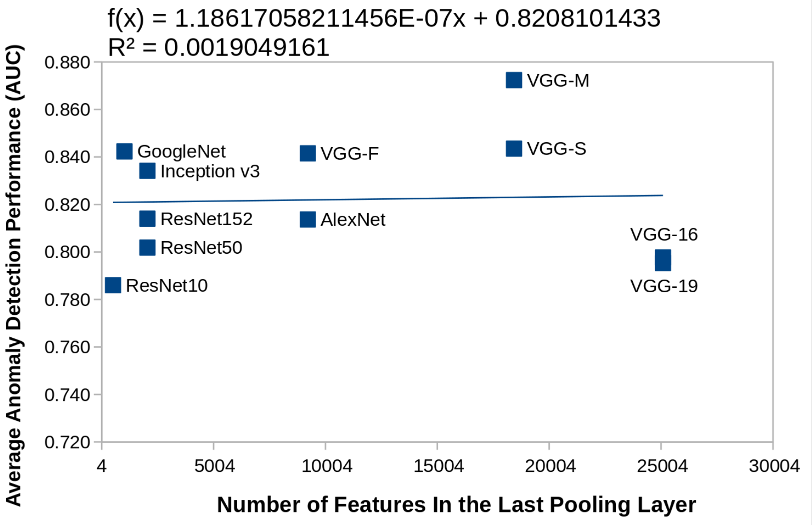


Figure 24: Relation between number of features output by a model and anomaly detection average performance.

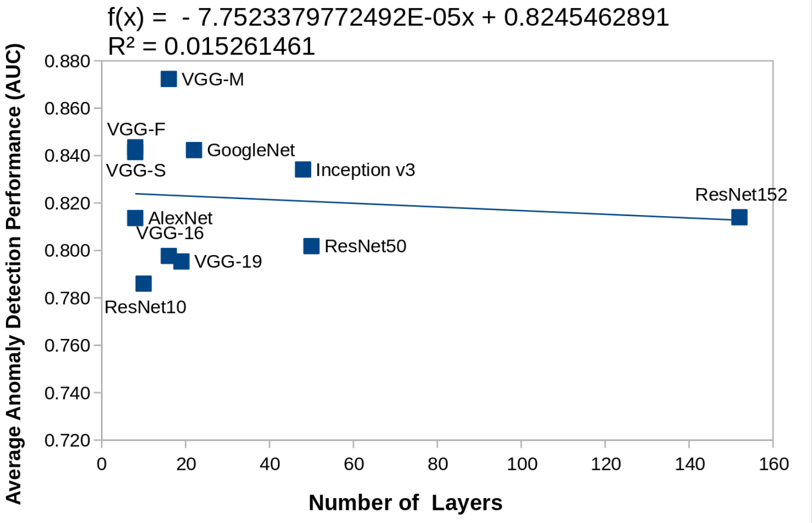


Figure 25: Relation between the number of layers in a model and anomaly detection average performance.

5.2 COMPACT REPRESENTATION APPROACH USING STACKED DENOISING AUTOENCODERS

This Section presents the results of a preliminary study regarding compact feature representations, which has been published in Gutoski et al. (2017). Section 5.2.1 describes the datasets and how the data was organized, followed by Section 5.2.2, which presents the experimental setup and the results obtained by the method.

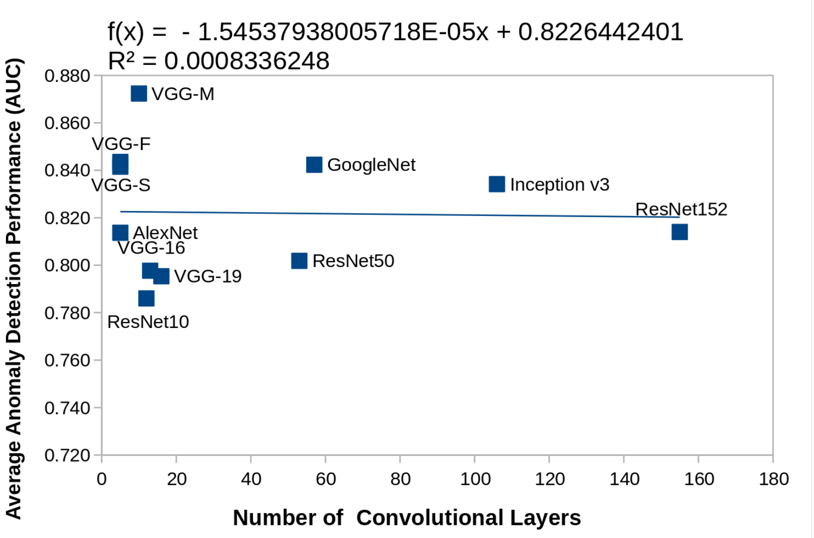


Figure 26: Relation between the number of convolutional layers in a model and anomaly detection average performance.

5.2.1 DATASETS

In this experiment, three different datasets were adapted from a multi-class classification problem to the context of OCC.

The **STL-10** [5] dataset (COATES; NG; LEE, 2011) originally contains 96x96 pixels color image data divided into 10 classes: airplane, bird, car, cat, deer, dog, horse, monkey, ship, truck. The dataset was divided as follows: unsupervised train, containing 100,000 unlabeled instances from all 10 classes and some variations; supervised train, containing 500 labeled images per class; and test, containing 800 images per class. In this work, only the supervised train and test sets were used. The data was organized as follows: classes containing animals were considered normal (6 classes), whilst the remaining classes were considered as anomalies (4 classes). The final train and test sets had 3,000 (normal) and 8,000 images (60% normal and 40% anomalies) images, respectively. Figure 27 displays samples from the dataset.

