Detection of Fake and Recontextualized Media against Disinformation

Omran Alamayreh

PhD in Information Engineering and Science

Omran Khaled Abdallah Alamayreh
Signed by Omran Khaled Abdallah Alamayreh

UNIVERSITÀ DEGLI STUDI DI SIENA

Dipartimento di Ingegneria dell'Informazione e Scienze Matematiche

Detection of Fake and Recontextualized Media against Disinformation

Omran Alamayreh

PhD in Information Engineering and Science

Supervisor
Prof. Mauro Barni

Siena, 22/04/2024

Contents

1 Introduction **1**
- 1.1 Motivation and Contribution 2
- 1.2 Thesis Overview 5
- 1.3 Publications 5
- 1.4 Activity Within Research Projects 6

I DeepFake Detection **7**

2 DeepFakes and Multimedia Forensics **9**
- 2.1 Multimedia Forensics 9
  - 2.1.1 Generation of Deepfake Videos 12
  - 2.1.2 Deepfake Detection Methods 15

3 Detection of GAN-synthesized Street Videos **19**
- 3.1 Motivation 19
  - 3.1.1 Video-to-Video synthesis 20
- 3.2 Methodology 22
  - 3.2.1 DeepStreets Dataset 22
  - 3.2.2 The Developed Detector 23
- 3.3 Experimental Results 25
- 3.4 Conclusions 27

4 Detection of Human Body Reenactment Fake Videos **29**
- 4.1 Motivation 29
- 4.2 Human Body Reenactment Video Dataset 30
  - 4.2.1 Video synthesis pipeline 31
  - 4.2.2 FakeDance dataset 32
  - 4.2.3 Fake video quality assessment 34
  - 4.2.4 Limitations of the FakeDance dataset 37
- 4.3 Detection of fake dance videos 38

4.3.1 Features-based detector . . . 38
4.3.2 CNN-based detector . . . 40
4.4 Experimental Analysis . . . 41
4.4.1 Experimental setup . . . 41
4.4.2 Results . . . 42
4.4.3 Fusion of LDP-TOP and CNN detectors . . . 47
4.4.4 Comparisons with State-of-the-Art . . . 48
4.5 Conclusions . . . 49

5 Detecting Deepfake Videos in Data Scarcity Conditions 51
5.1 Motivation . . . 51
5.2 DeepFake Detection Based on Video Coding Features . . . 52
5.2.1 Video coding (H.264) in a nutshell . . . 53
5.2.2 Video coding features . . . 53
5.3 Experimental methodology . . . 56
5.3.1 Datasets . . . 56
5.3.2 SVM detection, training and setting . . . 56
5.4 Experimental Results . . . 59
5.4.1 Comparison results . . . 60
5.5 Conclusions . . . 63

II Image Geolocalization 65

6 Photo Geolocalization 67
6.1 Introduction . . . 67
6.2 Motivation and Contributions . . . 68
6.3 Related Work . . . 69
6.3.1 Global scale Image Geolocalization . . . 69
6.3.2 Visual Place Recognition . . . 71
6.3.3 Image Geolocalization datasets . . . 72

7 Country Recognition 75
7.1 Introduction . . . 75
7.2 The VIPPGeo Dataset . . . 76
7.2.1 Pre-filtering . . . 78
7.2.2 Dataset composition . . . 81
7.3 DNN-based classifier . . . 84
7.4 Experiments and Results . . . 86
7.4.1 Ablation study . . . 87
7.4.2 Plausibility analysis . . . 89
7.5 Conclusions . . . 89

8 City Verification 91
8.1 Introduction . . . 91
8.2 ViT and Siamese Neural Networks . . . 92
8.2.1 Vision Transformers . . . 92
8.2.2 Siamese Networks . . . 93
8.3 Methodology . . . 93
8.3.1 City Verification Dataset . . . 93
8.3.2 City Verification System . . . 94
8.4 Experiments and Results . . . 98
8.4.1 Siamese Network Training . . . 98
8.4.2 Verification Results - Closed Set . . . 99
8.4.3 Verification Results - Open Set . . . 100
8.4.4 Ablation Study . . . 101
8.4.5 Comparisons with the State-of-the-Art . . . 102
8.5 Verification Examples . . . 103
8.5.1 Closed set . . . 105
8.5.2 Open Set . . . 105
8.6 Conclusion . . . 112

9 Conclusion 113
9.1 Summary and Final Remarks . . . 113
9.2 Open Issues . . . 115

Bibliography 117

Index 129

List of Figures

2.1 The historical evolution of image manipulation. 10
2.2 A typical GAN designed to produce synthetic human facial images. 12
2.3 Autoencoder-based face swapping. 13
2.4 Progressive transformation of noise into real data and vice versa within a Diffusion model. 14

3.1 Vid2vid network architecture. 21
3.2 Example of frames generated by vid2vid architecture. The first row displays frames from Cityvid, the second row exhibits Citywcvid frames, and the third row presents frames from the Kittivid sub-datasets. 22
3.3 Xception network architecture [1]. 24
3.4 Detection pipeline. 24

4.1 Video synthesis pipeline. In the training phase, a motion transfer model learns to transfer the motion to the target subject from pose stick figures and video frames. In the transfer phase, the motion is transferred from the source to the target subject by feeding the model with normalized pose stick figures of the source person. 31
4.2 Video frames selected from the FakeDance dataset. In green and red, the frames are related to the "real" class and the "fake" class, respectively. . . 35
4.3 Example of frames utilized in the video quality assessment experiments. The first row displays frames from *Fit_dance*, the second row exhibits *Puppet_dance* frames, and the third row presents frames from the *Splice_dance* sub-datasets. 36
4.4 Examples of frames showing the limitations of the FakeDance dataset. . . 37
4.5 Feature-based classification pipeline. The video sequence under analysis is pre-processed in two ways extracting sequences corresponding to five different body parts. Then, we extract the LDP-TOP features from each sequence, later, we pass them through an MLP classifier and finally, we obtain the prediction through fusion and majority voting. 39
4.6 Detection pipeline of the CNN-based detector. 41

5.1 Frequency of *Intra*, *Inter* and *Skip* for raw videos in the DeepStreets dataset. 54
5.2 Feature distribution for 1,000 videos in the DeeperForensics-1.0 dataset and 394 videos in the VFHQ dataset. 55
5.3 Examples of real and fake videos from the three datasets. From top to bottom row: 3 real and 3 fake from DeepStreets, 3 real and 3 fake from DeepFor, and 3 real and 3 fake from VFHQ. 57

7.1 Examples of images with Creative Commons licenses retrieved by the Flickr API for the query *'London church'*. 77
7.2 Three sample photos taken from Flickr, Mapillary and Unsplash. 78
7.3 Examples of filtered images from the VIPPGeo dataset. 79
7.4 The geographical distribution of images in our dataset. 80
7.5 Histogram showing the distribution of the number of images for each of the 61 classes. The x axis shows the 61 classes and the y axis shows the number of images in each class. 81
7.6 Histogram showing the distribution of the number of images per country. The vertical axis shows the 243 countries of the world from which we have crawled the images, while the horizontal axis represents the number of images for each country. 82
7.7 Histogram with the distribution of the number of countries for each class. The x axis shows the 61 classes and the y axis reports the number of countries for each class. 83
7.8 World coloured map of the 61 country classes we have used in our experiments. 83
7.9 The overall architectural workflow. On the top of the image we show the overall workflow, with the input image, the central Resnet-101 architecture. At the bottom of the figure, we display the two types of blocks of Resnet-101. 84
7.10 The input image is first resized, then split into 5 crops. Each crop is analyzed independently. The final result is obtained by fusing the results obtained on each crop. 85
7.11 Some examples of predictions made by our system. 87
7.12 An example showing the result provided by the network when two images representing different countries are put together to form a composite image. 88
7.13 Insertion of an iconic monument of one country into a picture representing a different country. The network identifies the two contributing countries in the first positions. 89

8.1 The figure illustrates the overall workflow of the developed architecture. For more verification examples refer to Section 8.5. The code implementing the City Verification systems is publicly available at the following link: `https://github.com/alamayreh/city_verifier`. 95

8.2 Confusion matrix showing the performance of the Siamese network in determining positive and negative pairs on $64,000$ image pairs sampled from the Siamese-testing set. 99
8.3 Verification accuracy using different Top-N predictions by GeoVIPP country classifier 102
8.4 True Positive rate comparison with SOTA [2] for various values of proximity circle diameters (from left to right, 25 Km, 50 Km, and Flexible). 103
8.5 The image on the left is from Tokyo which has been claimed to be from Berlin. The 5 images on the right are from Berlin. 104
8.6 Similarity: 0.78, Sigmoid output: 0.84 105
8.7 Similarity: 0.99, Sigmoid output: 0.60 105
8.8 Similarity: 0.99, Sigmoid output: 0.09 106
8.9 Similarity: 0.72, Sigmoid output: 0.43 106
8.10 Similarity: 0.84, Sigmoid output: 0.38 106
8.11 Similarity: 0.76, Sigmoid output: 0.34 107
8.12 Similarity: 0.95, Sigmoid output: 0.60 107
8.13 Similarity: 0.82, Sigmoid output: 0.71 107
8.14 Similarity: 0.99, Sigmoid output: 0.33 108
8.15 Similarity: 0.99, Sigmoid output: 0.65 108
8.16 Similarity: 0.97, Sigmoid output: 0.94 108
8.17 Similarity: 0.98, Sigmoid output: 0.99 109
8.18 Similarity: 0.96, Sigmoid output: 0.93 109
8.19 Similarity: 0.68, Sigmoid output: 0.45 109
8.20 Similarity: 0.10, Sigmoid output: 0.43 110
8.21 Similarity: 0.99, Sigmoid output: 0.43 110
8.22 Similarity: 0.53, Sigmoid output: 0.41 110
8.23 Similarity: 0.97, Sigmoid output: 0.79 111
8.24 Similarity: 0.74, Sigmoid output: 0.30 111
8.25 Similarity: 0.88, Sigmoid output: 0.44 111

List of Tables

2.1 List of most popular deepfake datasets. 15

3.1 A summary of DeepStreets dataset. 23
3.2 Detection accuracy (the detector is asked to distinguish the original sequences used to create the segmentation masks and the synthetic videos). 25
3.3 Compression mismatch experiments. The detector has been trained (and tested) on the entire DeepStreet dataset. 25
3.4 Cross-dataset analysis between subsets of DeepStreets dataset (RAW videos). 26

4.1 Summary of the four sub-datasets composing the FakeDance dataset. All the videos of the dataset have a common resolution of $1024 \times 512 \times 3$ pixels. In all the scenarios, every real video and its corresponding fake version have the same temporal length. 34
4.2 Average similarity scores of different FakeDance sub-datasets. In the case of *Fit_dance* and *Puppet_dance*, we have computed the full-frame SSIM, while for *Splice_dance*, we have computed the Mask-SSIM, considering only the subject area. 36
4.3 Number of videos in each sub-dataset. Every set contains an equal number of real and fake videos. However, video lengths vary, with a frame rate of 25. The average lengths of videos in each sub-dataset are reported in Table 4.2.2. 41
4.4 Performance of the detectors on the FakeDance dataset in cross-dataset scenarios (training and testing are carried out on RAW videos). 43
4.5 Performance of the detectors on FakeDance dataset in cross-dataset scenarios (training and testing are carried out on HQ videos). 43
4.6 Performance of the detectors on FakeDance dataset in cross-dataset scenarios (training and testing are carried out on LQ videos). 43
4.7 Detection performance in cross-dataset scenarios and mismatched compression conditions. Detectors are trained on *Fit_dance*. 45

4.8 Detection performance in cross-dataset scenarios and mismatched compression conditions. Detectors are trained on *Fit_dance*. We have set the threshold on *Fit_dance* validation set considering a compression quality matched with testing. 46

4.9 Relative accuracy change between the results shown in Table 4.8 and those shown in Table 4.7. "-" refers to no changes. 46

4.10 Performance of the detectors on *Splice_dance*, when training is performed on *Fit_dance* videos, RAW quality. 46

4.11 Performance of the detectors in data scarcity conditions (training on RAW *Fit_dance* videos and testing on RAW videos of all four sub-datasets). . . 47

4.12 Performance of the detectors after fusion on RAW videos. LDP-TOP and CNN refer to the relative accuracy change for both detectors after fusion, defined as Δ-ACC. The symbol "-" indicates no changes. 47

4.13 Performance of the detectors after fusion on HQ videos. LDP-TOP and CNN refer to the relative accuracy change for both detectors after fusion, defined as Δ-ACC. The symbol "-" indicates no changes. 48

4.14 Performance of the detectors after fusion on LQ videos. LDP-TOP and CNN refer to the relative accuracy change for both detectors after fusion, defined as Δ-ACC. The symbol "-" indicates no changes. 48

5.1 Detection accuracy for the three different subsets in the DeepStreets for both our method and the state-of-the-art XceptionNet method in [3]. Results are reported for different video qualities (Raw, HQ and LQ). 61

5.2 Cross dataset results for DeepStreets on raw videos (120 videos are used for training). 62

5.3 Detection accuracy for DeepFor and VFHQ datasets for our method and the state-of-the-art methods. 62

5.4 Comparison in terms of training time, number of parameters and model size on DeepFor and VFHQ datasets for the case of 16 training videos. . . 63

6.1 Available datasets for VPR and Planet-scale Image Geolocalization. . . . 73

7.1 Accuracy of our method and the method in [2] for country classification. 86

7.2 Accuracy of the method proposed in [2] trained on VIPPGeo dataset. . . 87

7.3 Accuracy when different strategies are used to feed the Resnet-101 classifier with images of fixed size (224 × 224). 88

8.1 Number of images in the City Verification Dataset. 94

8.2 Results on the closed Set. In each cell, the value on the right shows the total number of images that have successfully passed the GeoVIPP classifier test. The value on the left indicates the fraction of images that passed the overall verification procedure. The overall accuracy is 0.82 for positive pairs (True Positive rate) and 0.96 for the negative pairs (off-diagonal elements, True Negative rate). The blue cells refer to images from the same cities, while the grey cells indicate the results obtained on different cities of the same country. 99
8.3 Verification accuracy results in the open set scenario. In each cell, the value on the right shows the total number of images that have successfully passed the GeoVIPP classifier test. The value on the left indicates the fraction of images that passed the overall verification procedure. The overall accuracy is 0.71 for positive pairs (True Positive rate) and 0.98 for the negative pairs (off-diagonal elements, True Negative rate). The blue cells refer to images from the same cities, while the grey cells indicate the results obtained in different cities of the same country. 101
8.4 Impact of GeoVIPP country classifier on verification accuracy at different stages of the verification pipeline. 101
8.5 False Positive rate comparison with SOTA [2] (Equation (8.4)). 103

Acknowledgements

Throughout my undergraduate studies, I have always thought about pursuing a Ph.D. in Computer Engineering with one primary goal in mind: to acquire technical knowledge and utilize this knowledge to address real-world problems. Then the question arose: Who would be the ideal mentor for my Ph.D.?

After an extensive search, I met Professor Mauro Barni. Mauro, by his expertise and passion, turned the doctoral journey into a smooth and enjoyable experience, resulting in fruitful research. The knowledge I've gained from him will undoubtedly resonate with me for years to come. Actually, my words cannot express how much I'm grateful to him. I am also grateful to my thesis reviewers and members of the committee, Professor Irene Amerini, Professor Roberto Caldelli, Professor Symeon Papadopoulos and Professor Monica Bianchini for their insightful comments, support and their perceptive suggestions. I would also like to thank all of my colleagues in the VIPP group, I learned a lot from them and I wish them all the best in their life.

Finally, thanks to my family, my father and mother who taught me the ABCs of reading and writing, and thanks as well to my brothers and sisters whose words of encouragement always supported me in my journey. Thanks also to my beloved wife for her constant support and encouragement during the years we've spent in our second home Siena.

Chapter 1

Introduction

خيرُ جَليسٍ في الزّمانِ كِتابُ.
*The Best Companion of All is a **Book**.*
Al-Mutanabbi

The recent development of Artificial Intelligence (AI) tools, which enables even non-expert users to generate fake images and videos of extraordinary quality, is raising increasing concerns about the credibility of digital media accessible on the internet and diffused by information channels and social networks. The ability to create such media content with minimum user intervention, coupled with the widespread use of social media raises enormous concerns about the authenticity and credibility of what we see. The consequences of the fake media diffusion in our society could be devastating, virtually changing the meaning of evidence in every domains ranging from everyday life to journalism, criminal justice, and national security. The necessity of distinguishing between fake and original media has renewed interest in Multimedia Forensics [4], a discipline that has attracted the attention of researchers and research funding agencies worldwide [5].

One notable form of manipulated media is the category known as deepfakes. These consist primarily in videos, although the notion of deepfakes could be extended to images or audio recordings, generated using Deep Learning (DL) techniques. In deepfakes, the identities, intentions, and emotions of portrayed human characters are altered and swapped with those of other persons [4,6]. Extensive efforts have been made to differentiate between real and fake videos. Nevertheless, the primary focus has exclusively been on detecting facial manipulations. Comparatively, the detection of artificial non-facial fake videos, produced using DL methods, has garnered considerably less attention. However, new AI tools for generating non-facial videos are rapidly advancing and are projected to soon attain the quality standards of facial deepfakes [7,8].

The majority of efforts to combat deepfake videos involves data-driven methods relying on DL, which have shown superior performance over traditional feature-based techniques [4]. However, DL-based methods typically require a large volume of training examples, which contrasts with the scenarios encountered. For example, in social media, where forensic analysts often need to discern the authenticity of one or two videos only, without resorting to any additional information.

In addition to the escalating phenomena of image and video manipulation, together with the diffusion of deepfakes, a significant challenge is posed by the deceptive recontextualization of photos [9, 10]. This practice involves the misrepresentation of images in news posts or articles, where an image depicting a war, flood, protest, or earthquake, is claimed to be captured in one specific location, but instead, it has been taken from an entirely different country or city [2,11]. To combat the deceptive practice of the geographical recontextualization of images, Image Geolocalization is gathering growing attention, its primary objective being to ascertain the specific geographical location where an image was captured.

All such forms of media manipulation and recontextualization fuel the diffusion of misinformation campaigns by exploiting the powerful impact of visual content in shaping public perception and beliefs. The compelling nature of images and videos significantly influences public opinion and can sway the narrative of a reported event. This trend does not only undermine the authenticity of news and social media posts but also raises profound concerns regarding the trustworthiness and credibility of the information shared in the digital sphere. Consequently, addressing the misrepresentation and recontextualization of images emerges as a critical need, closely intertwined with the larger context of the detection of manipulated media, as it influences public understanding and shapes opinions within the global information landscape.

1.1 Motivation and Contribution

Within the framework outlined in the previous section, the first part of the thesis is dedicated to the detection of artificial non-facial fake videos, covering both of their generation and detection. This endeavour is crucial in addressing a significant research gap, where most research focuses on detecting facial manipulations and less attention has been paid to non-facial videos.

Regarding the generation of non-facial fake videos, we have created a dataset of fake street videos, using video-to-video synthesis tools [7], which we have called the DeepStreets dataset. The dataset consists of 1200 videos, half of them fake and the other half real. In addition, we have built a dataset of human-body fake reenactment videos using the “Everybody Dance Now” framework [8]. We refer to this dataset as the FakeDance dataset. In particular, we have generated 594 videos using different generation settings. The dataset contains the fake videos together with the original real videos used to train the “Everybody Dance Now” network. In both datasets, we also provide the videos at two different quality levels: HQ and LQ. The two datasets are accessible online [1] [2]. When they were created, DeepStreets and FakeDance datasets marked a milestone in the landscape of deepfake datasets, providing the first comprehensive collections of non-facial fake

[1]`http://clem.dii.unisi.it/~vipp/datasets.html`
[2]`https://bit.ly/3J2IENp`

videos. While most existing datasets were focusing (and still focus) on facial manipulation, these datasets introduced a crucial shift, encompassing a diverse range of non-facial fake videos. Their creation was not without challenges, demanding significant computational resources, involving the use of high-performing GPUs for weeks. Moreover, the datasets provide a crucial resource to develop, compare and improve detection methods for deepfake videos. Particularly, the datasets are necessary for detectors leveraging DL, which heavily rely on substantial data for robust training and testing.

With regard to the detection of non-facial fake videos, we have investigated the detectability of these kinds of videos (both DeepStreets and FakeDance videos), using data-driven methods and feature-based methods. From such analysis we could understand that each detection method has its peculiarities and strengths and could work well in different operative scenarios. Nevertheless, the generalization capability of both methods strongly depends on the experimental conditions.

In addition to the detection of non-facial fake videos, we have also addressed a common problem encountered by deepfake detectors. The majority of current methods, in fact, rely on DL and require extensive volumes of task-specific representative training data to ensure effective training. To avoid this necessity, we have developed a lightweight deepfake detector leveraging simple video-coding features. The detector is specifically designed to operate in scenarios characterised by data scarcity, catering to instances when forensic analysts have very limited examples available of fake videos. Specifically, the detector is based on a straightforward footprint derived from the motion prediction modes within the video sequence. The footprint is extracted from video sequences and utilized to train a basic linear SVM classifier. The findings we have obtained from experiments conducted on three distinct datasets demonstrate the effectiveness of our approach, showcasing that the detector requires only a minimal number of video samples for training.

The second part of the thesis is devoted to Image Geolocalization (as a tool to detect and prevent the geographical recontextualization of media), with a particular focus on recognizing cities and countries from images. Previous efforts in this field have concentrated on the estimation of the GPS coordinates of the scene depicted in the image [2,12]. In this thesis, we adopt a different approach, aiming at identifying the country or city where an image was captured. In many cases, recognizing either the country or the city is of a greater importance, from both semantic and forensic perspectives, than estimating the GPS location. Simply knowing the country or city depicted in an image is often sufficient to discern the authenticity of news content, particularly in the battle against fake news and misinformation. Moreover, identifying either the country or the city potentially involves a less complex process than accurately determining the geographical GPS coordinates. Being human constructs, countries and cities typically delineate geographic regions sharing similar human-related features, such as historical landmarks, language, architecture, and social customs. These distinct attributes serve as viable cues to identify the specific country (or city) wherein an image has been taken.