MNIST [6] is a well known handwritten digit recognition dataset introduced by LeCun et al. (1998). The handwritten digits range from 0 to 9 (10 classes). The train set contains 60,000 images, whereas the test set contains 10,000 images. Images are in grayscale and have 28×28 pixels. The MNIST dataset was organized as follows: digits 0 and 8 were considered the normal class, and the remaining images were considered anomalous. Therefore, final train set had 11,774 normal samples and test sets had 10,000 images (20% normal and 80% anomalies). Figure 28 displays samples extracted from the dataset.

[5] STL-10: found at `http://cs.stanford.edu/~acoates/stl10`

[6] MNIST: found at `http://yann.lecun.com/exdb/mnist/`

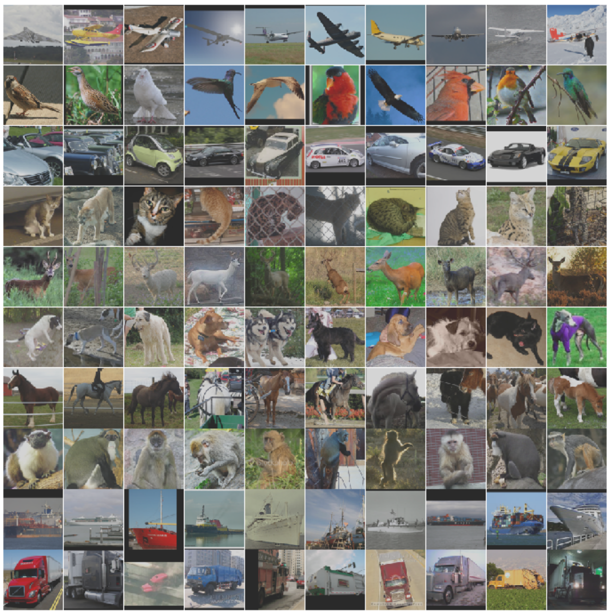


Figure 27: Samples from the STL10 dataset.

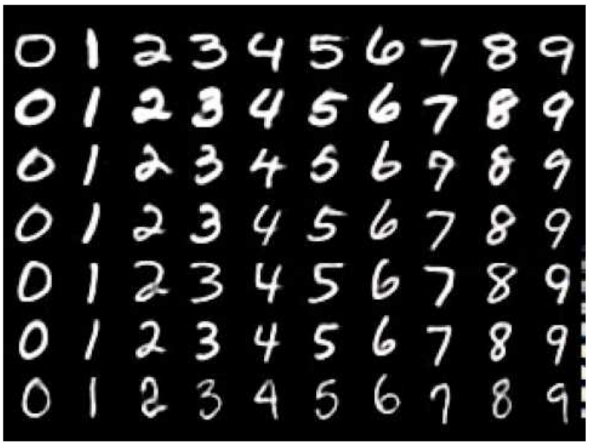


Figure 28: Samples from the MNIST dataset.

NOTMNIST [7] is a printed character dataset containing 10 classes of letters from A to J. It contains 200,000 images in the training set, 10,000 images in the validation set and 10,000 images in the test set. Images are in grayscale and have 28×28 pixels. Only the test set was used in this work. The NOTMNIST dataset was merged with the MNIST dataset. Classes that contain numbers (MNIST) were considered normal, whereas the printed characters (NOTMNIST) were considered anomalies. The final train and test sets had 60,000 images (normal) and 20,000 images (50% normal and 50% anomalies), respectively. Figure 29 shows samples of the dataset.

[7] NOTMNIST: found at `http://yaroslavvb.blogspot.com.br/2011/09/notmnist-dataset.html`

Figure 29: Samples from the NOTMNIST dataset.

5.2.2 EXPERIMENTS AND RESULTS

The experiments have been devised with the goal of investigating if the increased compactness brings a performance gain in the anomaly detection context. To do so, a baseline classification performance is drawn from performing classification using features extracted from a standard SDAE. The features are clustered using the K-Means algorithm with the same number of clusters defined for DEC. The classification is accomplished by using Equation 18, as described in Section 4.2.1.3. In this preliminary experiment, the number of clusters was set according to the number of subclasses in the normal class. Both methods were evaluated using AUC.

The general results of the preliminary experiments are presented in Table 7. The column "gain" presents the amount of improvement in terms of AUC when using compact feature representations.

Table 7: Experimental results (AUC) for the SDAE and DEC in three different experiments.

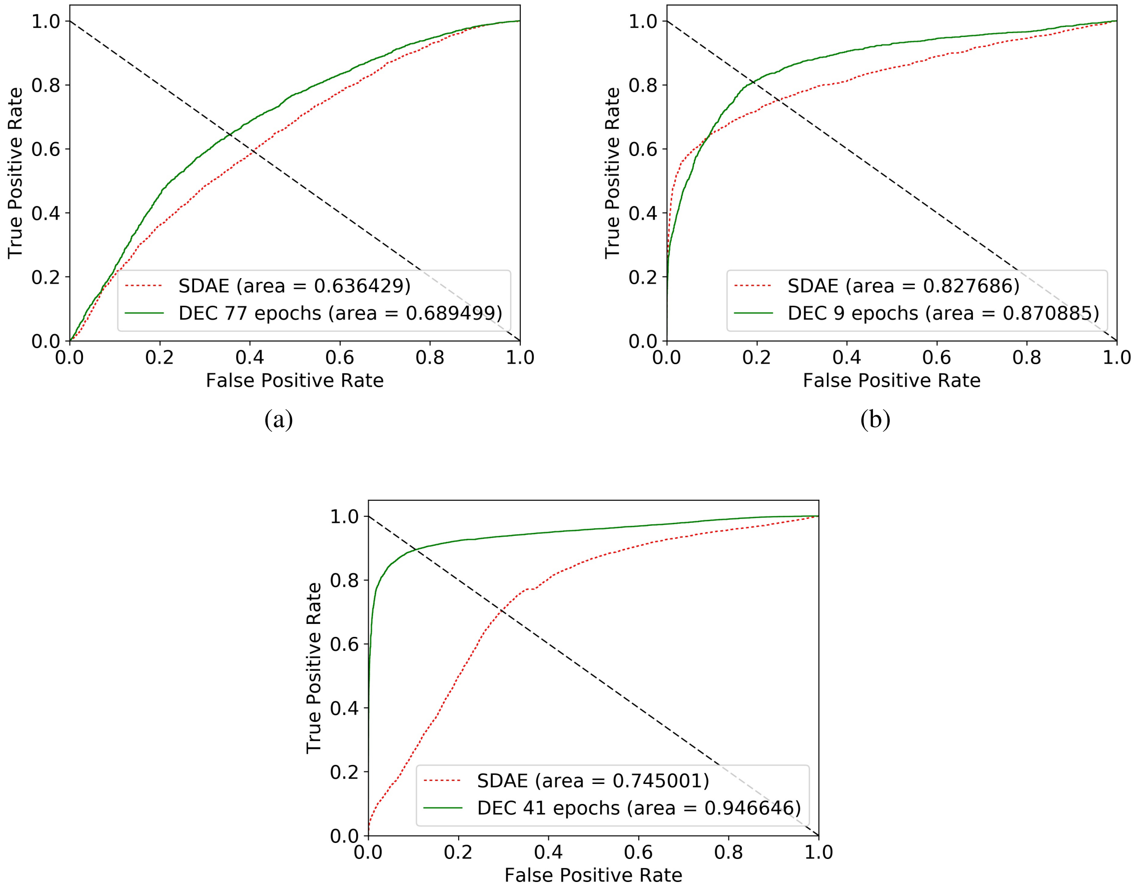
Experiment	SDAE	DEC	GAIN (%)
STL10	0.636	0.689	8.33
MNIST	0.827	0.870	5.19
NOTMNIST	0.745	0.946	26.97

In the **STL-10** experiment, the size of the bottleneck was set to 100, whilst the number of clusters (DEC parameter) was set to 6. In this experiment, DEC achieved its best performance after 77 epochs of training, outperforming the SDAE in the anomaly detection task.

In the **MNIST** experiment, the bottleneck was set to 10 neurons, and the number of clusters was set to 2. The bottleneck, in this case, has fewer neurons than the STL-10 experiment due to the smaller complexity of the MNIST dataset. Features extracted from the standard SDAE and clustered with K-Means performed worse than DEC in terms of AUC. DEC achieved its best performance after only 9 training epochs.

In the **NOTMNIST** experiment, the bottleneck contained a size of 10 neurons, and the number of clusters was set to 10. This time, DEC was trained for 41 epochs and had a very significant performance gain when compared to the SDAE.

The results show that, in all three experiments, there was an increase in performance by simply adding compactness to the feature representations. The differences can be better visualized through Figure 30. Figures 30(a), 30(b), and 30(c) shows the ROC curves for the STL10, MNIST and NOTMNIST experiments, respectively.



(a)

(b)

(c)

Figure 30: ROC curves using SDAE and DEC (best epoch) for each preliminary experiment.

Through these experiments, it can be concluded that compact feature representations can improve classification performance on OCC problems, but require further experiments on real-world scenarios. The next Section aims at investigating if this performance gain applies to proper anomaly detection benchmark datasets.

5.3 MAIN COMPACT REPRESENTATION APPROACH USING CAE-CE

This Section presents the experiments performed for detecting anomalies in six video anomaly detection benchmarks using the proposed CAE-CE. The datasets used in this experiment are identical to the datasets used in the TL experiments, presented in Section 5.1.1

This experiment aims at investigating the anomaly detection performance when compactness is introduced into the feature extraction process by the CAE-CE, instead of using a regular CAE. Both methods were evaluated according to the method presented in Section 4.2.2.

The experimental results are displayed in Table 8. The gain columns indicate the increase in performance after the insertion of compactness into the bottleneck feature representation. Results suggest that the CAE-CE outperforms the regular CAE in all experimented datasets. The results reported for the CAE-CE were obtained in the epoch of maximum sensitivity. Moreover, the tuning of other parameters was accomplished by combinatorial searches, such as the OC-SVM parameters (found individually for each method and dataset), and the number of cluster centers for the CAE-CE. These combinatorial experiments require very heavy computational endeavor and take about 40 hours to complete per dataset using GPUs and CPUs.

Table 8: AUC/EER results for all datasets using the CAE-CE.

Datasets	CAE		CAE-CE		GAIN (%)	
	AUC	**EER**	**AUC**	**EER**	**AUC**	**EER**
LABIC3	0.876	0.195	0.912	0.148	4.109	-24.102
LABIC4	0.909	0.174	0.916	0.146	0.077	-16.091
Avenue	0.816	0.269	0.828	0.247	1.470	-8.178
UCSD Ped 1	0.565	0.458	0.652	0.362	15.390	-20.960
UCSD Ped 2	0.700	0.365	0.768	0.306	9.710	-16.164
UMN	0.692	0.359	0.792	0.271	14.450	-24.512

The performance gain obtained with the CAE-CE can be better visualized in Figure 31. The left chart shows the AUC performance of the CAE (blue) and CAE-CE (red). Similarly, the right chart shows the EER obtained by each method.

By using the EER threshold, it is possible to perform anomaly detection for each frame with a balance between sensitivity and specificity, i.e. the classifier is not biased towards the normal class that has most samples. The confusion matrices for each dataset are shown in Table 9, where “N” stands for normal and “A” for anomalous. The confusion matrices allow computing the TPR and TNR for each method and dataset, highlighting the improvement in terms of classification.