As a first, crucial, step to address the country and city recognition tasks, we have collected a dataset of 3.8 million geo-tagged images, obtained through web crawling. These images exclusively feature urban outdoor scenes, chosen for their relevance in highlighting architectural and semantic disparities among countries. The images were systematically crawled based on a list of 243 countries, ensuring a comprehensive representation of the entire world's nations. The dataset was named VIPPGeo and it is publicly available online [3]. The VIPPGeo dataset stands as a significant milestone in the field of Image Geolocalization research, given the scarcity of similar datasets available online. Furthermore, the development and testing of DL algorithms for Image Geolocalization, which is the current focus of many researches, necessitate extensive data that encapsulates the diverse architectural, stylistic, and social differences among countries and cities. The performance of deep neural networks heavily relies on abundant, diverse training data for optimal training and testing.

In the subsequent phase of our research, we have employed the VIPPGeo dataset to train a DL model for Country Recognition. Remarkably, our findings indicate that instructing the network to directly identify the country yields superior results compared to estimating the geo-coordinates and subsequently tracing them back to the country where the picture was taken.

The last contribution of this thesis in the realm of Image Geolocalization (hereafter referred to as City Verification), aims at identifying the city where a photo has been captured. With respect to Country Recognition, City Verification poses a substantially harder challenge due to two critical factors: i) the vast number of cities in the world, and ii) the considerable variability of scenes originating from the same city. To address the first challenge, we have departed from the conventional inference and classification methods employed thus far and re-framed the city recognition problem as a verification task. Specifically, we have developed a system based on a Siamese network designed to determine whether a source image was captured in a target city, with the assumption that a small set of images from the target city is available. For the second challenge, our system compares the query image to several reference images from the claimed city. In addition, we have designed our system in such a way to assign greater voting power to reference images that exhibit semantic similarity to the query image.

In summary, the key contributions of this thesis are as follows:

- Creation of a new dataset for city streets fake videos: the DeepStreets dataset.
- Creation of a new dataset for Human Body Reenactment Fake Videos: the Fake-Dance dataset.
- Development of detectors for non-facial deepfake identification.
- Development of a feature-based method for deepfake video detection that can work under data scarcity conditions.
- Creation of an Image Geolocalization dataset, containing almost 4 million urban images: the VIPPGeo Dataset.
- Development of a novel classification method for automatic Country Recognition.
- Introduction of a novel framework and development of a DL tool for City Verification.

[3]`https://github.com/alamayreh/VIPPGeo_Dataset`

1.2 Thesis Overview

This thesis is structured into two distinct parts. The first part deals with the detection of non-facial deepfake videos and addresses the detection of fake videos within data scarcity scenarios. Chapter 2 serves as an introduction to multimedia forensics, covering the generation and detection of deepfake videos. Chapter 3 focuses on our specific contributions to detecting and generating non-facial deepfake videos, particularly the Deepstreets fake videos. Subsequently, chapter 4 explores our work on FakeDance fake videos. Following this, chapter 5 centres on our contributions to the detection of deepfake videos in data-scarce conditions.

In the second part of the thesis, the focus shifts to Image Geolocalization. Chapter 6 introduces the topic of Image Geolocalization and provides an overview of the current state of the field. Chapter 7 is dedicated to our work on Country Recognition, while chapter 8 outlines our efforts in City Verification.

Finally, chapter 9 summarizes the work done during the thesis and explores some of the unresolved issues associated with the detection of non-facial fake videos and Image Geolocalization.

1.3 Publications

The research carried out within this thesis has led to the following publications.

- O. Alamayreh, and M. Barni, 2021, August. "Detection of gan-synthesized street videos." In *2021 29th European Signal Processing Conference (EUSIPCO).* (pp. 811-815). IEEE.
- O. Alamayreh, C. Fascella, S. Mandelli, B. Tondi, P. Bestagini, and M. Barni, M., 2022. "Just Dance: Detection of human body reenactment fake videos." Submitted to *EURASIP*, under review.
- O. Alamayreh, G. M. Dimitri, J. Wang, B. Tondi, and M. Barni, 2023, June. "Which country is this picture from? New data and methods for DNN-based country recognition." In *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* (pp. 1-5). IEEE.
- O. Alamayreh, G. M. Dimitri, J. Wang, B. Tondi, and M. Barni, "A Siamese Based System For City Verification." In *ECAI 2023* (pp. 69-76). IOS Press.
- J. Wang, O. Alamayreh, B. Tondi, A. Costanzo, and M. Barni, 2022. "Detecting Deepfake Videos in Data Scarcity Conditions by Means of Video Coding Features." *APSIPA Transactions on Signal and Information Processing,* 11(2).

  Other Papers published during my PhD on topics that are not covered by the thesis

- J. Wang, O. Alamayreh, B. Tondi, and M. Barni, M., 2023. "Open Set Classification of GAN-based Image Manipulations via a ViT-based Hybrid Architecture." In

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (pp. 953-962).

- J. Wang, O. Alamayreh, B. Tondi, and M. Barni, 2022, June. "An Architecture for the detection of GAN-generated Flood Images with Localization Capabilities." In *2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)* (pp. 1-5). IEEE.

1.4 Activity Within Research Projects

The research carried out during the PhD and described in this thesis, has been carried out in collaboration with the Semantic Forensics (SemaFor)[4] project, funded by DARPA and the Air Force Research Laboratory (AFRL) under agreement number FA8750-20-2-1004[5], and aiming at developing forensics-semantic tools to automatically detect, attribute, and characterize falsified, multi-modal media assets (e.g., text, audio, image, video) to defend against large-scale, automated disinformation campaigns, and to ensure the integrity, semantic consistency, source, and intent of media assets such as images, videos, and documents. The program's forensic tools included semantic detection algorithms to determine if multi-modal media assets have been generated or manipulated. Attribution algorithms discern the multi-modal media origin, whether from a specific organization or individual. Characterization algorithms assess whether the media was manipulated or generated with malicious intent. These SemaFor tools aim to detect, attribute, and characterize adversary disinformation campaigns.

In addition to SemaFor, the activity of the thesis has also been supported by the PREMIER project under contract PRIN 2017 2017Z595XS-001, funded by the Italian Ministry of University and Research (MUR) [6].

[4]`https://engineering.purdue.edu/SEMAFOR`

[5]The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA and AFRL or the U.S. Government.

[6]`https://www.darpa.mil/program/semantic-forensics`

Part I

DeepFake Detection

Chapter 2
DeepFakes and Multimedia Forensics

One Never Notices What has been Done;
One can Only See What Remains to be Done.
Marie Curie

Image manipulation has an old history and can be traced back to the 1860s, shortly after Niepce's creation of the first photograph in 1814[1]. Since then, manipulated images have been often used to shape public perception and influence historical narratives. With the start of the 1990s, with the introduction of high-resolution digital cameras, powerful personal computers, and advanced photo editing software, the manipulation of photographs has become a ubiquitous practice, and, consequently, detecting forged photos has become an urgent and challenging need. Even later, starting from 2014, the landscape of media manipulation underwent a significant transformation with the advent of AI, marking the dawn of a new era in the history of fake media generation and detection). Figure 2.1 sketches the historical evolution of manipulated media.

In this chapter, we provide a brief introduction to the field of Multimedia Forensics, followed by a review of the most common state-of-the-art methods for the generation and detection of Deepfake Videos. In this regard, we observe that the research described in the first part of the thesis accounts for the research activity we have carried out in the first part of the PhD, roughly until the beginning of 2022. The techniques described in such a part were at the edge of research at the time in which they have been proposed and represented an important contribution to the state-of-the-art. From 2022 on, our research activity shifted to geolocalization, so our research on deepfake detection has not followed the subsequent, extremely rapid, progress in the field. For this reason, the analysis of the state of the art and the comparison of the proposed techniques to existing methods consider the state of the art up to 2022, without any attempts to compare our techniques to the methods proposed later on.

2.1 Multimedia Forensics

In today's digital era, digital multimedia content has largely surpassed traditional media as the favoured source of information, particularly among the youngest generations. At the same time, millions of users globally rely on multimedia editing tools like GIMP, Photoshop, Lightroom, and After Effects as essential tools for their work [13], modifying

[1]`https://en.wikipedia.org/wiki/History_of_photography`

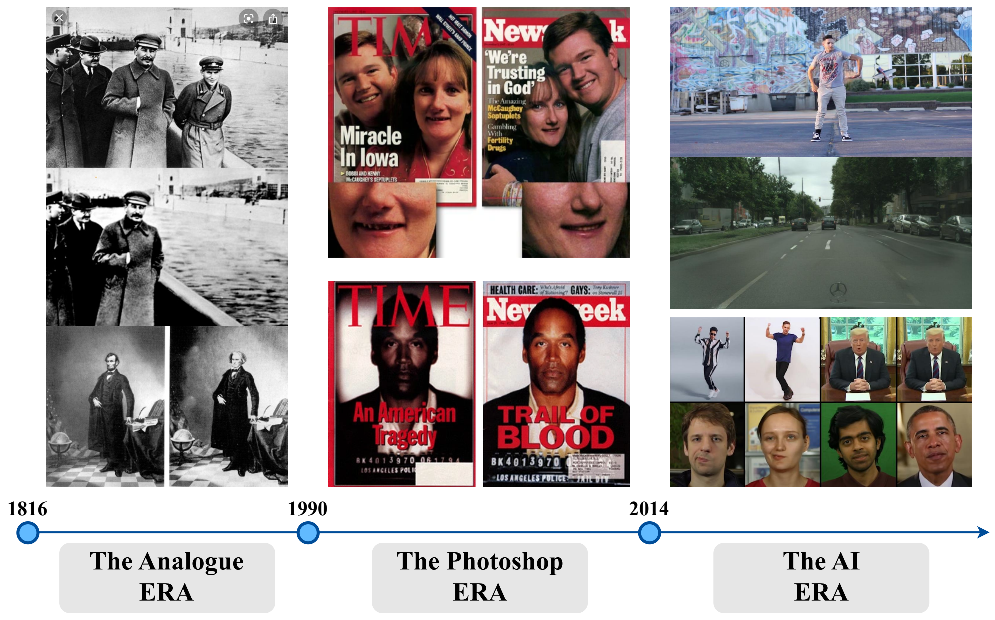


Figure 2.1: The historical evolution of image manipulation.

the original content of multimedia content with respect to the original, pristine, versions produced by the acquisition devices. By also considering the increasing diffusion of AI manipulation tools and the ease with which contents can be shared across social media platforms, the need to examine, authenticate, and verify the origins of digital media is more and more evident, triggering the birth of a new discipline, known as Multimedia Forensics [14, 15].

Multimedia forensics involves principles and approaches from diverse research areas, such as computer vision and signal processing, to analyze and evaluate digital content. It focuses on uncovering the truth behind images, videos, and audio recordings, determining their legitimacy, identifying manipulations or alterations, and shedding light on the circumstances surrounding their creation or modification.

For more than two decades, researchers in multimedia forensics have extensively explored the challenge of identifying manipulated content. The methodologies developed in this period can identify a wide range of modifications of images and videos, including splicing, copy-move, resampling, double compression, blurring, semantic alterations, and the deletion or addition of video frames [4, 13, 16].

Recently, advanced AI architectures like Autoencoders [17, 18], Generative Adversarial Networks (GANs) [19] and Diffusion Models [20] capable of producing remarkably realistic fake images and videos have been developed. The emergence of these kinds of deepfake media has raised huge concerns regarding the authenticity and credibility of what we see. At the same time, researchers are exploiting AI techniques as a potential solution to detect fake media.

Broadly speaking, Multimedia Forensics primarily explores three key areas:

1. Forgery detection, focusing on authenticating images or videos and identifying any manipulations.

2. Source identification, namely tracing the origin and history of digital content, such as determining the camera model or brand used to acquire them, or whether the content has been downloaded from social media.

3. Deepfake detection, encompassing the identification of synthetic media where a person in an existing image or video has been replaced with someone else's image [4, 13], or the detection of full synthetic videos like Deepstreets and FakeDance videos [3, 21].

Forgery detection covers the identification of any changes made to multimedia content, potentially undermining its authenticity. Such manipulations range from manual interventions, performed by individuals using software like Photoshop or GIMP, to computer-generated modifications, such as Computer-Generated Imagery (CGI). Both methods aim to alter the content, potentially distorting the original information within the multimedia file [13].

By referring to images and videos, a non-exhaustive list of the most common manipulation operations includes (1) splicing, where a portion of one image is copied and pasted into another [22]; (2) copy-move, which duplicates a section within the same image or removes a portion of it [23]; (3) resampling, occurring whenever an image is resized, rotated, or subjected to an affine transformation [24]; (4) contrast enhancement, typically used to adjust lighting within an image; and (5) inter-frame video forgery, manipulating videos by introducing or removing frames to alter the content maliciously [25, 26]. Multimedia forensics endeavours to identify these forms of forgery by analyzing various traces left behind by photo or video editing operations [4, 13, 27–29].

Identifying the source of multimedia content stands as a critical forensic challenge. It is not solely about validating the authenticity of multimedia content; it involves unravelling the origin of images or videos. This process delves into more than confirming the content's legitimacy; it helps discerning if two photos or videos originate from the same camera, smartphone, or even from the same camera brand. Moreover, source identification extends to uncovering the social network or messaging app from which the content has been downloaded, providing possible insights into the content's origin from a specific social media platform. Moreover, source identification plays a significant role in scenarios such as crime scene analysis, verification of news, or validation of the source of a multimedia file downloaded from a social media platform [30–32]

Deepfake detection involves identifying and recognizing manipulated or synthetic media content created through AI, encompassing images and videos. Typically, this manipulated media focuses on human faces, and in some instances, it may also involve generic scenes. In the following sections, we delve into the current state-of-the-art techniques for both the generation and detection of deepfake videos. In this context, AI-based methods demonstrate promising results compared to traditional approaches. However, they encounter three key limitations, which are very relevant for Multimedia Forensics applications: the demand for extensive training data, the difficulty of interpreting the results provided by the networks, and the vulnerability to adversarial attacks [4, 13].

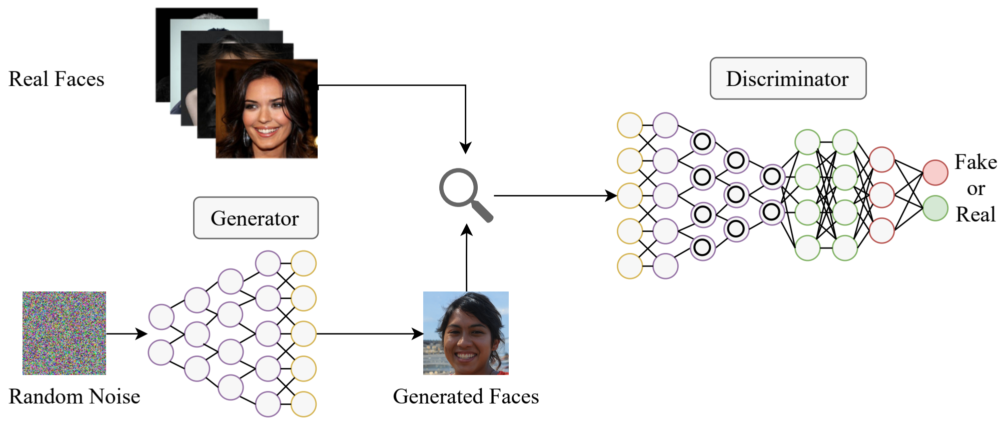


Figure 2.2: A typical GAN designed to produce synthetic human facial images.

2.1.1 Generation of Deepfake Videos

With the rise of DL techniques, the automatic generation of falsified media has reached an unprecedented level. Several manipulation approaches based on AI have been proposed in the literature for fake image and video generation. These methods generally rely on three DL architectures; GANs [19], Autoencoders [17, 18] and Diffusion Models [20].

A GAN architecture, as outlined in [19], operates within a game-theoretic framework, training two networks a generator G and a discriminator D in an adversarial manner. Figure 2.2 illustrates the architectural framework of a standard GAN. Using an input noise sample $\boldsymbol{z}$, the generator produces new samples that closely resemble the distribution of a specific source data p_{data}. The discriminator is trained with samples generated by G and real data distributed as p_{data}. The discriminator analyzes these samples and tries to differentiate between real and synthesized samples generated by the generator. In other words, D and G play the following two-player minimax game with value function $V(G, D)$:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\boldsymbol{x}\sim p_{\text{data}}(\boldsymbol{x})}[\log D(\boldsymbol{x})] + \mathbb{E}_{\boldsymbol{z}\sim p_{\boldsymbol{z}}(\boldsymbol{z})}[\log(1 - D(G(\boldsymbol{z})))] \tag{2.1}$$

where $G(\boldsymbol{z})$ indicates the generator's output for a given noise sample $\boldsymbol{z}$, $D(\boldsymbol{x})$ indicates the discriminator's assessment of the probability that the original data $\boldsymbol{x}$ is a real sample, and $D(G(\boldsymbol{z}))$ denotes the discriminator's probability estimation that the synthetic sample $G(\boldsymbol{z})$ is real. Additionally, $\mathbb{E}_{\boldsymbol{x}}$ and $\mathbb{E}_{\boldsymbol{z}}$ denote the mean likelihood across original and synthetic data, respectively. According to equation (2.1), the discriminator is trained in such a way to maximize $\log D(\boldsymbol{x})$ and $\log 1 - D(G(\boldsymbol{z}))$, while the generator G aims at minimizing $\log(1 - D(G(\boldsymbol{z})))$, with no direct influence on $\log D(\boldsymbol{x})$.

In addition to GANs, autoencoders are often used to generate falsified media. A popular approach to create deepfake videos is face replacement, where two autoencoders, with a shared encoder, are trained in parallel on two datasets (see Figure 2.3). The encoder extracts latent features from face images, while the decoder is used to reconstruct face images. The datasets include face images of person A and B, respectively. The idea is to allow the shared encoder to learn common features for both persons A and B while keeping the corresponding decoders detached. The resulting architecture is shown in Figure 2.3. To swap the faces between person A and B, a video framing person A is fed to the common encoder. Afterwards, the resulting features in the latent space are input to the decoder trained on B. This principle is applied in several manipulation tools, like DeepFaceLab [33].

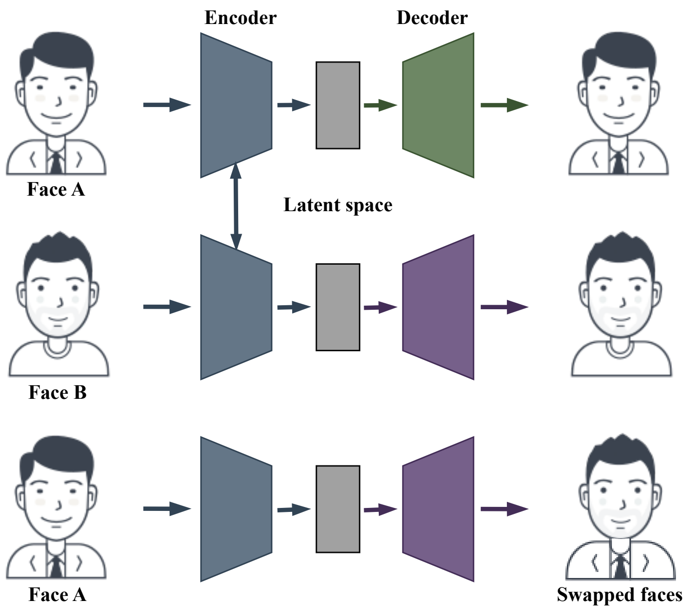


Figure 2.3: Autoencoder-based face swapping.

GANs and autoencoders can also be used together to improve the quality of the synthesised videos. High-quality deepfakes based on GANs called Faceswap-GAN [34] have been obtained by adding the adversarial loss and perceptual loss implemented in VGGFace [35] to the encoder-decoder architecture. The VGGFace perceptual loss is added to make eye movements more realistic and consistent with input faces. The autoencoder performance is improved by adding an adversarial loss during training, under a GAN framework. The discriminator tries to distinguish between the fake images generated by the autoencoder and the real images. At the equilibrium point, the autoencoder will generate realistic fake images. This approach is applied in other works such as DFaker [36].

Another intriguing contribution to the field is Face2Face [37]. This innovative technique enables real-time facial reenactment within monocular target video sequences, allowing the animation of facial expressions in the target video through the emulation of a source actor. The result is a meticulously re-rendered output video, with photo-realistic quality.

Additionally, Neural Voice Puppetry, detailed in [38], presents another important advancement. This method is centred on audio-driven facial video synthesis. It empowers the generation of videos featuring a speaking individual, based on an audio sequence of a different person, all achieved through a 3D representation of the face.

More recently, Diffusion models have emerged as the latest cutting-edge family of deep generative models. They have challenged the longstanding dominance of GANs in the complex task of image and video synthesis and have showcased promise across various domains. These models can be applied to computer vision tasks, natural language processing, temporal data modelling, multi-modal modelling, robust machine learning, and interdisciplinary applications like computational chemistry and medical image reconstruction [20, 39].

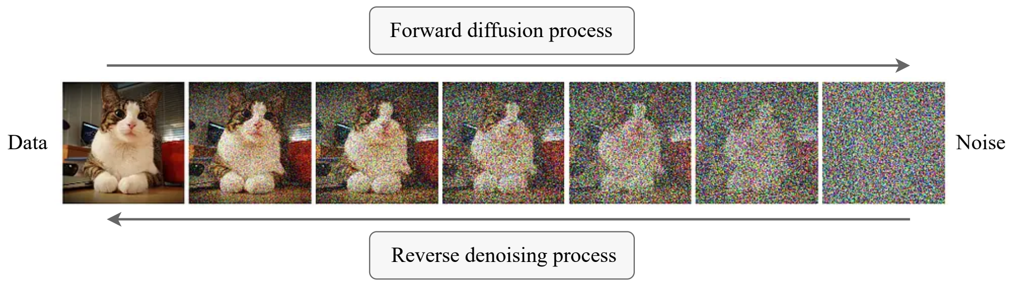


Figure 2.4: Progressive transformation of noise into real data and vice versa within a Diffusion model.

Moreover, diffusion Models represent a family of probabilistic generative models. During training, they gradually introduce Gaussian noise to the original data through a forward diffusion process. These models subsequently learn to reverse this procedure, eliminating the noise in a reverse diffusion process. At each step of this reversal, a score function estimation aims at determining gradients that point toward data with higher likelihood and less noise (see Figure 2.4). Upon training, these models can generate new synthetic data by sampling from the distributions they have learned. Examples of Diffusion Models include DALL-E 2 [40], Imagen [41], Stable Diffusion [42] and GLIDE [43].