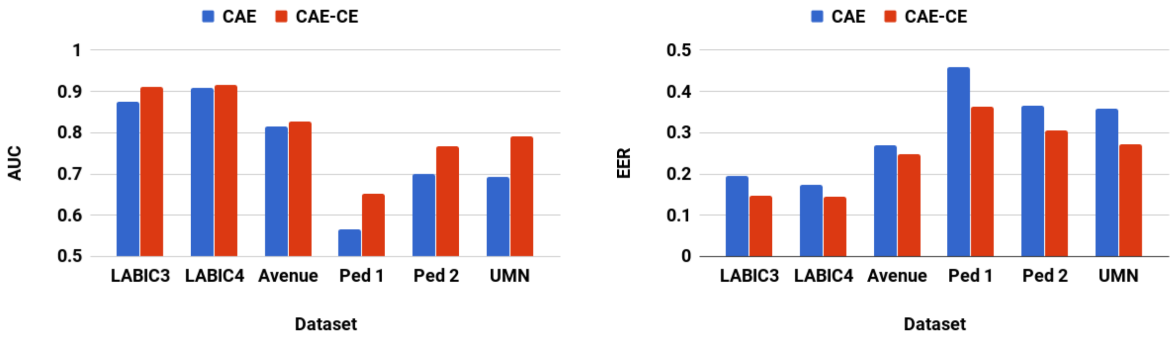


Figure 31: Performance comparison between CAE and CAE-CE on six anomaly detection benchmark datasets in terms of AUC/EER.

Table 9: Confusion matrices using the EER threshold for all datasets.

LABIC3			LABIC4			Avenue			UCSD Ped1			UCSD Ped2			UMN		
CAE Pred.			CAE Pred.			CAE Pred.			CAE Pred.			CAE Pred.			CAE Pred.		
	N	A		N	A		N	A		N	A		N	A		N	A
N	1029	74	N	431	253	N	8486	998	N	415	566	N	230	603	N	604	155
A	251	306	A	91	1211	A	3126	2714	A	350	669	A	132	1045	A	342	281
CAE-CE Pred.			CAE-CE Pred.			CAE-CE Pred.			CAE-CE Pred.			CAE-CE Pred.			CAE-CE Pred.		
	N	A		N	A		N	A		N	A		N	A		N	A
N	1090	56	N	446	215	N	8742	917	N	488	447	N	251	504	N	689	118
A	190	324	A	76	1249	A	2870	2795	A	277	788	A	111	1144	A	257	318

The TPR and TNR obtained from the confusion matrices are shown in Table 10. Notice that each method presents almost identical TPR and TNR for each dataset, which is ensured by the EER threshold. Regarding these measures, the CAE-CE also outperforms the CAE in all datasets.

Table 10: Anomaly detection results using the EER threshold for all datasets

Datasets	CAE		CAE-CE		GAIN (%)	
	TPR	**TNR**	**TPR**	**TNR**	**TPR**	**TNR**
LABIC3	0.803	0.805	0.851	0.852	5.977	5.838
LABIC4	0.825	0.827	0.854	0.853	3.515	3.143
Avenue	0.730	0.731	0.752	0.752	3.013	2.872
UCSD Ped 1	0.542	0.541	0.637	0.638	17.527	17.929
UCSD Ped 2	0.635	0.634	0.693	0.694	9.133	9.463
UMN	0.638	0.644	0.728	0.729	14.106	13.198

As before, the performance gain is better visualized through a chart. Figure 32 shows the performance of both methods in terms of TPR. The TNR chart is not presented since it is almost identical to the TPR chart. The experiments presented in this Section have shown that compact feature representation seems to be a promising way towards robust video anomaly detection systems.

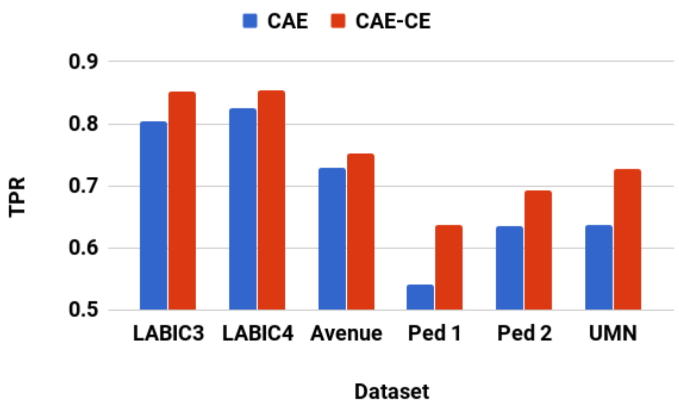


Figure 32: Performance comparison betweem CAE and CAE-CE on six anomaly detection benchmark datasets in terms of TPR.

5.4 COMPARISON AND DISCUSSION

This Section presents a performance comparison between a baseline method, TL, CAE-CE and state-of-the-art methods in the literature.

As the baseline feature extractor, the Histogram of Oriented Gradients (HOG) (DALAL; TRIGGS, 2005) was employed. The classification method using HOG follows the same steps as the TL approach. Tables 11 and 12 presents the results obtained by each method regarding the AUC and EER, respectively.

Table 11: Anomaly detection performance of all methods presented in this work in terms of AUC.

Dataset	Baseline (HOG)	TL (best)	CAE	CAE-CE	State-of-the-art	Reference
LABIC3	0.851	0.962	0.876	0.912	-	-
LABIC4	0.552	0.950	0.909	0.916	-	-
Avenue	0.798	0.847	0.816	0.828	0.843	(SMEUREANU et al., 2017b)
Ped1	0.616	0.719	0.565	0.652	0.927	(SALIGRAMA; CHEN, 2012)
Ped2	0.701	0.893	0.700	0.768	0.908	(XU et al., 2015)
UMN	0.873	0.988	0.692	0.792	0.997	(SUN; LIU; HARADA, 2017)
Average	0.747	0.893	0.759	0.811	-	-

As expected, the baseline method performed lower than all other methods in terms of average performance. However, its average performance was only slightly lower than the CAE. In fact, HOG has outperformed the CAE in the Ped1 and UMN datasets, where the performance of the CAE was exceptionally poor. Contrariwise, the CAE has outperformed HOG by a large margin In LABIC4.

In its turn, the CAE-CE presented an average performance much higher than HOG and

Table 12: Anomaly detection performance of all methods presented in this work in terms of EER.

Dataset	Baseline (HOG)	TL (best)	CAE	CAE-CE	State-of-the-art	Reference
LABIC3	0.203	0.086	0.195	0.148	-	-
LABIC4	0.557	0.102	0.174	0.146	-	-
Avenue	0.282	0.214	0.269	0.247	0.27	(RIBEIRO; LAZZARETTI; LOPES, 2017)
Ped1	0.398	0.317	0.458	0.362	0.16	(SALIGRAMA; CHEN, 2012)
Ped2	0.367	0.161	0.365	0.306	0.17	(XU et al., 2015)
UMN	0.186	0.047	0.359	0.271	-	-
Average	0.308	0.154	0.303	0.246	-	-

CAE. Since the CAE-CE is an improvement upon the CAE, it still carries some of its traits. For instance, the CAE-CE also performed poorly on the UMN dataset. Despite improving the results with respect to the CAE by introducing compactness, the improvement was not enough to surpass the performance of HOG on the UMN dataset. One possible explanation for this result is the nature of anomalies present in the UMN dataset, which is sole of movement. Therefore, the moving average filter, which was applied to the HOG anomaly scores may have caused a greater impact than the temporal features learned by the CAE. Despite this exception, the CAE-CE has achieved the second best results among the methods studied in this work. Its performance on the Avenue dataset was comparable to the state-of-the-art results.

The TL approach presented surprisingly good results, despite being a very simple method. The results obtained by extracting features using TL achieved state-of-the-art results on three of the datasets, and also greatly outperformed other methods regarding the average performance on all six datasets. Achieving results near to the state-of-the-art on three datasets is indeed impressive since those methods are usually fine-tuned for a specific dataset. Notwithstanding, TL is a more general method, and has been shown to be effective in six different scenarios. It is possible that this effectiveness can be extended to a much greater number of scenarios in the context of video anomaly detection.

Figure 33 displays the average performance obtained by each method in terms of AUC (left) and EER (right). It is possible to observe that the performance obtained by the baseline HOG and the CAE are very similar, followed by the CAE-CE and finally TL, which performed much better than all other methods. As a general conclusion, it is possible to state that TL is a very promising approach. It not only performs better than the other methods but, also, has much lower computational cost, since pretrained models are widely available for the major DL frameworks.

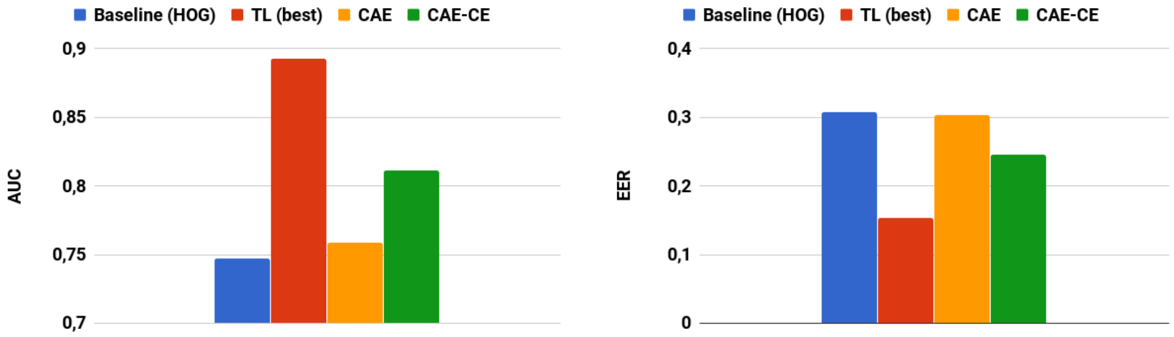


Figure 33: Average anomaly detection performance of all methods.

6 CONCLUSION

Automatic anomaly detection on surveillance videos is becoming ever more important. Along time, the amount of security cameras in public and private spaces outgrows the number of human observers available, hindering the performance of security systems. It is of extreme importance that surveillance systems achieve the highest possible degree of effectiveness, since lives may depend on them. Thus, devising methods for automatic video surveillance is a task of great importance.

In this work, two approaches for performing anomaly detection in surveillance videos were proposed. The first approach was the feature extraction process using TL. Experiments performed on six benchmark datasets have shown the robustness of the features obtained in this way. From the results, it is possible to conclude that features learned in the ImageNet dataset are transferable to completely unrelated tasks, such as video anomaly detection. Moreover, the results suggest that, perhaps, TL could become a key element towards achieving a more general CI, where a model can perform multiple tasks, including some for which it was not designed for.

The second approach for video anomaly detection was the learning of compact feature representations using AEs. First, a preliminary experiment was performed on static images, with the objective of testing the compactness hypothesis for OCC problems: does compact feature representations improve the classification performance? Indeed, experimental results have confirmed the hypothesis. Then, the method was applied to six anomaly detection video datasets, where compactness also improved the classification results. These results are interesting because it is possible that by fine-tuning for compactness, this improvement in performance may be achievable by using other types of features, not only those learned by the AE from raw images.

In comparison to other methods in the literature, both of the approaches used in this work have shown competitive results. In special, the TL approach has achieved near the state-of-the-art results on three out of four datasets in which this information is available. This reinforces the idea that TL can have a strong generalization capacity, performing well across different tasks.