Deepfake datasets play a crucial role in training robust detectors, enabling them to recognize manipulated content. However, creating high-quality realistic fake videos demands significant resources and effort. While in the past, only a few deepfake datasets were available, recently the number of available deepfake datasets has grown significantly [4]. For an overview of the most popular deepfake datasets, please refer to Table 2.1.

In addition to the techniques for generating manipulated images and videos of human faces, some works also focus on the creation of generic fake videos. Vid2vid [7] introduces a video-to-video synthesis technique that can be used to generate manipulated videos from diverse sources such as car dash cameras or other non-facial videos. This is achieved by employing semantic segmentation at the input, facilitating the creation of desired fake video sequences. Another notable example is the work presented in [8], where the authors proposed a video synthesis generator capable of transferring motion between subjects, enabling the manipulation of a person's body movements based on provided input body poses.

Taking into account the potential societal implications of videos produced using these methods and to fill the gap in the literature, we have focused our work on the gener-

Table 2.1: List of most popular deepfake datasets.

Dataset	**Description**	**Year**	**Reference**
FaceForensics++	The dataset contains 1000 original video sequences manipulated by four automated face manipulation methods: Deepfakes, Face2Face, FaceSwap, NeuralTextures, and FaceShifter.	2019	Rössler et al. [44]
DFDC	The DFDC (Deepfake Detection Challenge) dataset comprises over 100,000 video deepfake detection.	2019	Dolhansky et al. [45]
Celeb-DF	The dataset includes 590 original videos collected from YouTube with subjects of different ages, ethnic groups and genders, and 5639 corresponding DeepFake videos.	2020	Li et al. [46]
DeeperForensics-1.0	The dataset comprises 60,000 videos, totalling 17.6 million frames. It consists of 48,475 source videos and 11,000 manipulated videos. The source videos are collected from 100 actors from 26 countries.	2021	Jiang et al. [47]
WildDeepfake	The dataset consists of 7,314 face sequences extracted from 707 deepfake videos that are completely collected from the internet.	2021	Zi et al. [48]
KoDF	The Korean DeepFake Detection Dataset (KoDF) is a large-scale collection of synthesized and real videos focused on Korean subjects. It encompasses 62,166 real videos along with 175,776 fake videos involving 403 subjects.	2021	Kwon et al. [49]

ation and detection of non-facial videos generated by [8] and [7]. More comprehensive information is given in Section 4.2 and Section 3.2.

2.1.2 Deepfake Detection Methods

The techniques developed so far to distinguish between real and deepfake videos belong to two main classes: those based on handcrafted features and those relying on end-to-end completely automatic detectors based on DL. On the one hand, handcrafted methods exploit a set of predefined traces that result from the production pipeline of deepfakes, like visual artifacts [50], semantic inconsistencies [51, 52], unnatural eye blinking [53], face-warping artifacts [54], face landmark locations [55], and inconsistencies of head pose [56]. For instance, Matern et al. [50] have developed a detector using several visual features that focus on the eyes, teeth, facial contours and other visual artefacts. Li et al. [53] have exposed deepfake videos through unnatural eye-blinking signals. Human eye-blinking possesses a unique frequency and duration not present in deepfake videos. In [51, 52] the absence of variations linked to heartbeat has been used to reveal deepfakes, a concept previously explored in [57] for computer-generated faces. Nonetheless, in [51], the coherence of these biological signals has been assessed both spatially and temporally. In [54], Li et al. have highlighted the limited resolution in current deepfake generation methods, often necessitating further warping to align with the original face in the source video. This process leaves recognizable traces that a Convolutional Neural Network (CNN), focusing on the facial region and its surroundings, can detect. Meanwhile, Yang et al. [55] suggest that GAN-based face synthesis produces highly detailed yet spatially unconstrained faces. As a result, the locations of facial landmarks like the eyes, nose, and mouth tips may be used as discriminative features for authenticating GAN-generated images. This issue is also apparent in deepfake videos and can be revealed through 3D head pose estimation techniques, as outlined in [56]. A different approach is followed in [58]. The idea is to protect individuals by acquiring some peculiar soft traits that characterize them and are very difficult to reproduce for a generator. In particular, it is observed that facial expressions and head movements are strongly correlated, and changing the former without modifying the latter may expose a manipulation. However, to apply this approach, a large and diverse collection of videos in a wide range of contexts must be available for all individuals of interest.

On the other hand, DL-based methods, specifically those based on CNNs, deliver end-to-end solutions to the forgery detection problem. Most of these methods rely on a frame-based analysis and resort to CNNs classification and automatically learned features [4,59]. Approaches like MesoInception [60], Capsule [61], XceptionNet [44], EfficientNet [62, 63], provide superior performance compared to the traditional handcrafted ones [4].

Afchar et al.'s [60] have proposed two simple architectures employing a small number of levels and parameters to leverage different image features. The first model, Meso-4, integrates four layers of convolutions and pooling, succeeded by a dense network housing a single hidden layer. Conversely, the MesoInception-4 model is built on a variant of the inception module integrating dilated convolutions. Nevertheless, experiments that have been outlined in the work of Rossler et al. [44] distinctly demonstrate that, in a supervised setting, deeply layered general-purpose networks such as Xception [1], DenseNet [64], and Inception [65] outperform shallow forensics-oriented CNNs and methods relying on handcrafted features, especially when dealing with the heavy compression typical of video codecs [4].

In general, deepfake generation methods often conduct face manipulation frame by frame, leading to potential inaccuracies in capturing natural face movements. Several techniques have emerged to exploit this fact by relying on temporal correlation and motion artefacts. Usually, these methods resort to Recurrent Neural Networks (RNN) [66] or Long Short-Term Memory (LSTM) [67] based classifiers. In [68, 69, 69] a convolutional LSTM network has been used to exploit such dependencies and to improve upon single-frame analysis. Sabir et al. [70], instead, introduce a solution based on RNN convolutional models. Moreover, in [71], the authors demonstrate that it is even possible to leverage the temporal information within the video to enhance the resilience of DL-based deepfake detectors against black-box adversarial attacks.

Further, approaches combining handcrafted features with DL features have also been proposed. In [72, 73], a combination of handcrafted and DL-based temporal features has been employed. Specifically, features based on Inter-frame prediction errors have been investigated in [73] jointly with an LSTM-based network cable to learn the temporal correlation among consecutive frames. In [72], typical motion-related features, that is, the optical flow fields have been extracted and used as input of a CNN classifier. However, the performance of such approaches relying on LSTM architectures is often lower than those achieved by frame-based methods [59].

Despite the advantages of DL methods, they still suffer from significant shortcomings like the difficulty of interpreting the output they provide, the vulnerability to adversarial attacks, and the difficulty of maintaining good performance when the training and test phases work in mismatched conditions.

Recently, traditional CNNs have gradually been substituted by Vision Transformers (ViTs) [74] as the preferred model in the field of computer vision. While CNNs have a proven track record in various computer vision tasks and handle large-scale datasets efficiently, ViTs offer advantages in scenarios where global dependencies and contextual understanding of images are crucial. When trained on extensive datasets, ViTs achieve excellent results and approach or surpass the state of the art on multiple image recognition benchmarks. Further insights on ViT can be found in section 8.2.

Several deepfake detection frameworks rely on ViTs as the backbone for their detectors. Khormali et al. [75] present an end-to-end deepfake detection framework that utilizes transformer models. This approach harnesses the distinctive features of transformers, capturing hidden traces of perturbations within local image features and global pixel relationships across different forgery scales. Dong et al. [76] propose the Identity Consistency Transformer (ICT), a ViT-based approach that protects celebrities from DeepFake by detecting identity consistency in the suspect image, i.e., whether the inner face and the outer face belong to the same person. Wang et al. [77] introduce the Multimodal Multi-scale TRansformer (M2TR), which operates on diverse patch sizes to identify local inconsistencies across different spatial levels. M2TR additionally learns to detect forgery artefacts in the frequency domain, complementing RGB information through a specialized cross-modality fusion block. Moreover, some methods integrate both CNNs and ViTs within a single architecture for deepfake detection, leveraging the strengths of CNNs with ViTs to enhance detection accuracy, as demonstrated in [78–80].

Another investigation avenue for the identification of manipulated videos involves video coding. The editing process of a video sequence invariably concludes with the re-encoding of the edited video, potentially using a different codec or coding parameters. Consequently, footprints left by video codecs during recompression have been extensively studied to detect conventional manipulations and video editing [81]. In many cases, the traces left by the double encoding process can be exposed by looking at the distribution of the macroblock prediction types in the frames of the re-encoded videos [82–85]. In [84] and [85], the statistics of the prediction modes have been used to detect fake high-quality videos. In particular, in [84], a very simple footprint obtained by counting the number of macroblocks of type *Intra*, *Inter* and *Skip* has been successfully employed to distinguish low-quality H.264 (AVC) videos, re-encoded in H.265 (HEVC), from native HEVC videos, i.e., obtained from an uncompressed video sequence. In this thesis, we have exploited video-coding features to develop a lightweight deepfake detector. For a more in-depth exploration of this topic, refer to chapter 5.

Concerning the detection of non-facial manipulations, only a few works have been proposed. In "Everybody Dance Now" [8], a forgery detector based on CNNs has been presented. Such a detector corresponds to the discriminator used by GANs to create the fake videos, which has been trained in parallel with the generator. However, we argue that in a real-world scenario, the forgery detector does not know the details of the generation process of the fake media, and then any practical detector has to be trained separately from the generator, only exploiting the availability of fake videos generated by the model.

The lack of methods specialized for non-facial manipulation detection highlights that some efforts must be put in this direction. In this thesis, our focus lies on the examination of the detectability of different kinds of non-facial synthetic videos. To this end, we have constructed two datasets, referred to as FakeDance and Deepstreets which serve as the foundation for the subsequent development of deepfake detection methods. More comprehensive information on this topic can be found in chapter 3 and chapter 4.

Chapter 3

Detection of GAN-synthesized Street Videos

Segui il tuo corso et lascia dir le genti.
Follow Your Road and Let the People Say.
Dante Alighieri, da "Divina Commedia"

Research on the detection of AI-generated videos has focused almost exclusively on face videos. Manipulations like face swapping, face reenactment and expression manipulation have been the subject of intense research with the development of a number of efficient tools to distinguish artificial videos from genuine ones. Nonetheless, much less attention has been paid to the detection of artificial non-facial videos. Yet, new tools for the generation of such kinds of videos are being developed at a fast pace and will soon reach the quality level of deepfake videos. This chapter investigates the detectability of a new kind of AI-generated videos framing driving street sequences (here referred to as DeepStreets videos), which, by their nature, can not be analysed with the same tools used for facial deepfakes. Specifically, we present a simple frame-based detector, achieving very good performance on state-of-the-art DeepStreets videos generated by the Vid2vid architecture. Noticeably, the detector retains very good performance on compressed videos, even when the compression level used during training does not match that used for the test videos.

3.1 Motivation

With regard to the detection of fake videos generated by means of AI techniques based on DL, most of the attention has been devoted to the detection of face facial videos, known as Deepfakes. As a consequence, a wide range of tools have been developed to detect several kinds of deepfake manipulations, including face swapping [86] and face re-enactment [87]. The interest towards this kind of videos is justified by the fact that information conveyed by speeches and facial expressions has an essential role in human social interactions. Besides, human face manipulation techniques have received increasing attention in the last years, reaching a maturity level and making them ready to be exploited in several real-life applications (including malicious ones) [4].

In comparison, the detection of non-facial artificial video sequences has received much less attention. Yet, new AI tools for the generation of non-facial videos are being developed at a fast pace and will soon reach the quality level of deepfake videos [7]. To move a first step to fill this gap, in this chapter, we focus on the detection of a new class of fake videos, hereafter referred to as DeepStreets, framing driving street sequences similarly to those described in [7, 88, 89].

To start with, we observed that the forgery detection methods designed to work on facial videos can not be directly applied to DeepStreets. The main reason for such a difficulty is that deepfake detectors rely strongly on the facial features of the videos, all the more that they usually crop the face region before inputting the video frames to the detectors. Besides, some detectors are based on the analysis of a set of geometric and semantic features that are directly related to human faces; as for instance eye color [50] or facial pose [56]. No such features obviously exist in DeepStreets videos, which are characterised by greater diversity and for which it is not possible to identify a single attention region to focus the analysis on.

To address the above limitations, we have developed an end-to-end detector based on CNNs. The detector works on a frame-by-frame basis and does not focus on any specific region of the input frames.

To train (and test) the detector, we have generated 600 videos of fake driving scenes by means of the Vid2vid architecture presented in [7] and briefly summarized in Section 3.1.1. To evaluate the generalization capability of the detector and its ability to work in non-ideal conditions, we have considered both raw sequences and videos compressed at different quality levels. The experiments we have carried out prove the validity of the detector, which exhibited an extremely high accuracy, demonstrating that even DeepStreets kind of videos can reliably be distinguished from real videos (at least in the highly controlled setting adopted in this chapter). A remarkable property of our detector is its immunity to video compression, given that the detection accuracy remains good even when the compression setting used during training is different than that used for the test videos.

The rest of this chapter is structured as follows. In Section 3.1.1, we introduce the Vid2vid synthesis framework. In Section 3.2, we present the dataset we have used in our experiments and the detection pipeline. The results of the experiments we have carried out to validate the developed detector are described in Section 3.3. In Section 3.4, we wrap up the chapter with some concluding remarks.

3.1.1 Video-to-Video synthesis

Most deepfake manipulation methods have been developed to create fake videos that contain human faces. Their counterpart for non-facial fake videos has been less explored. However, general-purpose video synthesis techniques are presented in several forms, including video prediction [90], unconditional video synthesis [91] and conditional video synthesis [7]. The first methods produce future video frames based on past frames. The second kind of methods take in random variables and generate synthesised videos. The latter methods convert input semantic sequences into photorealistic videos.

In our work, we have used a conditional video synthesis method, known as Vid2vid [7], to generate the training and testing datasets. Vid2vid is a video-to-video synthesis architecture operating under the generative adversarial learning framework. The goal of Vid2vid is to convert an input semantic video, for example, a sequence of semantic segmentation masks, to an output photorealistic video that represents the theme of the source video. Vid2vid is considered the counterpart of image-to-image translation methods such as Pix2pix [92] and Covst [93]. The advantage of Vid2Vid with respect to the application of image-based architectures is that working on a frame-by-frame basis, image-based techniques do not guarantee the temporal coherence of the synthetic output video. In contrast, Vid2vid is designed in such a way to preserve the temporal dynamics of the source video.

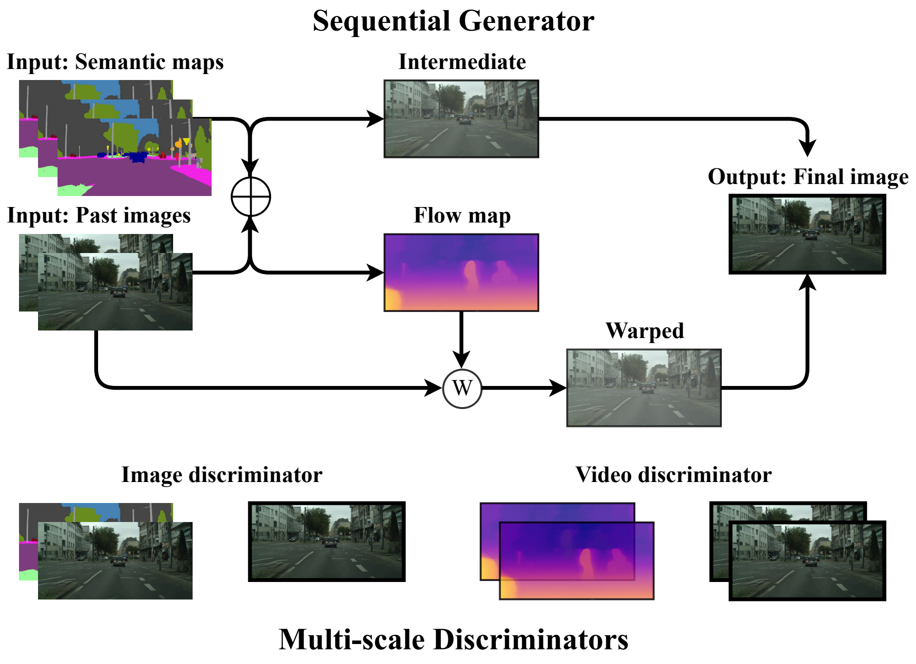


Figure 3.1: Vid2vid network architecture.

The architecture of Vid2vid consists of two networks: a sequential generator and a set of multiscale discriminators. The generator takes as input a sequence of segmentation semantic maps and the frames generated previously and produces an intermediate frame and a flow map. The source of the semantic maps could be extracted from a video or generated manually. Afterwards, the flow map is used to warp the previous frame. Then, the warped frame is combined with the intermediate frame to output the final frame, which is then used to generate the next frame and so on. The multiscale discriminators contain one discriminator for images and one for videos. The image discriminator analyses the input semantic map and the output images to ensure that each output frame resembles a real image. Simultaneously, the video discriminator takes in the flow maps and neighbouring frames to guarantee the temporal coherence of the generated videos. The architecture of Vid2vid is depicted in Figure 3.1.

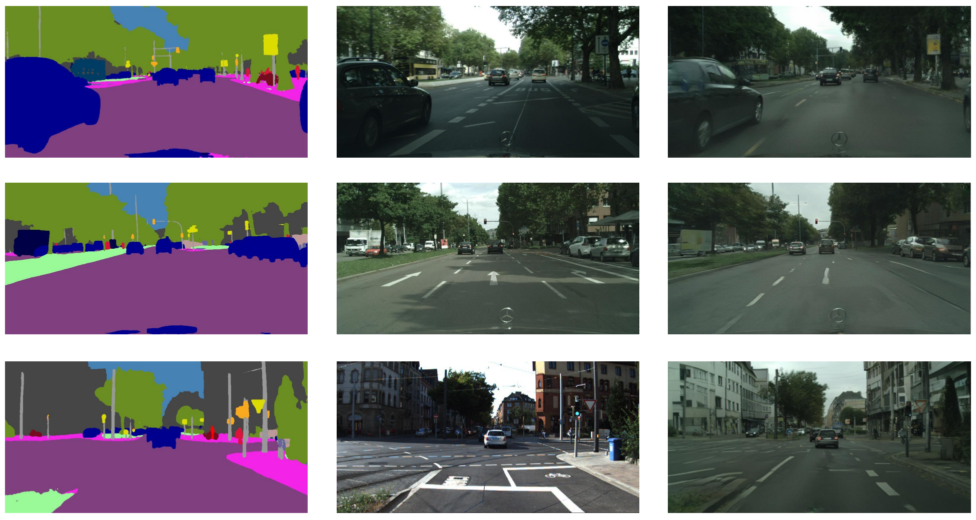


Figure 3.2: Example of frames generated by vid2vid architecture. The first row displays frames from Cityvid, the second row exhibits Citywcvid frames, and the third row presents frames from the Kittivid sub-datasets.

Two versions of Vid2vid have been released: Few-shot video-to-video synthesis [89] and World-Consistent Video-to-Video Synthesis [88]. The first work increases the generalisation capability of Vid2vid by using a novel network weight generation module based on an attention mechanism. The latter improves the generated videos' temporal consistency by exploiting, so-called, *guidance images*, which are a consolidation of the *3D world* rendered in the previous frames. In our experiments, we have used Vid2vid and Wc-vid2vid architectures to generate the DeepStreets dataset, as described in section 3.2.

3.2 Methodology

In this section, we describe the generation process of the DeepStreets dataset, along with the developed forgery detector.

3.2.1 DeepStreets Dataset

To generate the dataset we have used in our experiments, we have started from the pre-trained models of Vid2vid and Wc-vid2vid on the Cityscapes dataset [94]. Both models require a sequence of semantic segmentation masks as input to generate a video. Additionally, Wc-vid2vid requires a sequence of guidance images to improve the output video's temporal consistency [88]. The models' output is a photorealistic video of street scenes that renders the content of the input sequence of the semantic segmentation masks. Figure 3.2 shows an example of the generated videos.

We have used Cityscapes [94] and Kitti [95] datasets to produce the input segmentation maps. The cityscapes dataset consists of diverse urban street videos from several German cities. The videos have 2048 × 1024 resolution captured using a pair of stereo cameras at varying times of the year. The Kitti dataset contains driving videos that have been captured near Karlsruhe, Germany, using colour and grayscale camera images. In our experiments, we have used only the colour videos with 1392 × 512 resolution.

Since not all the images of Cityscapes are labelled with segmentation masks, and to be consistent with the Vid2vid pipeline, we have annotated the images of Cityscapes and Kitti by using the network from Zhuet *et al.* [96]. We have also followed the Wc-vid2vid pipeline to generate the guidance images, and we have performed *structure from motion* (SfM) on the video sequences using OpenSfM [97] to obtain motion fields for the video sequences.

We have divided the dataset into three subsets; cityvid, Citywcvid and Kittivid. We have generated Cityvid by inputting the segmentation masks of Cityscapes into Vid2vid, and Citywcvid by feeding Wc-vid2vid with segmentation masks and guidance images of Cityscapes. For Kittivid, we have used the segmentation masks of the Kitti dataset as input to Vid2vid. Each sub-dataset contains 400 videos, half of them are fake, and the other half are real (the real videos are the original sequences from Cityscapes and Kitti datasets). The videos resolution is 512 × 1024 and their duration is 3 seconds, with 30 frames. Table 3.1 summarises the characteristics of the sub-datasets.

sub-dataset	network	segmentation map
Cityvid	vid2vid	cityscapes
Citywcvid	wc-vid2vid	cityscapes
Kittivid	vid2vid	kitti

Table 3.1: A summary of DeepStreets dataset.

In addition to the raw videos produced by the Vid2vid models, we have compressed the videos with two quality levels; *HQ*, and *LQ*. In particular, we have compressed the raw videos by using the H.264 codec with a constant rate quantisation parameter equal to 23 and 40, respectively. The compressed videos as well as the raw videos are available on our website; `http://clem.dii.unisi.it/~vipp/datasets.html`.

3.2.2 The Developed Detector

To distinguish genuine videos from DeepStreets ones, we have used a data-driven forgery detector based on CNN. Since the constructed dataset has no clear predefined traces and it is difficult to rely on handcrafted features to differentiate between fake and genuine videos. In addition, the existing deepfake detectors usually detect and crop the faces in each frame of the videos. In our case, we need to input the whole frame to the detector, thus making a CNN-like architecture a good candidate for our task. Having said that, we have used the Xception network [1] as the backbone for our detector.

Xception is a CNN-like architecture based on depthwise separable convolution layers.

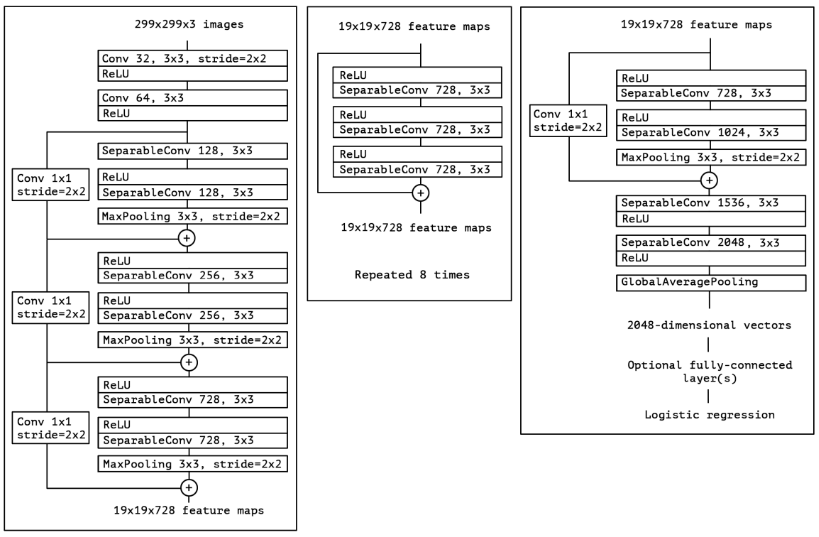


Figure 3.3: Xception network architecture [1].

It uses 36 convolutional layers for the feature extraction part. The layers are organised into 14 modules; each module has linear residual connections, excluding the first and last modules. The convolutional part is followed by an optional fully connected layer and a logistic regression layer. For our detector, we have added a fully connected layer with two outputs on top of the feature extraction part. The architecture of Xception is depicted in Figure 3.3.

We have formalised the detection task as a *per-frame binary classification* problem. We started by extracting the frames from each video. Then, to limit the computational burden, we resized each frame to 256×512 resolution. Afterwards, the frames are sequentially fed to the Xception network. The detection pipeline is shown in figure 3.4.

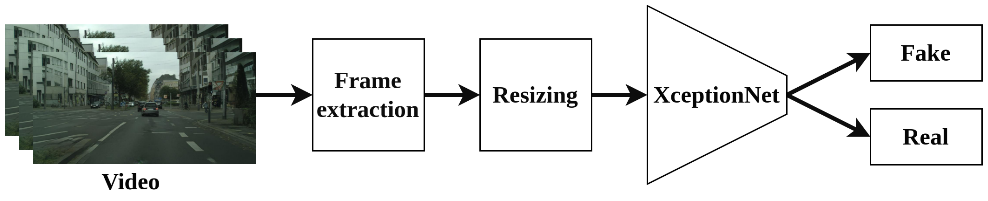


Figure 3.4: Detection pipeline.

In the following, we describe the setting that we have used to conduct the experiments. We have split the dataset into training, testing, and validation sets, with ratios of 60%, 25%, and 15%, respectively. All the results reported hereafter regard the detection accuracy of the test set. During the training, we used Adam optimizer with learning rate $(\alpha) = 0.0001$ and the default values for the first and second-order moments $(\beta_1 = 0.9, \beta_2 = 0.999)$, and an epsilon value $(\epsilon) = 10^{-8}$. Finally, we set the batch size to 8. The training was stopped whenever validation loss did not decrease for 10 consecutive epochs. All the experiments were performed using PyTorch [98] framework on a workstation with one Intel core i9 and four NVIDIA GeForce RTX 2080 Ti GPUs.

3.3 Experimental Results

To assess the performance of the developed forgery detector, we have conducted several experiments on the DeepStreets dataset at various compression levels, including matched and mismatched conditions. Furthermore, we have investigated its generalization capability by carrying out cross-dataset analysis between all subsets of the DeepStreets dataset.

Dataset\Quality	RAW	HQ	LQ
Cityvid	100.00	100.00	97.93
Citywcvid	99.76	99.62	99.96
Kittivid	100.00	100.00	100.00
DeepStreets	99.86	100.00	99.19

Table 3.2: Detection accuracy (the detector is asked to distinguish the original sequences used to create the segmentation masks and the synthetic videos).

In our first experiments, we have trained the detector to distinguish the original sequences used to generate the segmentation masks and the sequences produced by the synthetic video generators. Table 3.2 shows the detection accuracy of the detector on all subsets of the DeepStreets dataset. The results refer to matched training and testing, that is when training and testing are carried out on the same subsets. As it can be seen, the classification accuracy is almost perfect for all the subsets even when training and testing are carried out on the whole dataset. With regard to the impact of video compression on detection accuracy, we have carried out two sets of experiments. In Table 3.2, the detector is trained and tested in matched compression conditions, while in Table 3.3 different training and testing conditions are used. In both cases, compression has a minor impact on the accuracy. This marks a difference with respect to deepfakes videos, where the presence of compression leads to a significant performance drop (see for instance the results reported in [44] with regard to low-quality videos).

Training\Testing	RAW	HQ	LQ
RAW	99.89	99.90	95.41
HQ	100.00	100.00	95.72
LQ	99.70	99.63	99.19

Table 3.3: Compression mismatch experiments. The detector has been trained (and tested) on the entire DeepStreet dataset.

To investigate the generalisation capability of the developed forgery detector, we have performed a cross-dataset analysis between all the subsets of the DeepStreets dataset. As illustrated in Table 3.4, in some cases, dataset mismatch causes a dramatic drop in the performance, while in other cases the performance loss is less significant. Such behaviour can be explained by the different production pipelines used for the three sub-datasets. As mentioned in section 3.2.1, we have generated each subset by adopting different settings, including the source of the semantic maps and the specific Vid2vid architecture that generated the videos.

Training\Testing	Cityvid	Citywcvid	Kittivid
Cityvid	100.00	71.50	88.16
Citywcvid	98.76	99.76	50.00
Kittivid	50.03	50.00	100.00

Table 3.4: Cross-dataset analysis between subsets of DeepStreets dataset (RAW videos).

For instance, we can see that performance significantly deteriorates if we perform cross-dataset analysis between Kittivid and Citywcvid sub-datasets since these sub-datasets have a different source for the semantic sequences. We have used the Kitti [95] and the Cityscapes [94] datasets to generate Kittivid and Citywcvid sub-datasets, respectively. Besides, the Vid2vid architectures used to generate them are different (see table 3.1). In contrast, the performance is quite adequate when we perform cross-dataset analysis between Cityvid and Citywcvid sub-datasets since both of them have the same source of semantic sequences (Cityscapes [94] in this case) and differ only for the Vid2vid architecture used for the generation.

We have also conducted a second set of experiments, wherein the detector is asked to distinguish the synthetic sequences from the real sequences used to train the generators (namely the sequences from real images of the Cityvid sub-dataset). The reason why we have carried out this additional experiment is that the generators tend to produce scenes with colours and light conditions similar to those of the sequences used to train them (that is Cityvid real images), so it may be relatively easy for the discriminator to distinguish original images belonging to a different dataset, e.g. Kitti sequences, since they are characterised by different environmental conditions (light and sun conditions, for instance). This is not the case when the real images are taken from the same dataset used to train the GANs. In the new setting, the real videos are driving scenes from the Cityscapes dataset, and the fake videos are synthesised videos generated by using the semantic sequences from the Kitti dataset. The detector trained in this way achieved 100% accuracy on the test set of the new sub-dataset. We have also tested the ability of the detector to distinguish between real Citivyd videos and fake videos generated starting from segmentation maps of Citivyd (the Cityvid sub-datasets in Table 3.1). Interestingly, the detector achieved an 80.80% accuracy, showing a certain capability to detect examples of fake videos that are not represented in the training set.

3.4 Conclusions

In this chapter, we have introduced a novel class of non-facial fake videos generated by video-to-video translation tools. We have demonstrated the possibility of detecting this types of fake video using data-driven methods, even when training and testing are carried out in compression mismatched conditions. We have chosen the street views as an example of non-facial fake videos. Nevertheless, the generation and detection pipeline can be extended to other contents.

Our effort to detect non-facial fake videos has to address the generalization challenge, a fundamental obstacle in the automatic detection of deepfake videos. The task demands comprehensive training data for effective generalization. However, ensuring the algorithm's capability to detect non-facial fake videos that have not been encountered during training remains an unsolved ongoing challenge.

Considering the strengths of our research, its adaptability extends beyond the vid2vid model, making it applicable to various generative models like any GAN-based models, Autoencoders, or Diffusion Models. Our methodology involves generating a representative dataset for the generative model. Then, leveraging the generated dataset to train a detector that can potentially detect fake videos generated by the respective model. We have applied this research approach in the next chapter to study the detectability of human body reenactment fake videos. Nonetheless, the challenge persists in ensuring the detector's generalizability to models that have not been directly used for training, emphasizing the need for further research in this area.

Chapter 4

Detection of Human Body Reenactment Fake Videos

Υπάρχει μόνο ένα καλό, η γνώση, και ένα κακό, η άγνοια.
The Only Good is Knowledge, and the Only Evil is Ignorance.
Socrates

This chapter addresses another forensic task, namely, the detection of fake videos of human body reenactment. To this purpose, we consider videos generated following the "Everybody Dance Now" framework [8]. To accomplish our task, we have constructed and released a novel dataset of fake videos of this kind, referred to as the FakeDance dataset. Additionally, we have developed two forgery detectors to study the detectability of FakeDance kind of videos. The first one exploits spatial-temporal clues of a given video by means of hand-crafted descriptors, whereas the second detector is an end-to-end detector based on CNNs trained for our purpose. Both detectors have their peculiarities and strengths, showing good performances in different operative scenarios.

4.1 Motivation

In this chapter, our discussion on the detection of non-facial fake videos continues with a specific focus on the forensic analysis of yet another category of fake human motion transfer videos, hereinafter referred to as FakeDance videos [8]. These videos represent the body of a person (target person or subject), moving with the motion inferred from another person (source person or subject). Even though creating these videos with friends looks like just a fun and entertaining activity, we expect that in the near future they could be used in a harmful way [99]. For example, to alter crime scene videos or to create fake surveillance videos.

In the light of this, the contribution of the research described in this chapter is twofold: the generation of the first dataset of fake videos of human body reenactment to train and test forensic methodologies, and the proposal of two forensic detectors aiming at studying the detectability of FakeDance videos. Concerning data generation, we have generated four sub-datasets. In each of the sub-datasets, we have mitigated one or more of the limitations presented in *Everybody Dance Now* framework [8]. We have generated 594 videos using different generation and compression settings. The dataset contains the fake videos together with the original videos used to train the *Everybody Dance Now* model. Regarding the detectability of FakeDance videos, we have developed two complementary detectors following two different approaches. One detector starts from a hand-crafted feature extraction process, limiting the data-driven part to the final classifier, while the other detector is purely data-driven and consists of a CNNs trained ad-hoc for this task.

To evaluate the generalization capability of the detectors and their ability to work in non-ideal conditions, we have considered both raw sequences and videos compressed at different quality levels. Additionally, we have trained the networks in data scarcity conditions, with only a few training examples available. The experiments prove the validity of the detectors we have developed in different settings. Results exhibited a very good accuracy, demonstrating that fake videos of the FakeDance dataset can be reliably distinguished from real videos.

The rest of the chapter is organized as follows. In Section 4.2, we present the FakeDance dataset. In Section 4.3, we describe the detectors we have developed. The results of the experiments we have carried out are reported and discussed in Section 4.4. Finally, we conclude the chapter with some remarks in Section 4.5.

4.2 Human Body Reenactment Video Dataset

To properly develop and test forensic detectors for fake human body reenacted videos, a well-defined dataset is needed. For this reason, we have created a new dataset, namely the FakeDance dataset, including original and synthetic videos in which the scene presents an in-the-wild single dancer filmed by a static camera. In particular, the synthetic videos depict human subjects whose dance motion has been automatically transferred from a source subject. We have generated the synthetic videos by leveraging the video-synthesis algorithm introduced by Chan et al. [8], hereafter referred to as the *Everybody Dance Now* generation model. Given a source video with a person dancing, *Everybody Dance Now* can generate a new synthetic video in which the source motion is transferred to a target person. The video synthesis model is based on GANs, in particular, on the image-to-image translation pix2pixHD model introduced by Wang et al. [92]. The model consists of a generator and a discriminator which are trained simultaneously and drive each other to improve. The generator learns to synthesize single frames of a person given a pose stick figure, while the discriminator learns the differences between the generated frames and the ground truth ones. With respect to the original pix2pixHD, the image generator of [8] is modified to enforce temporal coherence between adjacent frames. Instead of generating individual frames, the modified generator predicts two consecutive frames. The output frame is conditioned on its corresponding pose stick figure and the previously generated frame. At the same time, the discriminator is asked to determine the difference in realism and temporal coherence between the real and generated frames. Moreover, a specialized Generative Adversarial Network (GAN) is exploited to increase the realism of the generated facial region.

In the following, we provide additional details about the necessary steps to synthesize a video starting from a source video and a target one. Then, we describe the FakeDance dataset, providing information about the videos included and their characteristics.

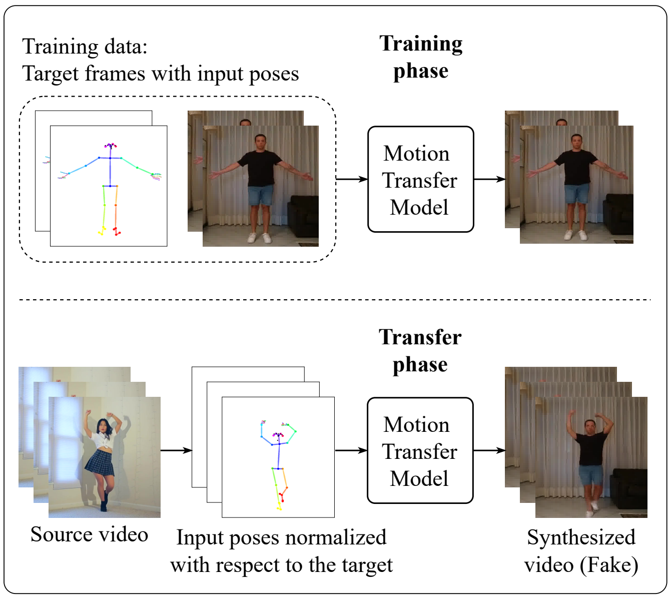


Figure 4.1: Video synthesis pipeline. In the training phase, a motion transfer model learns to transfer the motion to the target subject from pose stick figures and video frames. In the transfer phase, the motion is transferred from the source to the target subject by feeding the model with normalized pose stick figures of the source person.

4.2.1 Video synthesis pipeline

As depicted in Figure 4.1, the video synthesis pipeline proposed in [8] consists of two main phases: a *training phase* and a *transfer phase*. Given a target person, in the training phase, a model has to be trained that learns to transfer the motion from a pose stick figure, obtained through a pose detection step, to the target person. In the transfer phase, given a source video and a target video, the motion is transferred from the source subject to the target subject. The steps required for synthesizing new videos are the following:

Pose detection. The body poses of both the source and target person must be extracted in order to generate the synthesized video. As suggested in [8], we have exploited the OpenPose [100] detector to accurately estimate the 2D coordinates of the body skeletons. Given these coordinates, we can create a pose stick figure for each video frame.

Model training. Given a target video, we have to train a motion transfer model related to the target subject, providing as input the pose stick figures extracted from the video frames, together with the target video frames, that are used as a reference by ix2pixHD [101]. The trained model learns to map a pose into a video frame that looks like a frame from the target subject's video.

Pose normalization. Different humans might have different body shapes with different body parts proportions. Additionally, the distance between the video camera and the subject might differ from video to video. To accurately transfer the motion from the source to the target subject, we should take these considerations into account. Therefore, it might be necessary to modify the skeleton points of the source subject, so that they are "consistent" with the target subject. As suggested in [8], we have applied a linear transformation to all pose key points of the source subject for every video frame.

Reenactment of body motion. Once the source pose is normalized as detailed above, we can feed the normalized pose to the trained model. The network output consists of a sequence of synthesized video frames, in which the motion of the source subject is transferred to the target person. We have generated the final synthesized video by converting the synthesized video frames into an H.264 video sequence through FFmpeg [102].

4.2.2 FakeDance dataset

The FakeDance dataset is composed of four sub-datasets having different characteristics, as described in the following.

Swap_dance: This sub-dataset has a total of 20 videos, 10 real and 10 fake ones. The real videos depict 10 different dancers and include 5 original video sequences provided by [8], 2 videos selected from YouTube and 3 videos that have been filmed by ourselves, all longer than 6 minutes. The synthetic videos have been generated by following the steps reported in Section 4.2.1.

We have considered 10 target subjects, corresponding to the persons appearing in the real videos. Concerning the source subjects, we have collected 5 additional video sequences from YouTube (average length $\sim$ 2 minutes) depicting different dancing persons from the target subjects. We have transferred the motion of each source subject to the target videos, ending up with 50 fake videos. To obtain an equal number of real and fake videos for every target, we have selected only the synthetic video with the highest visual quality out of the 5 available ones. As the fake videos present significantly shorter lengths than the original target videos, we have cropped the original videos to the same temporal duration as the corresponding fake versions.

Puppet_dance: In this sub-dataset, the source and the target subjects used to synthesize the videos are the same. To this purpose, we have exploited the 5 original sequences provided by [8]. For every original video, we have generated one fake video with the same subject and the same motion.

To get the synthesized videos, we have temporarily cropped each original video by separating the first 70% of its frames from the remaining 30%. We have exploited the 70% of the frames (with the corresponding skeleton poses) for training the transfer model. Subsequently, during the transfer phase, the other 30% is employed as the source into which we have transferred the motion. In doing so, we end up with 5 fake videos with lengths equal to 30% that of the original sequences. To obtain a balanced dataset, in which the temporal duration of real and fake videos is the same, we have included in the "real" class only a 30% portion of the original sequences (the same used for the transfer phase).

The *Puppet_dance* sub-dataset simulates a potentially worrying scenario, in which the original motion of a target person is maliciously modified by exploiting other sequences depicting the same subject while she/he is moving differently. The threat is represented by out-of-context new motions transferred to the target video.

It is worth noticing that, even if the target and the source subjects are the same, *Puppet_dance* considers different input skeleton poses in the training and transfer phases. Thus, the transfer model does not completely overfit to the input data seen during the transfer phase.

Differently from *Swap_dance*, *Puppet_dance* does not need the pose normalization step to generate synthetic videos, as the source and target subjects are the same. As a consequence, no normalization errors were introduced and the synthesized videos have better visual quality.

Fit_dance: As we did for *Puppet_dance*, the *Fit_dance* sub-dataset also considers pairs of the same source and target subjects to synthesize new fake videos. However, contrarily to *Puppet_dance*, we have used *all* the frames of the original videos to train the transfer model for each subject. In the transfer phase, we fed the generative model with the same data already seen during training. Notice that the *Fit_dance* sub-dataset includes synthesized videos with high visual quality, reasonably better with respect to the fake videos in *Swap_dance* and *Puppet_dance*. Indeed, the transfer model is completely overfitted to the input data seen during the transfer phase. Furthermore, in this scenario, the pose normalization step is unnecessary, preventing the introduction of additional errors in video synthesis.

The original data used for generating *Fit_dance* subset consists of 74 video sequences downloaded from YouTube, depicting different dancing subjects.

Splice_dance: This sub-dataset includes 20 videos, 10 real and 10 fake, randomly selected from the test set portion of the *Fit_dance* sub-dataset, in which we have modified the background of the fake videos. More specifically, for each fake video frame, we have selected the pixel area of the synthesized human subject and pasted it into the background scene of the corresponding original video. The pixel region around the subject is selected by dilating its corresponding pose stick figure. The *Splice_dance* dataset enables us to investigate the challenging scenario in which the video under analysis might not be completely synthetic, but could have been doctored with the insertion of a small synthetic portion, e.g., a human subject which was not depicted in the original scene.

All the video sequences composing the four sub-datasets present common video codec parameters and pixel dimensions. Since we have generated H.264 fake sequences from the synthesized video frames, we re-encoded all the original videos using FFmpeg [102] with the same parameters used for the fake videos. We end up with a dataset including video sequences with a common resolution of $1024 \times 512 \times 3$ with a frame rate of 25 frames per second. Whenever the original videos have different dimensions, we have processed them to satisfy these conditions.

The four sub-datasets are always balanced for what concerns the temporal duration of the real and fake videos, i.e., any pairs of real-fake sequences present the same number of video frames. A summary of the sub-datasets of the FakeDance dataset is provided in Table 4.1, while Figure 4.2 shows a few selected video frames of the dataset. The

Table 4.1: Summary of the four sub-datasets composing the FakeDance dataset. All the videos of the dataset have a common resolution of $1024 \times 512 \times 3$ pixels. In all the scenarios, every real video and its corresponding fake version have the same temporal length.

	Swap_dance	*Puppet_dance*	*Fit_dance*	*Splice_dance*
Video subjects	Source $\neq$ Target	Source = Target	Source = Target	Source = Target
Video motion	Real $\neq$ Fake	Real = Fake	Real = Fake	Real = Fake
Pose normalization	Required	Not required	Not required	Not required
GAN input pose	Training phase $\neq$ Transfer phase	Training phase $\neq$ Transfer phase	Training phase = Transfer phase	Training phase = Transfer phase
Fake videos background	Synthetic	Synthetic	Synthetic	Original
No. real videos	10	5	74	10
No. fake videos	10	5	74	10
Avg. Duration [min:sec]	02:31	02:21	02:32	03:22

FakeDance dataset is publicly available and can be found at `https://bit.ly/3J2IENp`.