In this work, the importance of having proper feature representations has become clear. The variations in results of the methods presented in this work are mostly related to the feature learning process since the OC-SVM was used as the classification algorithm in all video experiments. Hence, devising robust feature extractors is a task of great importance.

It is possible to state that the objectives presented in the introduction of this work have been achieved. Naturally, many aspects of the methods can still be improved in order to achieve more robust anomaly detection models. Nevertheless, the expectations for this work is that it will contribute to the current video anomaly detection research and allow future research to be built upon it.

6.1 CONTRIBUTIONS

The contributions of this work were:

- a method for transferring a feature extractor from a supervised multi-class classification task to the video anomaly detection task;
- a moving average filter for removing noise from the video anomaly scores;
- the use of DEC for video anomaly detection tasks;
- extension of DEC to a convolutional architecture.

6.2 FUTURE WORKS

Future works mainly involve improving the feature extraction process. This work has shown that compactness can successfully improve the quality of the features obtained by a feature extractor, such as a SDAE or a CAE. Moreover, the TL approach has shown that features obtained this way are very powerful for discriminating anomalies. Hence, a combination of both methods may produce interesting results. Adding compactness to the TL features can be achieved through DEC or other methods, such as the Null-Space.

Another important aspect to explore is the automatic tuning of parameters. The experiments performed in this work have been done with a factorial experiment of parameters, including the OC-SVM, DEC number of clusters and moving average filter parameters. This search was done to achieve the best possible performance for each method and thus allowing for a fair comparison. However, this process is very expensive in terms of computational endeavor.

The temporal aspects of the video are also worth exploring in future works. Better results may be achieved by exploring methods for learning temporal dependencies. Another factor worth exploring in future works is anomaly localization and tracking within the frame, which has not been explored in this work.

REFERENCES

ABADI, M. et al. TensorFlow: large-scale machine learning on heterogeneous systems. **arXiv:1603.04467**, p. 1–19, 2016.

AMRAEE, S. et al. Anomaly detection and localization in crowded scenes using connected component analysis. **Multimedia Tools and Applications**, Springer, 2017. In Press.

BAR-HILLEL, A.; WEINSHALL, D. Learning a Mahalanobis metric from equivalence constraints. **Journal of Machine Learning Research**, v. 6, n. 1, p. 937–965, 2005.

BODESHEIM, P. et al. Kernel null space methods for novelty detection. In: **Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Piscataway, NJ: IEEE press, 2013. p. 3374–3381.

BODESHEIM, P. et al. Local novelty detection in multi-class recognition problems. In: **Proc. of the IEEE Winter Conference on Applications of Computer Vision (WACV)**. Piscataway, NJ: IEEE press, 2015. p. 813–820.

CHATFIELD, K. et al. Return of the devil in the details: Delving deep into convolutional nets. **arXiv preprint arXiv:1405.3531**, 2014.

CHENG, K.-W.; CHEN, Y.-T.; FANG, W.-H. Video anomaly detection and localization using hierarchical feature representation and Gaussian process regression. In: **Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Piscataway, NJ: IEEE press, 2015. p. 2909–2917.

CHONG, Y. S.; TAY, Y. H. Abnormal event detection in videos using spatiotemporal autoencoder. In: **Proc. of the International Symposium on Neural Networks**. Piscataway, NJ: IEEE press, 2017. p. 189–196.

COATES, A.; NG, A.; LEE, H. An analysis of single-layer networks in unsupervised feature learning. In: **Proc. of the Fourteenth International Conference on Artificial Intelligence and Statistics**. Piscataway, NJ: IEEE press, 2011. p. 215–223.

COLQUE, R. V. H. M. et al. Histograms of optical flow orientation and magnitude and entropy to detect anomalous events in videos. **IEEE Transactions on Circuits and Systems for Video Technology**, v. 27, n. 3, p. 673–682, 2017.

CORTES, C.; VAPNIK, V. Support-vector networks. **Machine Learning**, v. 20, n. 3, p. 273–297, 1995.

DALAL, N.; TRIGGS, B. Histograms of oriented gradients for human detection. In: **Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Piscataway, NJ: IEEE press, 2005. v. 1, p. 886–893.

DIELEMAN, S. et al. **Lasagne: First release.** ago. 2015. Disponível em: <http://dx.doi.org/10.5281/zenodo.27878>.

ERFANI, S. M. et al. High-dimensional and large-scale anomaly detection using a linear one-class SVM with deep learning. **Pattern Recognition**, v. 58, n. 1, p. 121–134, 2016.

FUNG, G. P. C. et al. Text classification without negative examples revisit. **IEEE transactions on Knowledge and Data Engineering**, v. 18, n. 1, p. 6–20, 2006.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. Cambridge: MIT press, 2016.

GUTOSKI, M. et al. A clustering-based deep autoencoder for one-class image classification. In: **Proc. of the IEEE Latin American Conference on Computational Intelligence**. Piscataway, NJ: IEEE press, 2017.

HASAN, M. et al. Learning temporal regularity in video sequences. In: **Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Piscataway, NJ: IEEE press, 2016. p. 733–742.

HE, K. et al. Deep residual learning for image recognition. **arXiv preprint arXiv:1512.03385**, 2015.

HINAMI, R.; MEI, T.; SATOH, S. Joint detection and recounting of abnormal events by learning deep generic knowledge. **arXiv preprint arXiv:1709.09121**, 2017.

HINTON, G. E. et al. Improving neural networks by preventing co-adaptation of feature detectors. **arXiv:1207.0580**, p. 1–18, 2012.

HORNIK, K.; STINCHCOMBE, M.; WHITE, H. Multilayer feedforward networks are universal approximators. **Neural Networks**, v. 2, n. 5, p. 359–366, 1989.

IONESCU, R. T. et al. Detecting abnormal events in video using narrowed motion clusters. **arXiv preprint arXiv:1801.05030**, 2018.