It is worth stressing that building the FakeDance dataset has required a noticeable computational power. Indeed, a different model has to be trained for each target person. Different training times have been observed to be necessary to reach a good visual quality of the synthesized videos, also depending on the video length. As an example, it took 3 days to train a model to generate a 7-minute video for a target person using a 12GB Quadro M6000 GPU, while for our longest input video - that is 20 minutes long - it took over 2 weeks to train a model for a single target person using a 24GB Quadro P6000 GPU.

4.2.3 Fake video quality assessment

In this section, we assess the quality of the fake videos by computing the similarity between the generated videos and their corresponding real counterpart. In particular, by referring to Figure 4.2, we notice that we can evaluate the similarity between real and fake content for *Fit_dance*, *Puppet_dance*, and *Splice_dance* sub-datasets, where the motion of the subject in the real and fake sequences is matched. In the case of *Swap_dance* sub-dataset, the real and fake subjects perform completely different movements, thus it is more challenging to evaluate the quality of the synthesized videos.

With regard to *Fit_dance* and *Puppet_dance* sub-datasets, which consist of fully generated sequences, we have computed the Structural Similarity Index Measure (SSIM) [103] between the real and fake video frames which exhibit the same motion. Even if SSIM

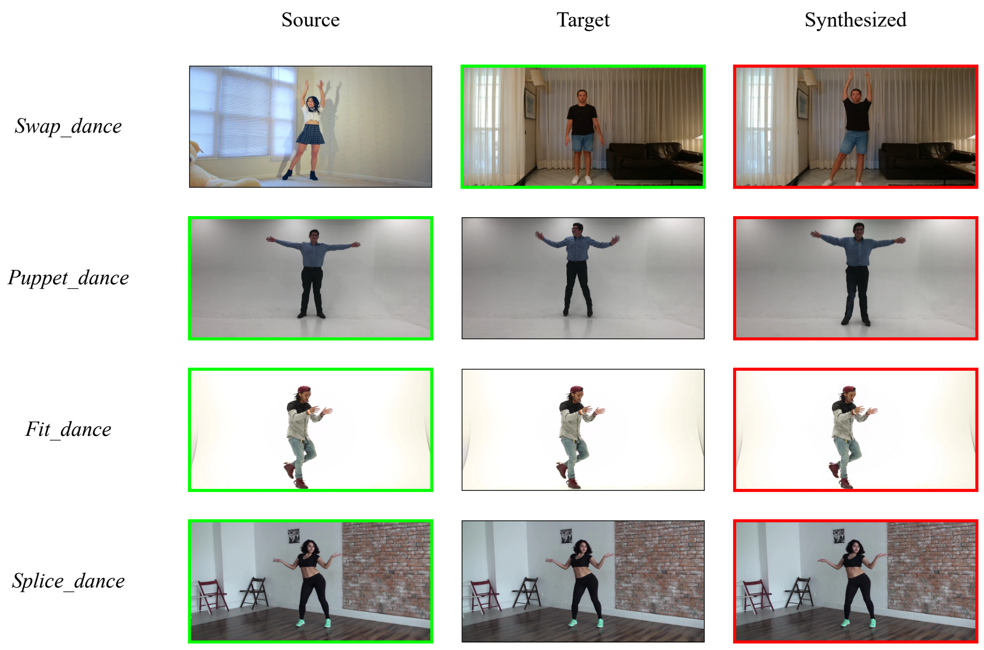


Figure 4.2: Video frames selected from the FakeDance dataset. In green and red, the frames are related to the "real" class and the "fake" class, respectively.

has been designed to assess the quality of still images, it is commonly utilized also to assess the quality of generated deepfake videos (even if, in this way, the quality of motion can not be evaluated properly). The SSIM score ranges from 0 to 1, where higher values indicate better quality.

In the case of the *Splice_dance* sub-dataset, given that videos are only partially synthetic (i.e., the background is real), we have computed a modified SSIM version inspired by the Mask-SSIM metric proposed in [46]. This metric was originally developed for facial deepfake evaluation. To evaluate the quality of the generated video frames, a mask is drawn to select the facial area and the SSIM is computed between the real and fake facial pixels. We have adopted this metric to work on *Splice_dance* videos, considering a mask that matches the subject motion in every video frame. Then, we computed the SSIM only between real and fake video frames by confining the computation to the pixels in the mask region. In doing so, we avoided being biased by the background pixels, which are exactly the same between real and fake sequences.

Applying SSIM to the Swap dance dataset is not feasible due to the absence of real frames for comparison with the generated fake frames. Consequently, we have excluded this sub-dataset from the experiments.

In all the considered scenarios, we have calculated the similarity of all the frames of the video. Then, we averaged the values obtained on each frame pair to obtain a global similarity value. The average similarity values obtained for *Fit_dance*, *Puppet_dance* and

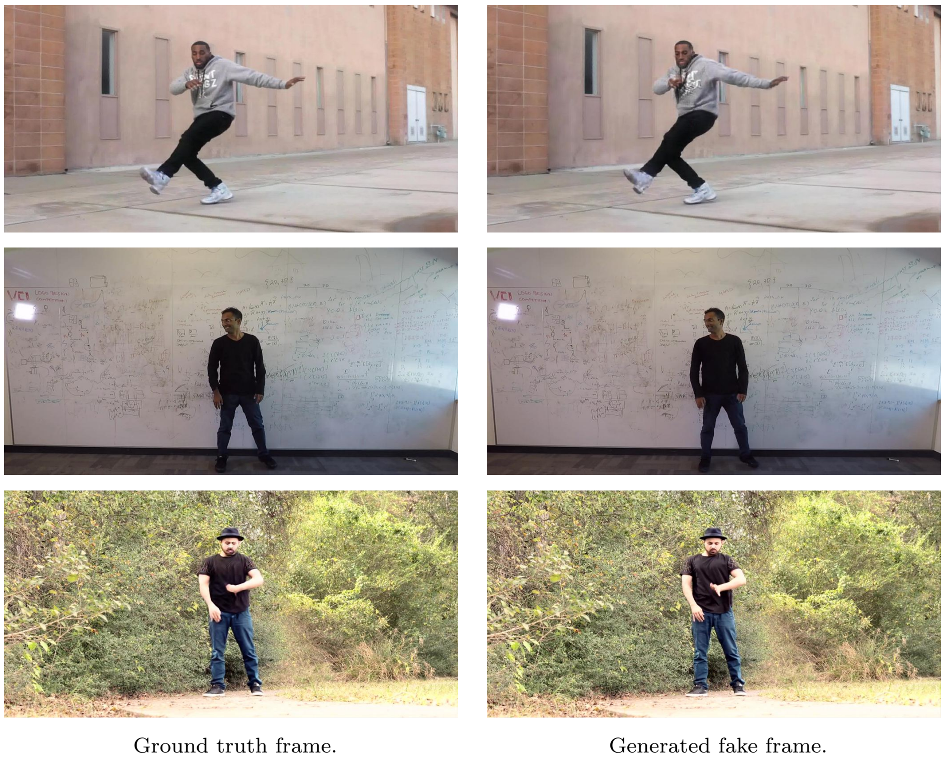


Figure 4.3: Example of frames utilized in the video quality assessment experiments. The first row displays frames from *Fit_dance*, the second row exhibits *Puppet_dance* frames, and the third row presents frames from the *Splice_dance* sub-datasets.

Splice_dance are reported in Table 4.2.

Table 4.2: Average similarity scores of different FakeDance sub-datasets. In the case of *Fit_dance* and *Puppet_dance*, we have computed the full-frame SSIM, while for *Splice_dance*, we have computed the Mask-SSIM, considering only the subject area.

Sub-dataset	*Puppet_dance*	*Fit_dance*	*Splice_dance* (Mask-SSIM)
SSIM	0.80	0.92	0.93

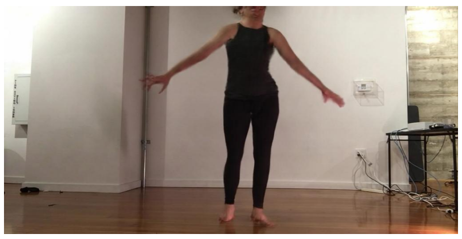
The results we have obtained reflect the differences in the production pipeline of each sub-dataset. Specifically, *Fit_dance* and *Splice_dance* videos have superior quality with respect to *Puppet_dance* videos. We attribute this fact to the consistency between the input pose during the training phase and the transferred pose, that characterizes *Fit_dance* and *Splice_dance* videos. In contrast, in *Puppet_dance*, the input poses in the training and transfer phase differ, yielding lower quality results. Some examples of frames used in the video quality assessment experiments are shown in Figure 4.3.

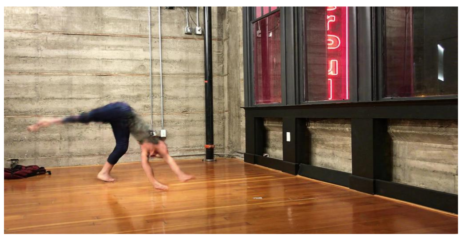
4.2.4 Limitations of the FakeDance dataset

Overall, we can highlight a few potential limitations of the FakeDance dataset. Indeed, the quality of the generated fake videos can be influenced by several factors. One of these is the background of the target video. If the target video lacks a static background, it may be challenging to maintain a consistent and high-quality background in the fake videos. Additionally, the Everybody Dance Now framework [8] may face difficulties in accurately portraying individuals wearing loose clothing or with intricate hairstyles. The current framework might struggle to effectively convey such details, potentially resulting in discrepancies or distortions in the appearance of clothing and hair in the generated fake videos.

Several other elements can influence the quality of the generated videos. Among them, we mention the variations in human body shapes and limitations within the pose normalization process, which might not account for diverse camera positions or extreme poses like handstands in both the source and target videos. Furthermore, the length of the target video is a crucial factor; longer target videos provide a broader spectrum of body movements to learn from, enabling *Everybody Dance Now* to generate movements that were not part of the training data. Such variations could lead to artefacts or inconsistencies in the generated fake videos, affecting the overall quality. Some examples highlighting the limitations of the FakeDance dataset are illustrated in Figure 4.4.

Intricate hairstyles.


Extreme camera position.


Handstand.

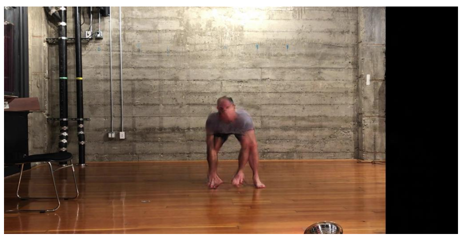

Extreme body pose.

Figure 4.4: Examples of frames showing the limitations of the FakeDance dataset.

4.3 Detection of fake dance videos

In this section, we introduce the two detectors that we have built to distinguish pristine from fake motion transfer videos. The first detector is based on handcrafted features and exploits the spatial-temporal information of the video sequences, while the second one is based on a CNNs architecture and works on a frame-by-frame basis. As we show in Section 4.4, both detectors work well, each of them having its strengths and weaknesses that make them suitable for different testing conditions.

4.3.1 Features-based detector

The first method we have developed is based on the extraction of handcrafted features known in the literature as Local Derivative Pattern on Three Orthogonal Planes (LDP-TOP) [104]. In a nutshell, given a query video sequence, the system runs two main steps: (i) pre-processing and body parts selection, in which the video is pre-processed in two different ways and five body parts are selected; (ii) feature extraction and classification, in which spatial-temporal features are extracted from the selected video and fed to a classifier. Then, it fuses the contributions of the pre-processing and aggregates the predictions related to the extracted body parts to obtain a final classification score related to the entire video sequence.

Pre-processing and data selection

We pre-processed each video $\mathbf{V}$ in two different ways to capture as many visual cues as possible: (i) $\mathbf{V}^{(g)}$, which is the grayscale version of $\mathbf{V}$; (ii) $\mathbf{V}^{(o)}$, which is the gradient of the optical flow extracted from $\mathbf{V}$ with the *Flownet2* architecture [105]. The information contained in $\mathbf{V}^{(o)}$ enhances the edges of the movement of the subjects. We call the generic pre-processed video as $\mathbf{V}^{(\rho)}$, where $\rho \in [g, o]$ indicates if we are considering $\mathbf{V}^{(g)}$ or $\mathbf{V}^{(o)}$.

We select frame-by-frame specific areas of each video $\mathbf{V}^{(\rho)}$ corresponding to relevant body parts of the subjects. The idea is to analyze the texture of the parts that contain important details which are useful for the recognition of the person and might be more difficult to synthesize. The chosen body parts are $\mathcal{B} = \{FC, LH, RH, LF, RF\}$: the face (FC), the left hand (LH), the right hand (RH), the left foot (LF) and the right foot (RF). To select the body parts for each frame, we first estimate the skeleton coordinates of the subject by means of the *Openpose* detector [100]. Then, following the approach suggested in [8], we exploit the skeleton coordinates to estimate the position of the bounding box surrounding each body part. For every body part, we spatially crop each pre-processed video sequence around the selected body part, defining a new sequence $\mathbf{V}_b^{(\rho)}$, where $b \in \mathcal{B}$.

Feature extraction and classification

Once the body parts have been selected, for each of them and for both pre-processed video versions (i.e., for every sequence $\mathbf{V}_b^{(\rho)}$), we extract the LDP-TOP features.

The LDP-TOP features are the evolution of the Local Derivative Pattern (LDP) features [106] already proposed to analyse natural 2D images. To extract the LDP features

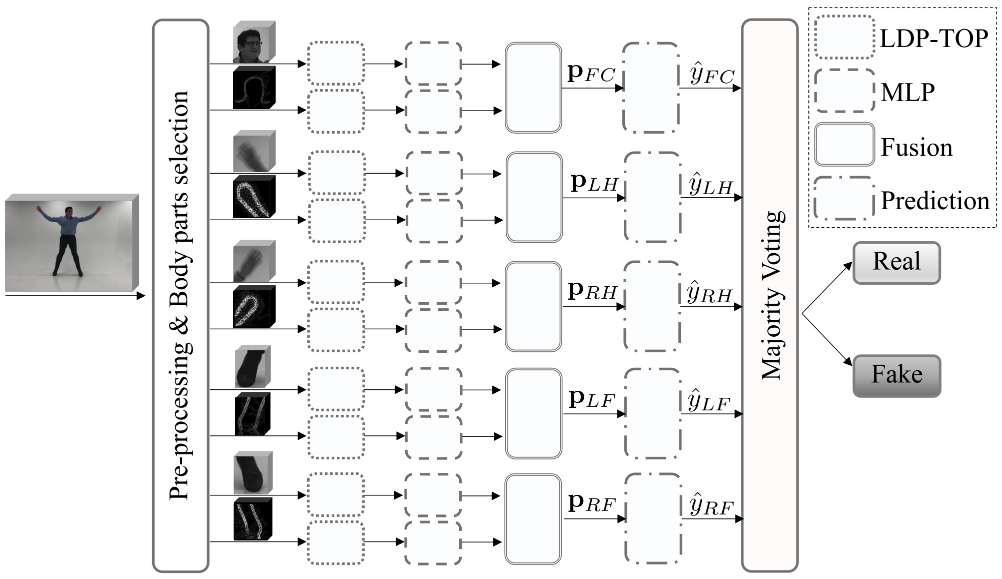


Figure 4.5: Feature-based classification pipeline. The video sequence under analysis is pre-processed in two ways extracting sequences corresponding to five different body parts. Then, we extract the LDP-TOP features from each sequence, later, we pass them through an MLP classifier and finally, we obtain the prediction through fusion and majority voting.

from a 2D image, we have to consider the 3×3 pixel neighbourhood surrounding each pixel. Then, we compute the difference between each pixel and its adjacent at $0°$ (i.e., the closest on the right) and replace its value with this difference. Then, we compare the magnitude of the values of the surrounding pixels and the central one, replacing these values with 0 if the central pixel is smaller, 1 otherwise. For each pixel, we obtain an 8-bit binary vector. The LDP feature extracted from the image corresponds to the histogram of the binary vectors obtained from all the pixels, consisting of 2^8 elements.

The LDP-TOP features exploit both of spatial and temporal domains in the analysis of grey-scale video sequences. Given a video with dimensions $H \times W \times T$ (H, W being the spatial dimensions and T the temporal one), we have to consider the three central 2D arrays along each dimension i.e., the 2D arrays with dimensions $H \times W$, $W \times T$ and $H \times T$ obtained by cutting temporally and spatially the entire video in $T/2$, $H/2$ and $W/2$, respectively. For every 2D array, we extract four LDP considering four adjacency angles in the calculation of pixel differences ($0°$, $45°$, $90°$, $135°$) and concatenate them. By concatenating the three resulting features of the 2D arrays, we obtain the LDP-TOP features of the video, consisting of $2^8 \times 4 \times 3 = 3072$ elements.

After the extraction of the LDP-TOP from each video $\mathbf{V}_b^{(\rho)}$, we apply a classifier that predicts the class of the query video $\mathbf{V}$ by combining all the contributions of the two pre-processing and the five body parts. The overall pipeline consists of five main steps (see Figure 4.5):

1. *Feature extraction*: Given a video sequence $\mathbf{V}_b^{(\rho)}$, we extract the LDP-TOP feature

vector defined as $\mathbf{L}_b^{(\rho)}$.

2. *Classification*: the features $\mathbf{L}_b^{(\rho)}$ are fed to one Multi-Layer Perceptron (MLP) classifier, in order to predict their class. The output vector $\mathbf{p}_b^{(\rho)} = [(\mathbf{p}_b^{(\rho)})_0, (\mathbf{p}_b^{(\rho)})_1]$ indicates the probability that the sequence $\mathbf{V}_b^{(\rho)}$ belongs to the classes "real" and "fake".

3. *Fusion*: For each $b \in \mathcal{B}$, we calculate the probability associated with the sequence $\mathbf{V}_b$ as the average between the probability calculated for the corresponding sequences $\mathbf{V}_b^{(g)}$ and $\mathbf{V}_b^{(o)}$:
$$\mathbf{p}_b = \frac{\mathbf{p}_b^{(g)} + \mathbf{p}_b^{(o)}}{2}. \tag{4.1}$$

4. *Prediction*: we compute the predicted label $\hat{y}_b$ of the sequence $\mathbf{V}_b$ as:
$$\hat{y}_b = \arg \max_{i \in [0,1]} (\mathbf{p}_b)_i. \tag{4.2}$$

5. *Majority voting*: the final video prediction $\hat{y}$ is computed by applying majority voting to the predictions $\hat{y}_b$ obtained on each body part:
$$\hat{y} = \text{maj}\Big(\{\hat{y}_{FC}, \hat{y}_{LH}, \hat{y}_{RH}, \hat{y}_{LF}, \hat{y}_{RF}\}\Big). \tag{4.3}$$
The operator maj$(\cdot)$ returns the most recurring value between the input predictions.

Furthermore, we also provide a probability score p associated with the synthetic class for the analyzed video sequence $\mathbf{V}$. This is computed by soft majority voting on the probability scores of each body part (see (4.1)). In practice, the score p is the arithmetic mean of the body part probabilities associated with the synthetic class.

4.3.2 CNN-based detector

The second solution we have developed follows a common approach in recent state-of-the-art [62, 63], and consists of a completely data-driven synthetic video detector based on CNNs working at the frame level. In particular, we have followed [62] where the best deepfake detection results have been obtained by using an EfficientNet-B3 architecture [107] pre-trained on Imagenet [108]. EfficientNet relies on a scaling method that uniformly scales the depth, width, and resolution of the network, adapting them to the input image, exploiting the intuition that if the image is bigger then the network needs more layers to get similar receptive fields and more channels to capture patterns having the same graininess. It has been shown that EfficientNet achieves better accuracy and efficiency and requires fewer parameters with respect to other CNNs [107].

The EfficientNet-B3 is fed with the frames extracted from the videos. During testing, given an input query video, we first extract the video frames. Subsequently, we aggregate the scores associated with the frames to achieve a final prediction score. The decision on the whole video (sequence of frames) is taken by applying soft majority voting, that is by averaging the frame predictions. The pipeline of the CNN-based detector is illustrated in Figure 4.6.

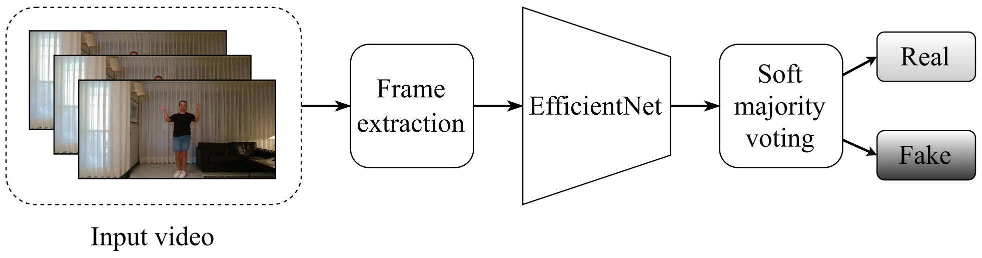


Figure 4.6: Detection pipeline of the CNN-based detector.

4.4 Experimental Analysis

In this section, we present the experimental campaign we carried out to evaluate the performance of the two detectors we have developed. We start by providing the details of the setup used to train the detectors, and afterwards we present the results we got and compare them with those achieved by some state-of-the-art methods.

4.4.1 Experimental setup

We have trained and tested the developed detectors exploiting the dataset split policy reported in Table 4.3, randomly splitting the available videos into different sets. All the sub-datasets are used both in the training and evaluation phases except for *Splice_dance*, which is employed only for testing. In fact, as the background of fake videos corresponds to the original one, the CNNs-based detector would reasonably fail if trained on this dataset. On the contrary, the feature-based detector, working only on the subject body parts, would likely report similar results to training on the *Fit_dance* sub-dataset.

Since *Swap_dance* and *Puppet_dance* have few examples and we do not have enough data for the validation process, we have performed validation only when we trained on *Fit_dance*, which is the largest sub-dataset. Even if we are aware of the risk of overfitting in case no validation set is used, in Section 4.4.2, we show that the detectors trained on *Swap_dance* and *Puppet_dance* demonstrate no signs of overfitting and provide a high test accuracy (aligned with that obtained during training).

Table 4.3: Number of videos in each sub-dataset. Every set contains an equal number of real and fake videos. However, video lengths vary, with a frame rate of 25. The average lengths of videos in each sub-dataset are reported in Table 4.2.2.