JARRETT, K. et al. What is the best multi-stage architecture for object recognition? In: **Proc. of the IEEE 12th International Conference on Computer Vision**. Piscataway, NJ: IEEE press, 2009. p. 2146–2153.

JIA, Y. et al. Caffe: Convolutional architecture for fast feature embedding. In: **Proc. of the 22nd ACM International Conference on Multimedia**. New York City, NY: ACM, 2014. p. 675–678.

KHAN, S. S.; MADDEN, M. G. A survey of recent trends in one class classification. In: **Proc. of the Irish Conference on Artificial Intelligence and Cognitive Science**. Berlin, Heidelberg: Springer-Verlag, 2009. p. 188–197.

KIRAN, B. R.; THOMAS, D. M.; PARAKKAL, R. An overview of deep learning based methods for unsupervised and semi-supervised anomaly detection in videos. **arXiv preprint arXiv:1801.03149**, 2018.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. Imagenet classification with deep convolutional neural networks. In: **Proc. of the 25th International Conference on Neural Information Processing Systems**. USA: Curran Associates Inc., 2012. v. 1, p. 1097–1105.

KUMAR, B. S.; RAVI, V. Text document classification with PCA and one-class SVM. In: **Proc. of the 5th International Conference on Frontiers in Intelligent Computing: Theory and Applications**. Singapore: Springer, 2017. p. 107–115.

LECUN, Y. et al. Gradient-based learning applied to document recognition. **Proc. of the IEEE**, v. 86, n. 11, p. 2278–2324, 1998.

LI, X. et al. Deep learning for RFID-based activity recognition. In: **Proc. of the 14th ACM Conference on Embedded Network Sensor Systems**. New York, NY: ACM, 2016. p. 164–175.

LIANG, H. et al. Text feature extraction based on deep learning: a review. **EURASIP Journal on Wireless Communications and Networking**, v. 2017, n. 1, p. 211, 2017.

LU, C.; SHI, J.; JIA, J. Abnormal event detection at 150 fps in MATLAB. In: **Proc. of the IEEE International Conference on Computer Vision**. Piscataway, NJ: IEEE Press, 2013. p. 2720–2727.

LU, J. et al. Transfer learning using computational intelligence: A survey. **Knowledge-Based Systems**, v. 80, n. 1, p. 14 – 23, 2015.

LUO, W.; LIU, W.; GAO, S. A revisit of sparse coding based anomaly detection in stacked RNN framework. In: **Proc. of the IEEE International Conference on Computer Vision (ICCV)**. Piscataway, NJ: IEEE Press, 2017. p. 341–349.

MAHADEVAN, V. et al. Anomaly detection in crowded scenes. In: **Proc. of the IEEE Conference on Computer Vision and Pattern Recognition**. Piscataway, NJ: IEEE Press, 2010. p. 1975–1981.

MANEVITZ, L. M.; YOUSEF, M. One-class SVMs for document classification. **Journal of Machine Learning Research**, v. 2, n. 1, p. 139–154, 2001.

MASCI, J. et al. Stacked convolutional auto-encoders for hierarchical feature extraction. In: **Proc. of the 21st International Conference on Artificial Neural Networks**. Berlin, Heidelberg: Springer-Verlag, 2011. p. 52–59.

MEHRAN, R.; OYAMA, A.; SHAH, M. Abnormal crowd behavior detection using social force model. In: **Proc. of the IEEE Conference on Computer Vision and Pattern Recognition**. Piscataway, NJ: IEEE Press, 2009. v. 2, p. 935–942.

NARASIMHAN, M. G.; KAMATH, S. Dynamic video anomaly detection and localization using sparse denoising autoencoders. **Multimedia Tools and Applications**, 2017. In Press.

NG, A. Y. Feature selection, L1 vs L2 regularization, and rotational invariance. In: **Proc. of the 21st International Conference on Machine Learning**. New York, NY: ACM, 2004. p. 78–86.

OJALA, T.; PIETIKAINEN, M.; HARWOOD, D. Performance evaluation of texture measures with classification based on Kullback discrimination of distributions. In: **Proc. of the 12th IEEE International Conference on Pattern Recognition**. Piscataway, NJ: IEEE press, 1994. v. 1, p. 582–585.

PAN, S. J.; YANG, Q. A survey on transfer learning. **IEEE Transactions on Knowledge and Data Engineering**, v. 22, n. 10, p. 1345–1359, 2010.

PERLIN, H. **A Contribution to Semantic Description of Images and Videos: an Application of Soft Biometrics**. Thesis (Doctorate) — CPGEI - Federal University of Technology - Parana, 2015.

PIMENTEL, M. A. et al. A review of novelty detection. **Signal Processing**, v. 99, n. 1, p. 215 – 249, 2014.

PONTI, M.; NAZARE, T. S.; KITTLER, J. Optical-flow features empirical mode decomposition for motion anomaly detection. In: **Proc. of the IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)**. Piscataway, NJ: IEEE press, 2017. p. 1403–1407.

RIBEIRO, M.; LAZZARETTI, A. E.; LOPES, H. S. A study of deep convolutional auto-encoders for anomaly detection in videos. **Pattern Recognition Letters**, 2017. In Press.

ROODPOSHTI, M. S.; SAFARRAD, T.; SHAHABI, H. Drought sensitivity mapping using two one-class support vector machine algorithms. **Atmospheric Research**, v. 193, n. 1, p. 73 – 82, 2017.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. **Nature**, v. 323, n. 6088, p. 533–536, 1986.

RUSSAKOVSKY, O. et al. Imagenet large scale visual recognition challenge. **International Journal of Computer Vision (IJCV)**, v. 115, n. 3, p. 211–252, 2015.

SABOKROU, M. et al. Real-time anomaly detection and localization in crowded scenes. In: **Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops**. Piscataway, NJ: IEEE press, 2015. p. 56–62.