	Training	Validation	Testing
Swap_dance	16	0	4
Puppet_dance	8	0	2
Fit_dance	108	14	26
Splice_dance	0	0	20

Features-based detector (LDP-TOP extraction). As suggested in [104], we have extracted the LDP-TOP features considering video sequences of 50 consecutive frames, i.e., $T = 50$. In light of this, we have temporally cropped each video of the FakeDance dataset so that its temporal dimension amounts to 50 frames. To augment the dataset for the experimental evaluations, we have extracted all the possible sequences of 50 frames from each video, considering an overlap of 25 frames between adjacent sequences. In the following, we refer to each final video sequence as $\mathbf{V}$.

Features-based detector (MLP classification). We have used a straightforward MLP architecture, considering an input layer which is fed with the LDP-TOP features and returns an output vector of 100 elements. Then, we added three hidden layers, composed respectively of 100, 50 and 25 neurons. The output layer consists of 2 neurons, providing the probability vector $\mathbf{p}_b^{(\rho)}$ defined in Section 4.3.1. The activation function selected for each neuron is the Rectified Linear Unit (ReLU). As the number of selected body parts $|\mathcal{B}|$ is equal to five, we have trained 5 different classifiers for each of the 2 pre-processed videos, for a total of 10 MLP classifiers. For training, we have used cross-entropy loss and Adam optimizer with a constant learning rate equal to 10^{-3}. We have trained the MLP for at most 300 epochs, stopping training when the loss (the training loss in case validation is absent, the validation loss otherwise) does not decrease for 10 consecutive epochs. All the experiments have been performed on a workstation equipped with one Intel Core i7-6700K 8-Core CPU (4 GHz).

CNN-based detector. We have extracted frames from each video at a frame rate of 2 frames per second. Although we experimented with higher frame rates, we found that training required more time and resources without a corresponding increase in performance. All the frames are resized to a common size of 300×300 pixels before feeding them to the network. To be completely aligned with the feature-based detector setup and to fairly compare the two detectors, we have always considered sequences of 50 consecutive frames extracted as we described for the LDP-TOP setup.

The CNNs has been trained by using the cross-entropy loss and Adam optimizer with a constant learning rate of $2 \cdot 10^{-4}$. We stopped training when the loss did not decrease for 5 consecutive epochs. All the experiments have been performed using Pytorch [109] framework on a workstation equipped with one AMD Ryzen 9 3900X 12-Core CPU and one NVIDIA TITAN RTX 24GB.

4.4.2 Results

We have conducted several experiments to assess the performance of the developed forgery detectors on the FakeDance dataset. In particular, we have considered matched and mismatched dataset conditions, i.e., when the detectors are trained and tested on the same and different sub-datasets of the FakeDance dataset.

We have also considered compressed versions of the videos in addition to their RAW versions. Specifically, we have compressed the videos with two quality levels, using the H.264 codec with a constant rate parameter equal to 23 (HQ) and 40 (LQ), respectively. We have evaluated our method performance by means of two metrics: the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC), which is computed from the continuous scores returned by the two detectors, and the Balanced Accuracy (ACC) of the binary classification problem.

Performance under matched compression. Tables 4.4-4.6 show the ACC and AUC of the developed detectors trained and tested on the first three main subsets of FakeDance, namely, *Fit_dance*, *Puppet_dance* and *Swap_dance*. Table 4.4 reports the results for the RAW videos, Table 4.5 for the HQ videos and Table 4.6 for the LQ videos, when training and testing are done in matched compression conditions, that is, for the same compression quality of the videos. For each table, the results in the case of matched and mismatched datasets are reported.

Table 4.4: Performance of the detectors on the FakeDance dataset in cross-dataset scenarios (training and testing are carried out on RAW videos).

	LDP-TOP						CNN					
Test	*Fit*		*Puppet*		*Swap*		*Fit*		*Puppet*		*Swap*	
Train	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC
Fit	1.00	1.00	0.87	0.97	0.97	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Puppet	0.93	0.98	0.87	1.00	0.97	1.00	1.00	1.00	1.00	1.00	1.00	1.00
Swap	0.89	0.96	0.72	0.98	0.99	1.00	1.00	1.00	1.00	1.00	1.00	1.00

Table 4.5: Performance of the detectors on FakeDance dataset in cross-dataset scenarios (training and testing are carried out on HQ videos).

	LDP-TOP						CNN					
Test	*Fit*		*Puppet*		*Swap*		*Fit*		*Puppet*		*Swap*	
Train	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC
Fit	0.94	0.99	0.78	0.93	0.80	0.92	1.00	1.00	1.00	1.00	1.00	1.00
Puppet	0.64	0.69	1.00	1.00	0.82	0.91	0.73	0.82	1.00	1.00	0.87	0.96
Swap	0.62	0.67	0.52	0.84	0.87	0.99	0.76	0.85	0.90	0.96	1.00	1.00

Table 4.6: Performance of the detectors on FakeDance dataset in cross-dataset scenarios (training and testing are carried out on LQ videos).

	LDP-TOP						CNN					
Test	*Fit*		*Puppet*		*Swap*		*Fit*		*Puppet*		*Swap*	
Train	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC
Fit	0.87	0.98	0.52	0.74	0.60	0.74	0.80	0.94	0.60	0.76	0.50	0.56
Puppet	0.55	0.59	1.00	1.00	0.71	0.87	0.52	0.53	0.50	1.00	0.81	0.89
Swap	0.52	0.52	0.51	0.74	0.86	0.94	0.50	0.51	0.80	0.88	0.75	1.00

When training and testing on RAW videos (see Table 4.4), the performance of both detectors is extremely good for all sub-datasets, with the CNNs-based approach getting the best accuracy in all the cross-dataset scenarios. Not surprisingly, the performance decreases if we analyze compressed videos. Nonetheless, if we train and test on the same dataset, the performance remains quite good even when the compression is strong (LQ). In this scenario (see Table 4.6), we always achieve an AUC greater than 0.94.

In cross-dataset scenarios, the *Fit_dance* sub-dataset is the one showing the best generalization to the other sub-datasets. For instance, when working with HQ videos (see Table 4.5), perfect classification can be achieved with the CNNs-based method and a small performance drop is observed for the LDP-TOP-based detector (AUC always above 0.92). This behaviour confirms the expectation that training the detector on the most difficult cases, i.e., on the high-quality videos (i.e., *Fit_dance* videos) for which the discrimination is reasonably harder, permits to generalize to the other categories of fake videos.

In general, the CNNs-based detector outperforms the one based on LDP-TOP features whenever the video quality remains high, i.e., for RAW and HQ videos. However, when working on LQ videos, the LDP-TOP-based detector achieves better or similar performance to the CNNs one, showing very good robustness to data compression.

Focusing on Table 4.6, we also observe that there are cases in which high AUC do not correspond to high ACC, and this is especially true for the CNNs detector. For instance, when training and testing on *Puppet_dance* with the CNNs-based method, we achieved perfect AUC but the ACC is only 50%. This result indicates that the real and fake score distributions are perfectly separated, but the separating threshold is different from 0. As argued in many works, e.g., [110], when testing conditions are very different with respect to those considered for training (and validation), using the same fixed threshold typically does not work and may cause a wrong decision, with a consequent drop in ACC. To realign the ACC to the AUC metric, threshold calibration needs to be performed on data that are representative of the testing conditions.

Performance under mismatched compression. In this scenario, we have trained and tested on videos that underwent compression with different compression levels. Motivated by the previous considerations on the generalization capability of *Fit_dance*, we have considered the detectors trained on this sub-dataset. Table 4.7 shows the results under mismatched compression conditions (the results in the matched compression case are not reported since they are the same as before). We notice that training on videos compressed with a certain quality level helps achieve a good level of robustness against similar or higher quality videos, that is, for a similar or weaker compression level. For instance, by training on HQ videos, we achieved excellent performance on RAW videos but poor on LQ ones; by training on LQ videos, we have achieved acceptable performance on RAW videos and only slightly worse on HQ videos. The ACC results are not very good. As observed before, this is due to the mismatch between training and testing conditions.

To address the threshold calibration issue, we have re-computed the threshold in such a way as to maximize the accuracy on the *Fit_dance* validation set, for the same quality of the compression. For example, when testing HQ images, we have considered the threshold set on the *Fit_dance* HQ validation set, that is, under matched compression conditions. From Table 4.8, we can see that the ACC increases in most of the cases where AUC and

Table 4.7: Detection performance in cross-dataset scenarios and mismatched compression conditions. Detectors are trained on *Fit_dance*.

		Train on RAW				Train on HQ				Train on LQ			
Test	Method	Test HQ		Test LQ		Test RAW		Test LQ		Test RAW		Test HQ	
		ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC
Fit	LDP-TOP	0.61	0.85	0.51	0.63	0.91	0.99	0.58	0.71	0.50	0.91	0.50	0.78
	CNN	0.88	0.99	0.53	0.73	1.00	1.00	0.65	0.71	0.61	0.88	0.50	0.77
Puppet	LDP-TOP	0.52	0.65	0.50	0.50	0.90	0.99	0.50	0.53	0.50	0.94	0.50	0.93
	CNN	0.70	1.00	0.50	0.68	1.00	1.00	0.60	0.80	0.60	0.96	0.60	0.68
Swap	LDP-TOP	0.52	0.69	0.50	0.40	0.96	0.99	0.50	0.40	0.50	0.91	0.50	0.73
	CNN	0.70	0.93	0.50	0.78	1.00	1.00	0.50	0.72	0.55	0.89	0.55	0.89

ACC were not aligned before.

To highlight the change, Table 4.9 reports the relative accuracy change between the results in Table 4.8 and Table 4.7.

When we train on RAW or HQ videos, a mismatch could remain between testing data and the data used for setting the threshold, i.e., when testing *Puppet_dance* and *Swap_dance*. Not surprisingly, then, in these scenarios, the ACC could improve further by calibrating the threshold on data taken from the same dataset. On the contrary, when training on LQ videos, the threshold calibration step always improves the results, with improvements above +15%.

Performance with locally manipulated videos. Results achieved on *Splice_dance* for various compression qualities are shown in Table 4.10 when training is performed on *Fit_dance* RAW videos.

In this scenario, the CNNs-based strategy achieves worse results with respect to the test performed on the *Fit_dance* videos (see Table 4.7, second row, train on RAW). We conjecture that the performance deterioration of the CNNs detector is due to the fact that, since this detector works full frame, it relies on artefacts introduced by the motion transfer generative model in both the foreground and the background. However, the background artifacts are not present in *Splice_dance*, due to the real background replacement. On the contrary, the performances of the LDP-TOP detector are similar to those achieved on *Fit_dance* (see Table 4.7, first row, train on RAW case). In fact, by looking only at the body parts and their movements, the LDP-TOP detector is by construction more robust against background manipulations and splicing.

Performance in data-scarcity conditions. We have investigated the performance of the detectors in data scarcity conditions, that is when only very few examples are available for training. These tests are particularly relevant since, in addition to relieving the burden of collecting large amounts of data, in some scenarios, it might be the case that a large number of samples representative of the task is not available. The results we got are reported in Table 4.11, for varying numbers of training videos selected from *Fit_dance* (RAW quality).

Table 4.8: Detection performance in cross-dataset scenarios and mismatched compression conditions. Detectors are trained on *Fit_dance*. We have set the threshold on *Fit_dance* validation set considering a compression quality matched with testing.

		Train on RAW				Train on HQ				Train on LQ			
		Test HQ		Test LQ		Test RAW		Test LQ		Test RAW		Test HQ	
Test	Method	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC
Fit	LDP-TOP	0.76	0.85	0.51	0.63	0.94	0.99	0.60	0.71	0.79	0.91	0.70	0.78
	CNN	0.92	0.99	0.65	0.73	0.92	1.00	0.65	0.71	0.76	0.88	0.73	0.77
Puppet	LDP-TOP	0.59	0.65	0.52	0.50	0.96	0.99	0.50	0.53	0.61	0.94	0.75	0.93
	CNN	0.90	1.00	0.50	0.68	0.90	1.00	0.70	0.80	0.70	0.96	0.70	0.68
Swap	LDP-TOP	0.63	0.69	0.50	0.40	0.93	0.99	0.43	0.40	0.78	0.91	0.67	0.73
	CNN	0.85	0.93	0.70	0.78	0.80	1.00	0.60	0.72	0.80	0.89	0.80	0.89

Table 4.9: Relative accuracy change between the results shown in Table 4.8 and those shown in Table 4.7. "-" refers to no changes.

		Train on RAW		Train on HQ		Train on LQ	
		Test HQ	Test LQ	Test RAW	Test LQ	Test RAW	Test HQ
Test	Method	ACC	ACC	ACC	ACC	ACC	ACC
Fit	LDP-TOP	+24%	–	+3.2%	+3.4%	+58%	+40%
	CNN	+4.5%	+22%	−8.0%	–	+24%	+46%
Puppet	LDP-TOP	+13%	+4.0%	+6.6%	–	+22%	+50%
	CNN	+28%	–	−10%	+16%	+16%	+16%
Swap	LDP-TOP	+21%	–	−3.1%	−14%	+56%	+34%
	CNN	+21%	+40%	−20%	+20%	+45%	+31%

Table 4.10: Performance of the detectors on *Splice_dance*, when training is performed on *Fit_dance* videos, RAW quality.

	Test RAW		Test HQ		Test LQ	
Method	ACC	AUC	ACC	AUC	ACC	AUC
LDP-TOP	1.00	1.00	0.60	0.90	0.50	0.58
CNN	0.70	1.00	0.60	0.72	0.55	0.65

As in the splicing case, also in this scenario, the LDP-TOP detector achieves superior performance with respect to the CNN detector which, being completely data-driven, requires a larger number of training videos to get good discrimination capabilities. In particular, for the LDP-TOP detector, 4 videos are already enough in most of the cases to get a balanced accuracy larger than 88%, while the CNN detector is not able at all to discriminate with so few videos. The capability to work in data-scarcity conditions is a noticeable strength of the LDP-TOP approach.

Table 4.11: Performance of the detectors in data scarcity conditions (training on RAW *Fit_dance* videos and testing on RAW videos of all four sub-datasets).

	Swap_dance				*Puppet_dance*				*Fit_dance*				*Splice_dance*			
No. Training	LDP-TOP		CNN		LDP-TOP		CNN		LDP-TOP		CNN		LDP-TOP		CNN	
videos	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC	ACC	AUC
2	0.63	0.96	0.55	0.65	0.63	0.90	0.40	0.60	0.94	0.99	0.73	0.69	0.88	0.94	0.55	0.58
4	0.89	1.00	0.55	0.66	0.71	0.96	0.70	0.80	0.98	1.00	0.61	0.73	0.95	1.00	0.55	0.57
8	0.85	0.98	0.80	0.91	0.94	0.98	0.80	0.96	0.98	1.00	0.96	0.99	1.00	1.00	0.50	0.68
16	0.90	0.99	0.90	1.00	0.87	0.99	1.00	1.00	0.99	1.00	1.00	1.00	1.00	1.00	0.60	0.79
32	0.96	1.00	1.00	1.00	0.89	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.76	0.80
54	0.97	1.00	1.00	1.00	0.87	0.97	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	0.70	1.00

4.4.3 Fusion of LDP-TOP and CNN detectors

As a final investigation, we have conducted an experiment involving the fusion of the two detectors. We have employed a straightforward fusion, considering the average of the output probabilities generated by the detectors.

The fusion results are presented in Tables 4.12, 4.13, and 4.14 for RAW videos, HQ videos and LQ videos, respectively, and compared with those achieved by the single techniques shown in Tables 4.4, 4.5, 4.6, i.e., in the matched compression scenario. In the columns labelled as "LDP-TOP" and "CNN", we highlight the relative accuracy change (defined as Δ-ACC) of the fusion process with respect to the single detector results shown in Tables 4.4, 4.5, and 4.6.

Table 4.12: Performance of the detectors after fusion on RAW videos. LDP-TOP and CNN refer to the relative accuracy change for both detectors after fusion, defined as Δ-ACC. The symbol "-" indicates no changes.

	LDP-TOP			CNN			Fusion		
Test / Train	*Fit* Δ-ACC	*Puppet* Δ-ACC	*Swap* Δ-ACC	*Fit* Δ-ACC	*Puppet* Δ-ACC	*Swap* Δ-ACC	*Fit* ACC	*Puppet* ACC	*Swap* ACC
Fit	–	+14.94%	+3.09%	–	–	–	1.00	1.00	1.00
Puppet	+7.52%	+14.94%	+3.09%	–	–	–	1.00	1.00	1.00
Swap	+12.36%	+38.89%	+1.01%	–	–	–	1.00	1.00	1.00

On average, the achieved results demonstrate an improvement in performance with respect to the single detectors, which in some cases is very relevant. This consideration is valid primarily for the LDP-TOP technique which, if considered as a single approach, indeed returned worse results than the CNN, especially on RAW and HQ videos (see Tables 4.4, 4.5). The CNN approach mainly benefits from the fusion strategy on LQ

Table 4.13: Performance of the detectors after fusion on HQ videos. LDP-TOP and CNN refer to the relative accuracy change for both detectors after fusion, defined as Δ-ACC. The symbol "-" indicates no changes.

	LDP-TOP			CNN			Fusion		
Test / Train	*Fit* Δ-ACC	*Puppet* Δ-ACC	*Swap* Δ-ACC	*Fit* Δ-ACC	*Puppet* Δ-ACC	*Swap* Δ-ACC	*Fit* ACC	*Puppet* ACC	*Swap* ACC
Fit	+6.38%	+28.20%	+25.00%	–	–	–	1.00	1.00	1.00
Puppet	+18.75%	–	−8.53%	+4.10%	–	−13.79%	0.76	1.00	0.75
Swap	+29.03%	+92.30%	+14.94%	+5.26%	+11.11%	–	0.80	1.00	1.00

Table 4.14: Performance of the detectors after fusion on LQ videos. LDP-TOP and CNN refer to the relative accuracy change for both detectors after fusion, defined as Δ-ACC. The symbol "-" indicates no changes.

	LDP-TOP			CNN			Fusion		
Test / Train	*Fit* Δ-ACC	*Puppet* Δ-ACC	*Swap* Δ-ACC	*Fit* Δ-ACC	*Puppet* Δ-ACC	*Swap* Δ-ACC	*Fit* ACC	*Puppet* ACC	*Swap* ACC
Fit	+10.34%	−3.84%	−16.66%	+20.00%	−16.66%	–	0.96	0.50	0.50
Puppet	−3.63%	–	+40.84%	+1.92%	+11.11%	+23.45%	0.53	1.00	1.00
Swap	−3.84%	+96.07%	+16.27%	–	+25.00%	+33.33%	0.50	1.00	1.00

video sequences (see Table 4.14). As a matter of fact, the LDP-TOP method in this case achieves comparable results to the CNN approach (see Table 4.6), therefore the fusion between the two methods reasonably helps to improve the CNN performances as well.

4.4.4 Comparisons with State-of-the-Art

To contextualize our detection methodologies in the wide literature of multimedia forensics, we have conducted a comparison with some relevant state-of-the-art deepfake detection methods.

Given the nature of our fake sequences which mainly consist of fully generated video frames (i.e., all sub-datasets except *Splice_dance* are fully synthetic), we have compared our detectors with state-of-the-art methods that deal with the detection of fully generated images [3, 111–113]. We had not retrained these techniques on our dataset, but we had utilized the trained models released by the authors. Notice that all these methods are designed for synthetic image detection (i.e., they are developed for natural photographs) rather than for video frame analysis. To assess their performance on our dataset, we have extracted one video frame per second and we have calculated the outcome for the entire video sequence by averaging the frames' results.

We have obtained an average ACC of 50% for all the compared techniques, demonstrating that none of the methods can work on our dataset, since they misclassify all

FakeDance videos as real videos, thus confirming the need for dedicated techniques for our task. Such results highlight a well-known challenge encountered with deepfake detectors, that is, the generalization to different semantic content and generation models. Another known issue is the poor generalization to deepfake video frames for methods trained on synthetic images only.

Notice that we could in principle compare our developed methodology also with state-of-the-art detectors developed for deepfake videos. However, the common forgery detection methods designed to work on "standard" deepfake facial videos (i.e., including human subjects whose facial characteristics have been manipulated) can not be directly applied to our case. The main reason for such a difficulty is that deepfake detectors strongly rely on the facial features of the videos, all the more that they usually crop the face region before inputting the video frames to the detectors. Furthermore, some detectors are based on the analysis of a set of geometric and semantic features that are directly related to human faces; like eye color [50] or facial pose [56]. No such traces can reasonably be exposed on FakeDance videos.

4.5 Conclusions

In this chapter, we have described our efforts to detect non-facial fake videos. In particular, we addressed the problem of the detection of human motion transfer videos. We have constructed and released the FakeDance dataset, with almost 600 fake and real videos, obtained by following different generation pipelines and different compression settings. We have also considered two detection approaches: a feature-based approach that relies on handcrafted features, namely the Local Derivative Pattern on Three Orthogonal Planes (LDP-TOP) features, extracted from the body parts of the human subject, and a completely data-driven method, based on CNNs trained directly on video frames.

We have carried out several experiments to test the effectiveness of the detectors and their generalization performance. Specifically, the performance of the detectors is assessed and discussed in the cross-dataset scenario, that is, when different subsets of the FakeDance dataset are considered for training and testing, for both matched and mismatched compression levels of the videos.

Working with high visual quality videos, the CNNs detector outperforms the LDP-TOP method in all the conditions. However, the detector based on LDP-TOP is more robust to strong data compression and achieves better results on low visual-quality videos. Furthermore, while the CNNs detector achieves superior performance when many video samples are available for training, the LDP-TOP method gets better performance in data-scarcity conditions when very few videos are available to the analyst. Finally, being focused on the analysis of body parts and their movements, the LDP-TOP detector achieves better results in the challenging splicing scenario, in which a synthesized human subject is copied and pasted over a real background.

Chapter 5

Detecting Deepfake Videos in Data Scarcity Conditions

بِالعِلْمِ يُدْرِكُ أَقصَى المَجْدِ مِنْ أَمَمٍ وَلا رُقِيَّ بِغَيْرِ العِلْمِ لِلأُمَمِ.

Through Knowledge, the Utmost Glory is Attained for Civilizations.