SABOKROU, M. et al. Deep-cascade: Cascading 3D deep neural networks for fast anomaly detection and localization in crowded scenes. **IEEE Transactions on Image Processing**, v. 26, n. 4, p. 1992–2004, 2017.

SALIGRAMA, V.; CHEN, Z. Video anomaly detection based on local statistical aggregates. In: **Proc. of the IEEE Conference on Computer Vision and Pattern Recognition**. Piscataway, NJ: IEEE press, 2012. p. 2112–2119.

SCHÖLKOPF, B. et al. Estimating the support of a high-dimensional distribution. **Neural Computation**, v. 13, n. 7, p. 1443–1471, 2001.

SHAO, L.; ZHU, F.; LI, X. Transfer learning for visual categorization: a survey. **IEEE Transactions on Neural Networks and Learning Systems**, v. 26, n. 5, p. 1019–1034, 2015.

SHENTAL, N. et al. Adjustment learning and relevant component analysis. In: **Proc. of the 7th European Conference on Computer Vision**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2002. p. 776–792.

SIMON, M.; RODNER, E.; DENZLER, J. Imagenet pre-trained models with batch normalization. **arXiv preprint arXiv:1612.01452**, 2016.

SMEUREANU, S. et al. Deep appearance features for abnormal behavior detection in video. In: **Proc. of the International Conference on Image Analysis and Processing**. Cham: Springer International Publishing, 2017. p. 779–789.

SMEUREANU, S. et al. Deep appearance features for abnormal behavior detection in video. In: **Proc. of the International Conference on Image Analysis and Processing**. Cham: Springer International Publishing, 2017. p. 779–789.

SODEMANN, A. A.; ROSS, M. P.; BORGHETTI, B. J. A review of anomaly detection in automated surveillance. **IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews**, v. 42, n. 6, p. 1257–1272, 2012.

SONG, C. et al. Auto-encoder based data clustering. In: **Proc. of the Iberoamerican Congress on Pattern Recognition**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. p. 117–124.

SUN, Q.; LIU, H.; HARADA, T. Online growing neural gas for anomaly detection in changing surveillance scenes. **Pattern Recognition**, v. 64, n. Suppl. c, p. 187 – 201, 2017.

SZEGEDY, C. et al. Going deeper with convolutions. In: **Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Piscataway, NJ: IEEE press, 2015. p. 1–9.

SZEGEDY, C. et al. Rethinking the inception architecture for computer vision. In: **Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Piscataway, NJ: IEEE press, 2016. p. 2818–2826.

TAX, D. M. J. **One-class classification**. Thesis (PhD Thesis) — Advanced School for Computing and Imaging, Technische Universiteit Delft, 2001.

Theano Development Team. Theano: A Python framework for fast computation of mathematical expressions. **arXiv e-prints**, maio 2016.

TRAN, H. T.; HOGG, D. Anomaly detection using a convolutional winner-take-all autoencoder. In: **Proc. of the British Machine Vision Conference 2017**. [S.l.: s.n.], 2017. In Press.

van der Maaten, L. Learning a parametric embedding by preserving local structure. In: **Proceedings of the 12th International Conference on Artificial Intelligence and Statistics**. [S.l.]: PMLR, 2009. v. 5, p. 384–391.

VAPNIK, V. N. **Statistical Learning Theory**. New York: Wiley-Interscience, 1998.

WANG, S. et al. Video anomaly detection and localization by local motion based joint video representation and OCELM. **Neurocomputing**, v. 277, p. 161 – 175, 2018.

WANG, T.; SNOUSSI, H. Detection of abnormal visual events via global optical flow orientation histogram. **IEEE Transactions on Information Forensics and Security**, v. 9, n. 6, p. 988–998, 2014.

WEINBERGER, K. Q.; SAUL, L. K. Distance metric learning for large margin nearest neighbor classification. **Journal of Machine Learning Research**, v. 10, n. 1, p. 207–244, 2009.

XIE, J.; GIRSHICK, R.; FARHADI, A. Unsupervised deep embedding for clustering analysis. In: **Proc. of the 33rd International Conference on Machine Learning**. [S.l.]: JMLR, 2016. p. 478–487.

XU, D. et al. Learning deep representations of appearance and motion for anomalous event detection. **arXiv preprint arXiv:1510.01553**, 2015.

XU, D. et al. Video anomaly detection based on a hierarchical activity discovery within spatio-temporal contexts. **Neurocomputing**, v. 143, n. 1, p. 144 – 152, 2014.

XU, J. et al. Unusual event detection in crowded scenes using bag of LBPs in spatio-temporal patches. In: **Proc. of the International Conference on Digital Image Computing: Techniques and Applications**. Piscataway, NJ: IEEE press, 2011. p. 549–554.

XU, J. et al. Dynamic texture reconstruction from sparse codes for unusual event detection in crowded scenes. In: **Proceedings of the 2011 Joint ACM Workshop on Modeling and Representing Events**. New York, NY, USA: ACM, 2011. p. 25–30.

YAN, K.; JI, Z.; SHEN, W. Online fault detection methods for chillers combining extended kalman filter and recursive one-class SVM. **Neurocomputing**, v. 228, n. 1, p. 205 – 212, 2017.

YOSINSKI, J. et al. How transferable are features in deep neural networks? In: **Proc. of the 27th International Conference on Neural Information Processing Systems**. Cambridge, MA, USA: MIT Press, 2014. v. 2, p. 3320–3328.

YU, H. Single-class classification with mapping convergence. **Machine Learning**, v. 61, n. 1, p. 49–69, 2005.

ZHANG, L.; XIANG, T.; GONG, S. Learning a discriminative null space for person re-identification. In: **Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. Piscataway, NJ: IEEE press, 2016. p. 1239–1248.