Kahlil Gibran

The most powerful deepfake detection methods developed so far are based on DL. As such, they require that large amounts of training data representative of the specific task are available to the trainer. In this chapter, we describe a feature-based method for video deepfake detection that can work in data scarcity conditions, that is, when only very few examples are available to the forensic analyst. The proposed method is based on video coding analysis and relies on a simple footprint obtained from the motion prediction modes in the video sequence. The footprint is extracted from video sequences and used to train a simple linear Support Vector Machine (SVM) classifier. The effectiveness of the developed method is validated experimentally on three different datasets, namely, a synthetic street video dataset and two datasets of Deepfake face videos.

5.1 Motivation

The goal of video deepfake detection is to distinguish AI-manipulated videos from real ones. The most powerful video deepfake detection techniques developed so far are based on DL [59], and require that a large amount of training data representative of the task at hand are available. To facilitate the progress of deepfake detection, many efforts have been made to generate deepfake datasets, for instance, the FaceForensics++ [44], DeeperForensics-1.0 [47], and VideoForensicsHQ [114] datasets. However, DL techniques trained on large datasets require that the data used for training matches the data analyzed when the model is deployed in operative conditions. The generalization capabilities of these techniques are in fact often poor, with performance dropping significantly on related - but unseen -

manipulations or when different DL architectures are used for the generation [28, 115]. In addition, in some scenarios, collecting such a large amount of data to train a DL model may simply be impossible. This is the case, for instance, of privacy-sensitive applications, like in the healthcare domain. In such situations, only very limited amounts of training data, collected under strict conditions, may be available. In other cases, the investigator may not have access to the model which has been used to generate the synthetic media, thus making it impossible to build a sample dataset which is large enough to train a DL architecture. Still, the investigator may rely on a few synthetic samples collected during his/her investigations [116]. Therefore, the availability of a tool that can be trained with limited available data would be of great help.

Starting from these considerations, in this chapter, we introduce a feature-based method for video deepfake detection that can work in data scarcity conditions, that is, when only very few examples of the same kind are available to the forensic analyst. The proposed method relies on the analysis of motion-related features, that have been already exploited in some works in order to detect deepfake videos by means of DL [68, 72, 73, 117]. However, instead of relying on DL architectures and intensive training procedures to extract discriminative features, we base the classification on a simple footprint that relies on the frequency of motion prediction modes in the video sequence. The footprint is inspired by the one used in [84] for the detection of re-encoded (double-encoded) videos. Based on our analysis of the behaviour of this footprint on real and synthetic videos, we argue that, for deepfake videos, the distribution of motion prediction modes is different with respect to real videos. Moreover, in many cases, deepfake videos tend to be less predictable than real videos, thus requiring the use of a large number of *Intra* predicted frames. The footprint extracted from video sequences is used to train an SVM in charge of discriminating between fake and real videos. Due to the simplicity of the adopted features, a simple linear SVM can reach high accuracy even if it is trained on very few video samples. The effectiveness of the developed method is experimentally demonstrated on three datasets: a dataset of fake street views, DeepStreets [3], corresponding to the case of fully synthetic videos generated by GANs, and two datasets of fake face videos, namely, DeeperForensics-1.0 [47] and VideoForensicsHQ [114], where the video contents are manipulated locally by means of autoencoders.

The rest of this chapter is organized as follows. In Section 5.2, we present the method that we have developed. Then, in Section 5.3, we describe the methodology that we have followed in our experiments. The results of the experiments we have carried out to validate the detector are described in Section 5.4. In Section 5.5, we conclude the chapter with some final remarks.

5.2 DeepFake Detection Based on Video Coding Features

Before presenting the method we have developed, we introduce the main features of the H.264 video coding standard [118], which is the most commonly used standard for encoding deepfake videos, by confining the discussion to the notions that are necessary to understand the forensic footprint. It is worth observing that these features are general ones, common to all macroblock-based video coding standards, e.g., MPEG-2, MPEG-4 or H.263, and also the modern H.265 (HEVC) standard.

5.2.1 Video coding (H.264) in a nutshell

During video encoding, each frame of the video stream is partitioned into macroblocks. In the following, we refer to the processing unit as a macroblock. The typical size of the macroblocks is 16 × 16 pixels. The prediction types are selected on a macroblock basis rather than being the same for the entire frame. The main types of macroblocks are: *Intra*-coded (I-type) macroblocks, and *Inter*-coded macroblocks, that can be forward predicted (P-type), and bi-directional predicted (B-type). I-type macroblocks are encoded by *Intra* prediction, with no reference to the other macroblocks, while P and B-type macroblocks are predicted via *Intra*-prediction. In the P-type, the previous frame is used for the prediction, while in the B-type case, the prediction is bi-directional, then both the previous and future frames are used. Therefore, for *Inter*-coded macroblocks, the motion vector pointing to the best matching macroblock in the reference frame is encoded, along with the prediction error (residuals), for motion compensation. A frame can belong to three different categories, defining the prediction types available to that frame: P frames, that contain I- or P-type macroblocks; B frames, that contain I-, P- or B-type macroblocks; I frames, that contain I-type macroblocks only. In early standards, motion compensation was performed with one motion vector per macroblock. In H.264/AVC, like in other modern standards, a macroblock can be split into multiple variable-sized prediction blocks (up to 4 × 4), called partition units, based on the content of the frame. Then, a separate motion vector is specified for each partition of the *Inter*-predicted macroblock.

To further improve the coding efficiency in P- and B-frames, the H.264 and H.265 standards resort to the *Skip* mode. A *Skip* macroblock consists of a single prediction unit whose motion data is derived from the neighbouring macroblocks. No residuals are transmitted in *Skip* mode.

In H264 and H265 video coding standards, the quality of the compressed video (or compression rate/level) is set by means of the Constant Rate Factor (CRF). The Raw format refers to the case CFR = 0. In High Quality (HQ) and Low Quality (LQ) video formats, the videos are compressed by the video codec with a CFR respectively of 23, and 40 [119].

5.2.2 Video coding features

As we mentioned, the common procedure to generate fake videos using generative models, like GANs or Autoencoders, introduces both spatial (pixel-level) as well as motion artefacts. By inspecting fake and real videos with similar contents, we can observe that the motion in fake videos tends to be less fluent and natural with respect to the case of real (original) videos. Therefore, we hypothesize that, in many cases, fake videos tend to be less predictable than real videos. We argue that such differences can be captured by video coding features. In particular, our conjecture is that the different behaviour can be reflected by looking at the prediction modes of the macroblocks. More specifically, we focus on the prediction modes in P- and B-frames and compute their frequency on a video (sub)sequence or batch of frames. For each frame, we extract the following footprint: $F = [f_{Intra}, f_{Inter}, f_{Skip}]$, where f_x denote the frequency/percentage of macroblocks of mode x in the frame. f_{Inter} accounts for both the P-type and the B-type macroblocks. Note that a similar feature has been used in [84] for the detection of low-quality H.264 video contents re-encoded in H.265 format with higher quality.

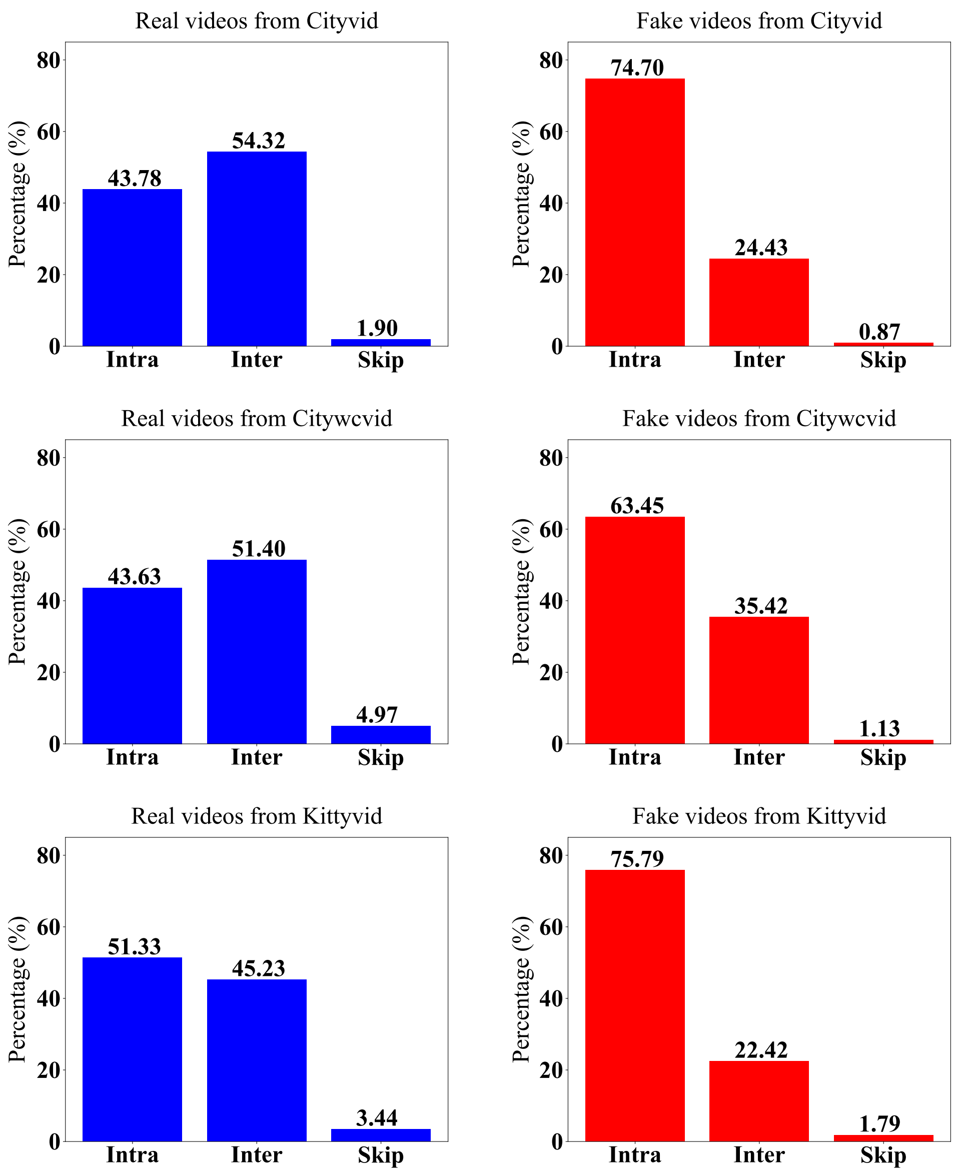


Figure 5.1: Frequency of *Intra*, *Inter* and *Skip* for raw videos in the DeepStreets dataset.

To verify our hypothesis, we have computed F for real and fake H.264 encoded video sequences. The comparison is carried out for the same video quality, e.g., for the same CRF. We have observed that, as argued, when the fake videos are fully GAN synthesized, the frequencies of the prediction modes are very different with respect to the case of real videos, and the distribution of the three features is very different. Figure 5.1 reports the distribution of f_{Intra}, f_{Inter}, and f_{Skip} averaged on all the frames of the raw videos for the three sets of the DeepStreets dataset [3], respectively for real (left) and fake (right) videos with similar content. In particular, by looking at the figure, we see that, for fake videos, *Intra*-coded macroblocks represent the large majority of the macroblocks, and their percentage is by far larger for fake (74.7%) than for real (43.8%) videos. This confirms our conjecture that fake videos are less predictable, requiring more *Intra*-coded macroblocks. Notably, the same behaviour is not observed when the fake videos are

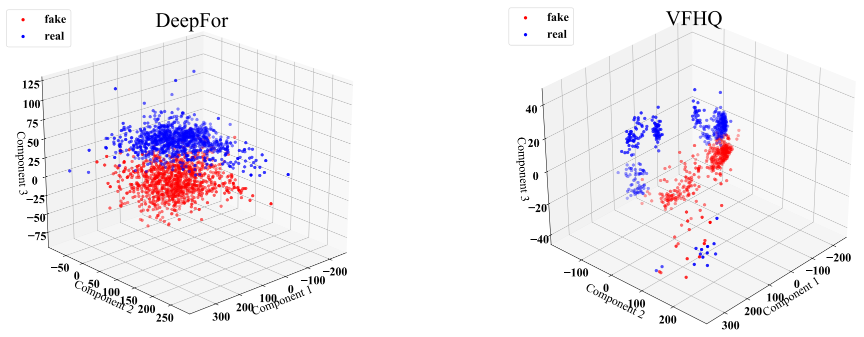


Figure 5.2: Feature distribution for 1,000 videos in the DeeperForensics-1.0 dataset and 394 videos in the VFHQ dataset.

generated by object swapping, in which case we verified that the percentage of *Intra*-coded macroblocks is similar for fake and real. Since those methods only modify a limited part of the frames via autoencoders, e.g., the faces or only the facial expressions, while the rest of the frame remains the same, deepfake videos generated in this way are not fully synthetic videos.[1] However, even in this case, discrimination is still possible based on the proposed feature, by looking at the joint behaviour of the three features, and in particular at their distribution over batches of N consecutive frames. Figure 5.2 shows the feature distribution after dimensionality reduction via Principal Component Analysis (PCA) for real and fake videos coming from the DeeperForensics-1.0 and VFHQ dataset. In order to apply PCA, for each video, the $(N \times 3)$-dimensional tensor with the N 3D footprints computed on the N frames, is reshaped to a vector of length $3N$. In the plot, N is set to 30. The reduced dimension is equal to 3.

Motivated by the above analysis, we have used the proposed feature to train an SVM

[1]Since the footprint relies on (spatially) global features, it is not surprising that the best discrimination is achieved in the fully synthetic case.

classifier to discriminate between real and deepfake videos under data-limited conditions. The details of SVM training are given in Section 5.3.2.

5.3 Experimental methodology

In this section, we describe the methodology that we followed in our experiments.

5.3.1 Datasets

To investigate the effectiveness of the developed detector, we have considered a dataset of fake street views (fully synthetic) and two datasets of Deepfake face videos, described in the following.

DeepStreets [3] is a GAN-synthesized street video dataset. The fake videos are generated from a sequence of semantic segmentation masks. The dataset consists of 3 subsets; Cityvid, Citywcvid and Kittyvid. The fake videos are generated using the Vid2vid [7] model (for Cityvid and Kittyvid) and the Wc-vid2vid [88] models (for Citywcvid), both trained on the Cityscapes dataset. Chapter 3 contains further details on the Deepstreets dataset.

DeeperForensics-1.0 (DeepFor) [47] is a large-scale face-swapping dataset consisting of 60,000 manipulated videos in total. For our tests, we have considered 1,000 raw manipulated videos and 1,000 corresponding real videos, that are taken from YouTube. The minimum video length is 6 seconds.

VideoForensicsHQ (VFHQ) [114] is a high-quality face reenactment dataset, where only the face is manipulated. Specifically, the facial expression of the source video is transferred to the target video. The dataset consists of 1,737 videos in total (1,141 real videos and 596 fake videos) representative of 8 different emotions. The video duration varies from 3 seconds to more than 5 minutes. To get balanced data, we have considered 596 real and fake videos from this dataset.

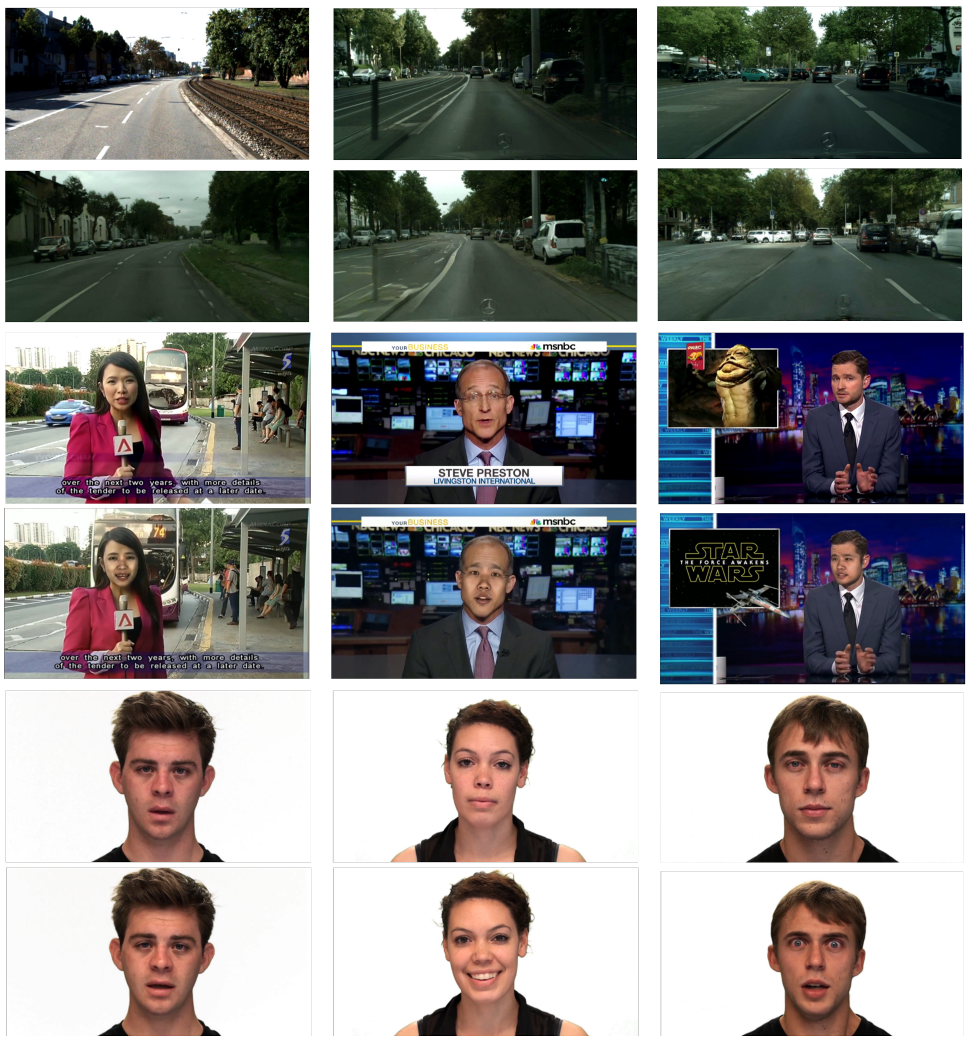
Some examples of videos from each of the three datasets are reported in Figure 5.3.

5.3.2 SVM detection, training and setting

From our discussion in Section 5.2.2 we argue that in some cases (e.g., with fully GAN-synthesized videos), a simple feature could be extracted from F and directly used for the detection, for instance, the difference $(f_{Intra} - f_{Inter})$. However, this approach would not work with non fully-synthetic fakes, e.g., in the swapping case. Since the information on the synthetic nature of the fake is unknown a priori, we resort to a more general approach and train an SVM classifier.

Given a video v_i, the footprint F_i is extracted from the first N frames of the video, skipping the first frame that is always Intra-coded.[2] Let $f_{ij} = [f_{Intra_{ij}}, f_{Inter_{ij}}, f_{Skip_{ij}}]$ be the footprint extracted from frame j. We denote with $F_i = [f_{i1}, f_{i2}, ..., f_{iN}]$ the concatenation of the N footprints. Then, F_i is the resulting feature vector extracted

[2]For all videos, the N frames following the first one are always P- or B-type, for the values of N considered in our experiments.


over the next two years, with more details of the tender to be released at a later date.
msnbc
STEVE PRESTON
LIVINGSTON INTERNATIONAL
STAR WARS
THE FORCE AWAKENS
Figure 5.3: Examples of real and fake videos from the three datasets. From top to bottom row: 3 real and 3 fake from DeepStreets, 3 real and 3 fake from DeepFor, and 3 real and 3 fake from VFHQ.

from video v_i, having dimensionality $N \times 3$. This vector is extracted from each video of the dataset $\mathcal{C}$, consisting of c videos, and used to train a linear SVM classifier. The regularization parameter C defining the separation margin adopted for the classification is set to 0.1. The procedure is reported in Algorithm 1. The notation Y is used to indicate the label vector associated with the videos in $\mathcal{C}$, that is, $Y = (y_1, \cdots, y_c)$. We denote the trained SVM model with F-SVM.

Algorithm 1: F-SVM

```
Input: $\{(v_i, y_i)\}_{i=1}^{c}$, $v_i \in \mathcal{C}$, $y_i \in \{\text{'real'}, \text{'fake'}\}$
Output: $F$-SVM model
Data  : Testing set $x$
Result: $F$-SVM
1 $F \leftarrow 0$;
2 for $i = 0$ to $\mathcal{C}$ do
3     for $j = 1$ to $N$ do
4         Extract $f_{\text{Intra}_{ij}}, f_{\text{Inter}_{ij}}, f_{\text{Skip}_{ij}}$ // Using Elecard StreamEye
5         $f_{ij} \leftarrow [f_{\text{Intra}_{ij}}, f_{\text{Inter}_{ij}}, f_{\text{Skip}_{ij}}]$;
6     $F_i \leftarrow [f_{i1}, f_{i2}, \ldots, f_{iN}]$;
7 $F$-SVM $\leftarrow$ train LinearSVM($C = 0.1$, $F$, $Y$) ;          // Using LibSVM
8 return $F$-$SVM$;
```

We have trained and tested an SVM model for all the 3 datasets described above. For DeepStreets, 50 out of 200 per-class videos are used for testing, while the remaining 150 are used for training. Actually, only 128 are made available for training, while the remaining ones are left out for validation, that is performed for the DL model used for comparison. For DeepFor, we utilized the standard dataset, which contains 1000 raw manipulated videos and 1000 real videos. The number of videos reserved for testing is 201 out of 1000 per class, while 703 are made available for training (the remaining 96 for validation). For VFHQ, testing is performed with 89 out of 596 videos per class, with 394 made available for training (113 for validation). In each case, the SVM is trained using different numbers of videos from the training set.

Since videos have different lengths and some of them are very short, for simplicity, for every video only the first $N = 30$ frames of the stream are considered (with the exclusion of the first frame which is Intra-coded), discarding the rest of the video. In the scenario considered in this chapter, we assume that the quality of the videos is matched in testing and training. In practice, this corresponds to reading the video quality from the encoded stream and using a tool trained for the same quality. To consider the same codec format, all the videos have been re-encoded H.264 with maximum quality (CRF = 0) before performing the video coding analysis. The FFmpeg software is adopted to encode the videos in H.264 codec format, while for the analysis of the video, we have used the Elecard StreamEye software [3].

Being a feature-based method, that is, a method based on handcrafted features, the developed method is expected to require less training data with respect to data-driven methods, which have also to learn suitable features for the classification during the training process. Moreover, due to the simplicity of the footprint and its low dimensionality, a simple linear SVM can be used for the classification (with the two parameters defining the separating hyperplane being the only parameters to be trained). Hence, we expect that very few videos are enough to train our model, with a significant gain in terms of the required amount of training videos.

[3]https://www.elecard.com/products/video-analysis/streameye

To prove that the detector we have developed to achieve better results than DL-based methods in data scarcity conditions, we have carried out the comparison with a frame-based XceptionNet detector, which is state-of-the-art for DeepStreet detection. For a proper comparison, we used the same splitting of the three datasets to define the training, validation and test data.

- **XceptionNet** [1] is a network with Depthwise Separable Convolutions and it represents the state-of-the-art for DeepStreets video detection [3]. Section 3.2.2 contains further details on XceptionNet. For our comparisons, we have trained the model from scratch, considering the first N frames from each video. The input face image is resized to 299×299. The network was trained using Adam optimizer with learning rate $\alpha = 0.0002$ and the default values for the first and second-order moments, that is, $(\beta_1 = 0.9, \beta_2 = 0.999)$. The batch size was set to 24 frames and the model was trained for 50 epochs. During testing, the decision on each video is taken by means of soft majority voting, i.e., considering the average of the detection scores of all the frames.

- **DeFakeHop [120]** is a lightweight deepfake detection method that can achieve detection accuracies with a small model size and a fast training procedure. The detection is conducted based on features extracted from patches (e.g., the left eye, right eye and mouth) by the PixelHop++ module and further processed by a feature distillation module with the output of the probability for the patches. Finally, the probabilities for all patches are integrated to get the final description of whether the face is fake or not. For the comparison, we have trained the model by taking N frames per video. The detection results are reported at the video level.

- **ResNetLSTM** is a network that combines ResNet and LSTM thus enabling the learning of temporal clues for deepfake detection. More specifically, the ResNet50 network is used to extract spatial features from face images. Afterwards, the extracted features from the last convolution layer are further passed through one convolutional layer (C=128, kernel=(1, 1)), an adaptive pooling layer, and an LSTM module with 512 hidden units to learn temporal discrepancies between frames. Finally, a fully connected layer is used for fake prediction. Similarly, for each video, we take N sequences of length 7 from N frames per video to train the model, and the input face is resized to 224×224. The loss is calculated for all the frames during the training. Only the features of the last frame are used for prediction during the test. For the other parameters, we have used the same settings as for XceptionNet training.

5.4 Experimental Results

We first demonstrate the effectiveness of the detector on the DeepStreets dataset. Then, we also test its effectiveness on the DeepFor and VFHQ datasets, where only a limited area of the video frames is manipulated.

5.4.1 Comparison results

Table 5.1 reports the accuracies achieved by a detector based on XceptionNet [3] and our method for different numbers of videos c used to train the classifier (the number reported refers to each class). Results are reported for different compression levels (Raw, HQ, LQ). We see that, for the Cityvid and Citywcvid subsets, in all of the cases, our detector can achieve an accuracy around 98% by training on just 4 videos, while XceptionNet needs 64 videos (32, in very few cases). For the Kittyvid subset, our method can not achieve the same accuracy of XceptionNet for a large number of training videos, which means that the features are not enough discriminative to distinguish the fake videos from the real ones. The lower discrimination capability, in this case, can also be argued by looking at the feature distribution of the Kittyvid dataset in Figure 5.1. A reason can be that in the case of Kittyvid, the pristine videos come from a completely different dataset with different backgrounds and scenes (temporal) variability with respect to the fake images. The different temporal variability might affect the video coding features. On the contrary, the significantly different content and background in real and fake videos in the Kittyvid case is beneficial for XceptionNet which can look at the diversity of the background to discriminate between real and fake videos. Nevertheless, the superior performance of our method under limited training data is confirmed also in this case, given that it reaches an accuracy of around 80% with 2 training videos only (for high-quality videos).

We have also performed a cross-data analysis for the three subsets in DeepStreets, and the results are reported in Table 5.2 for raw videos. The results refer to the case of training with 120 videos when both our and XceptionNet methods can achieve good performance. From the table, we see that our method generalizes extremely well, improving the poor generalization capability of XceptionNet, with the exception of the Kittyvid case where, however, the results of our methods are low also in the matched case, see Table 5.1.

The performance achieved on the DeepFor and VFHQ datasets are reported in Table 5.3. In the case of DeepFor, we see that we can achieve an accuracy of 91.4% by training our system with only 8 videos, while XceptionNet, ResNetLSTM and DeFakeHop need respectively 128, 16 and more than 256 videos to reach a similar accuracy (92%). Notably, the performance of our method is already close to the maximum around $c = 32$, and the accuracy does not increase further using many more videos, while the performance of other methods increases while increasing c. On the VFHQ dataset, the difference is lower, however, the good performance of the developed with few videos is confirmed. The F-SVM detector has much better performance than XceptionNet and DeFakeHop, reaching an accuracy of 90.1% with 16 videos. However, in this case, ResNetLSTM can get an accuracy larger than 90% with just 8 videos. Since in the case of VFHQ dataset only the facial expressions are manipulated, with the head poses remaining the same, it is not surprising that the detection based on the proposed footprint is more difficult. Moreover, in the VFHQ dataset, there is a significant motion in the background. However, since the background is not manipulated, the prediction modes from the macroblocks in the background negatively affect the footprint estimation for fake videos.

Another advantage of our method is the limited computational effort required to train it and the small size of the trained model. Table 5.4 reports the training time for the

Table 5.1: Detection accuracy for the three different subsets in the DeepStreets for both our method and the state-of-the-art XceptionNet method in [3]. Results are reported for different video qualities (Raw, HQ and LQ).

No.	Cityvid						Citywcvid						Kittyvid					
videos	Raw		HQ		LQ		Raw		HQ		LQ		Raw		HQ		LQ	
c	F-SVM	Xcept	F-SVM	Xcept	F-SVM	Xcept	F-SVM	Xcept	F-SVM	Xcept	F-SVM	Xcept	F-SVM	Xcept	F-SVM	Xcept	F-SVM	Xcept
2	**96.9**	50.0	**97.5**	50.0	**95.8**	50.0	**96.7**	50.0	**98.6**	50.0	**99.0**	50.0	**80.0**	50.0	**75.0**	50.0	**67.8**	50.0
4	**97.2**	50.0	**98.5**	50.0	**98.3**	50.0	**98.5**	55.0	**98.9**	67.0	**99.6**	52.0	**81.1**	50.0	**76.2**	50.0	**71.0**	50.0
8	**98.5**	88.0	**98.4**	78.0	**99.5**	84.0	**98.5**	70.0	**99.8**	71.0	**99.7**	84.0	82.1	**99.0**	81.2	**100**	72.1	**100**
16	**98.6**	90.0	**99.0**	94.0	**99.5**	81.0	**99.7**	90.0	**100**	87.0	**99.8**	92.0	85.0	**100**	82.8	**100**	74.2	**100**
32	**98.8**	97.0	**99.1**	66.0	**99.7**	89.0	**99.9**	96.0	**100**	97.0	**99.9**	97.0	86.8	**100**	86.8	**100**	76.6	**85.0**
64	**99.4**	87.0	99.4	**100**	**99.7**	99.0	**100**	**100**	**100**	**100**	**100**	86.0	87.9	**100**	85.3	**100**	79.5	**99.0**
128	99.0	**100**	**100**	**100**	**100**	96.0	**100**	**100**	**100**	**100**	**100**	**100**	91.6	**100**	91.3	**100**	81.5	**100**

Table 5.2: Cross dataset results for DeepStreets on raw videos (120 videos are used for training).

Test set	Method	Train set		
		Cityvid	Citywcvid	Kittyvid
Cityvid	Xcept	-	73.0	50.0
	F-SVM	-	**99.3**	**80.0**
Citywcvid	Xcept	49.0	-	50.0
	F-SVM	**99.0**	-	**53.8**
Kittyvid	Xcept	**99.0**	**75.0**	-
	F-SVM	85.5	69.0	-

Table 5.3: Detection accuracy for DeepFor and VFHQ datasets for our method and the state-of-the-art methods.

No. train	DeepFor				VFHQ			
videos (c)	F-SVM	Xcept	DeFakeHop	ResNetLSTM	F-SVM	Xcept	DeFakeHop	ResNetLSTM
2	67.5	50.0	61.3	65.9	61.7	50.0	60.6	58.4
4	74.8	50.0	62.8	70.6	75.5	50.0	63.8	77.5
8	**91.4**	50.0	62.0	85.9	84.8	50.0	69.9	**93.3**
16	**95.5**	53.0	72.8	**91.7**	**90.1**	73.6	72.6	**90.8**
32	**96.5**	56.0	79.8	**97.1**	**92.3**	**91.0**	77.8	88.6
64	**96.8**	83.0	85.6	**99.0**	**93.5**	**94.9**	80.9	88.3
128	**96.8**	**92.3**	88.5	**98.0**	**94.4**	**95.5**	79.8	**93.2**
256	**97.0**	**98.5**	89.7	**99.9**	**96.2**	**94.4**	82.7	**94.3**

various methods computed for the case of 16 videos for the DeepFor and VFHQ datasets. For our method, the time needed for the feature extraction is also considered.[4] By relying on a simple linear SVM classifier trained on the extracted low dimensional footprints, the training of our method is extremely fast (with only two parameters to be learnt from the data), much faster than the other methods. The size of the trained models, which is the same for the two datasets, is also reported in the table, showing that the SVM detector has a very small size compared to the other models.

For the Xcept and ResNetLSTM networks, we performed all of the experiments by using the PyTorch framework on a workstation with one Intel i7-6850K CPU and four NVIDIA GTX1080 12GB GPU while for F-SVM and DeFakeHop the experiments were performed on a local PC with one Intel core i7-8700 CPU and 32GB RAM.

[4]For the other methods, only the training time is considered since the data pre-processing time (the time for face detection and extraction - for XceptionNet and ResNetLSTM - and the time for landmark detection and patch - eyes and mouth - extraction for DeFakeHop) is negligible compared to the training time.

Table 5.4: Comparison in terms of training time, number of parameters and model size on DeepFor and VFHQ datasets for the case of 16 training videos.

Methods	F-SVM	Xcept	DeFakeHop	ResNetLSTM
Number of parameters	2	22.8M	42.8K	25.1M
Training time (DeepFor)	0.7281s	18m34s	23.7251s	1h15m24s
Training time (VFHQ)	0.8584s	17m45s	27.9549s	1h17m45s
Model size (DeepFor&VFHQ)	10KB	79.7MB	76KB	96.0MB

5.5 Conclusions

In this chapter, we have presented a feature-based method for video deepfake detection that can work in data scarcity conditions, that is, when only very few examples are available to the forensic analyst. The method is based on a simple video coding feature, counting the frequency of motion prediction modes in the video sequence. Results carried out on three different datasets show the effectiveness of the method, that only needs very few video samples to train the detector.

While our method demonstrates notable strengths, particularly its lightweight nature and efficacy in scenarios with limited fake video samples, time constraints, and the absence of high-end computing servers equipped with GPUs, it is important to acknowledge its limitations compared to DL-based approaches. In situations where abundant training data is available and extensive computational resources can be utilized, DL methods typically outperform our approach. However, the practicality and efficiency of our method in real-world scenarios make it a valuable tool, especially in the case in which immediate results are needed and there is an absence of servers equipped with GPUs.

Part II

Image Geolocalization

Chapter 6

Photo Geolocalization

La semplicità è la suprema sofisticazione.
Simplicity is the Ultimate Sophistication.
Leonardo da Vinci

Image Geolocalization is receiving increasing attention due to its importance in several applications, such as the identification of fake news, criminal investigations and the prevention of disinformation campaigns. Previous works focused on several instances of Image Geolocalization including place recognition and GPS coordinates estimation. Yet, recognizing in which country or city an image was taken could be more critical, from a semantic and forensic point of view, than estimating its spatial coordinates. In this and the following chapters, we address the problem of Image Geolocalization with a specific focus on recognizing the country and city where an image was captured.

6.1 Introduction

In the first part of the thesis, we have presented our contributions to the detection of deepfake videos. In this part, we direct our attention toward the deceptive recontextualization of photos. This deceptive practice involves misrepresenting images within news posts or articles, wherein a photo portraying events like war, floods, protests, or earthquakes is asserted to originate from a specific location, when, instead, the image has been taken in a different place.

The exact localization where an image has been taken is particularly crucial for fact-checking applications. This capability serves as a pivotal tool in the battle against the spread of fake news and misinformation campaigns. Furthermore, the effectiveness of accurately identifying the specific depicted location in a photograph proves highly valuable in uncovering discrepancies between textual content and images within news articles.

Image Geolocalization is the process whereby the geographical location where a photo has been taken is identified, be it through precise GPS coordinates or by identifying the street address, city, or country of the scene depicted in the photo. Geolocalization is an extremely challenging task due to the existence of a vast diversity of photos, encompassing varying landscapes, environmental factors, and architectural styles. Moreover, several factors contribute to increase the complexity of determining the geographic origin of an image. Elements like the time of the year, the lighting conditions (day or night), and camera settings, can significantly influence the accuracy of geolocalization. In the literature, numerous studies have taken on this formidable challenge, employing a variety of approaches and methodologies to tackle the complexities associated with Image Geolocalization.

Methods for Image Geolocalization can be broadly categorized into two main groups: those ones concentrating on identifying specific urban areas or landmarks and those tackling the problem on a global scale without imposing geographical restrictions. The former category, sometimes referred to as "Visual Place Recognition" usually relies on retrieval methods, involving the matching of a query image with a reference dataset. Within this group, a subset of methods places a strong emphasis on landmark recognition, utilizing either a predefined set of landmarks or unsupervised clustering techniques applied to photo collections [121–126]. In contrast, the later category take a more fine-grained approach, addressing the problem on a global scale. The fine-grained methods often make extensive use of DL techniques to infer the location of a photo. The methods divide the Earth's surface into distinct classes, with each class associated to a specific GPS coordinate. These approaches enable the model to directly predict the class, effectively by passing the need for feature comparison with a reference database [2, 127–130].

6.2 Motivation and Contributions

In this part of the thesis, we introduce two novel contributions in the realm of Image Geolocalization: the identification of the country and the verification of the city where an image was captured. Often, determining the country or city of origin is not only of a greater importance but also potentially more manageable than providing a precise estimation of the geo-coordinates pinpointing the scene within the image.

As a first contribution to address the Country Recognition challenge, we have built the VIPPGeo dataset, consisting of 3.8 million geo-tagged images primarily focused on urban outdoor scenes. This dataset is strategically curated to facilitate the detection of architectural and semantic disparities between countries, achieved through data filtering to exclude natural scenes, indoor images, grayscale content, and images dominated by irrelevant subjects like ships or aeroplanes. Then, we used the dataset to train a classifier based on DL to distinguish among countries. We have organized the countries into 61 classes, accommodating even the smallest countries and those with limited example images. The results we got, by testing the classifier on a subset of the VIPPGeo dataset, demonstrate a substantial performance enhancement compared to state-of-the-art baseline methods relying on estimated geospatial coordinates for Country Recognition.

Additionally, we have adopted a complementary perspective, focusing on the verification of the specific city where an image was taken. In some cases, recognizing the source city proves to be of greater significance than providing an exact geo-coordinates estimate of the scene. On the other hand, recognizing only the country where the image was taken often falls short of achieving the necessary level of localization precision.

City Verification poses a substantial challenge due to two critical factors: i) the huge number of cities potentially to be considered in the analysis, and ii) the considerable variability in scenes originating from the same city. To address these challenges, we have adopted neither an inference nor a classification approach and re-framed the city recognition problem as a verification task. Specifically, we present a system based on a Siamese network [131], designed to determine whether a source image was captured in a target city, with the assumption that a small set of images from the target city are available. In particular, our system assigns greater voting power to reference images that exhibit semantic similarity to the query image, allowing for more accurate verification. We argue that with a sufficiently large number of reference images, the verifier can effectively handle the diversity of images originating from the same city, ultimately improving its overall performance and reliability.

6.3 Related Work

In this section, we start by reviewing the most common approaches for global-scale Image Geolocalization. Then, we will briefly review Visual Place Recognition (for both methods relying on DL and image retrieval). Subsequently, we review the publicly available Image Geolocalization tools.

6.3.1 Global scale Image Geolocalization

The first attempt towards planet-scale Image Geolocalization, has been described in Im2GPS [132], in which a query image is matched against a database of six million geotagged images and the location is inferred from the retrieved set. Subsequently, in [133], Im2GPS has been improved by incorporating multi-class SVM, to refine the search process. Another noteworthy work is [127], where the authors proposed learning a feature representation with CNNs to enhance the performance of Im2GPS [132]

In Planet [129], the authors have formulated the problem of Image Geolocalization as a classification problem. The earth's surface was divided into thousands of multi-scale geographic cells and a CNN model was trained using millions of geotagged images. However, the granularity of partitioning in Image Geolocalization is crucial, as larger cells may yield lower location accuracy, while smaller cells may result in reduced training examples per class, making the model susceptible to overfitting. To address this issue, CPlanet [130] have proposed a combinatorial partitioning algorithm that generates a multitude of fine-grained output classes by intersecting multiple coarse-grained partitioning of the earth's surface.

One of the best-performing systems proposed to date is [2]. Images taken in various types of environments (urban outdoor, indoor or natural) are incorporated, so as to embed in the learning process specific features of several environmental settings. The DL architectures is based on the ResNet network [134]. Results obtained on benchmark datasets demonstrate the excellent performance of this system, positioning it as a reference benchmark for Image Geolocalization. In this thesis, we employed the system introduced in the study by Muller et al. [2] as a benchmark for our Country Recognition task. Furthermore, we adapted it to be used as a city verifier, facilitating a meaningful comparison with our outcomes in the City Verification task.

In [135] Izbicki et al. have introduced the Mixture of von Mises Fisher (MvMF) loss function which is able to exploit the spherical geometry of the Earth to improve geolocalization accuracy. The MvMF model exhibits a remarkable ability to merge aspects of both classification and image retrieval methods. The authors claimed that this unique feature allows to harness the strengths of both techniques while by passing their respective limitations.

In [136], the authors have introduced a mixed classification and retrieval scheme, combining the strengths of both methods in a unified solution, achieving new state-of-the-art performance at fine granularity scales. In particular, their approach involves two models: the first employs the EfficientNet architecture to robustly assign images to specific geographic cells, while the second introduces a novel residual architecture trained with contrastive learning. The second model maps input images to an embedding space that minimizes distances between images of the same location. To finalize location estimation, these two modules are merged using a search-within-cell approach. Here, similar images within the predicted geographic cell are aggregated based on a spatial clustering scheme.

In a more recent study [137], a selective prediction method has been introduced to assess the suitability of an image for the geolocalization task, resulting in the removal of non-localizable images and thereby increasing the overall accuracy. The authors have suggested a selective prediction method [138] to tackle this task. In the work, the authors introduce two novel selection functions that utilize the output probability distributions of geolocation models. These functions are designed to infer the feasibility of localization at various scales. For instance, when an image is inputted into the geolocation model, it generates a cell probability distribution across a world map. The selection functions evaluate whether the model's confidence is focused on a specific region or spread across the globe. These functions assess the localizability of the input image based on this assessment.

Pramanick et al. [128] recently have introduced TransLocator, a unified dual-branch transformer network that attends to tiny details over the entire image and produces robust feature representation under extreme appearance variations. By incorporating RGB images and their corresponding semantic segmentation maps, TransLocator coordinates interactions between parallel branches at each transformer layer, enabling simultaneous geo-localization and scene recognition in a multitasking framework. This method marks a significant improvement over existing state-of-the-art approaches.

Clark et al. [139] have presented GeoDecoder; a transformer-based architecture designed to exploit the relationship between various geographical levels and the visual scene information within an image. This is achieved through hierarchical cross-attention, where the model learns distinct queries for each geographic hierarchy and scene category. Similar to ISN [2], GeoDecoder involves dividing the earth into cells. However, they expand the hierarchy levels from 3 to 7 and broaden the scene categorization to encompass 16 environmental contexts, such as forests, workspace, industrial scenes, etc. Additionally, GeoDecoder develops separate representations for various environmental scenes, as different scenes in the same location often exhibit entirely different visual features. This approach allowed GeoDecoder to achieve state-of-the-art accuracy on standard geo-localization test datasets.

Cepeda et al. [140] have introduced GeoCLIP, a novel Image-to-GPS retrieval model inspired by CLIP [141]. GeoCLIP aligns images with their respective GPS locations using a location encoder that represents the earth as a continuous function. This encoder leverages positional encoding with random Fourier features and constructs a hierarchical representation capturing information at various hierarchy levels, resulting in semantically rich high-dimensional features. Notably, [140] is the first work employing GPS encoding for Image Geolocalization. GeoCLIP demonstrates competitive performance, achieving notable results with only 20% of the training data used in other methods like [2,128,139], which shows GeoCLIP efficacy even in limited-data scenarios.

In order to explain the behaviour of geolocalization models, a recent study, by [142], introduced a novel semantic partitioning method, capable of enhancing the interpretability of prediction results, still achieving state-of-the-art results in terms of geolocalization accuracy on benchmark test sets. Particularly, instead of dividing the earth into geometrically shaped cells, whether rectangular or arbitrary, semantic partitioning involves defining meaningful and understandable cells based on real-world features such as territorial boundaries (like streets, cities, or countries), natural elements (such as rivers or mountains), or human-made structures (like roads, railways, or buildings). These divisions are extracted from the Open Street Map (OSM) [143] for a more contextually relevant representation. In addition, the approach introduces a concept influence metric aimed at examining the interpretability of the results. This metric measures the impact of semantic visual concepts on individual predictions. Experimental findings demonstrate that semantic partitioning achieves state-of-the-art performance. Moreover, the concept influence metric offers valuable insights into the visual concepts contributing to both accurate and misleading predictions.

Most recently, and somewhat similarly to the work described in this thesis, research has focused on Country Recognition. In G^3 [144], the authors have shown how language can be leveraged to improve Image Geolocalization. Their approach involves predicting the country of an image by exploiting a set of clues extracted from a textual Guidebook for the GeoGuessr game. The Guidebook clues describe the salient and class-discriminative visual features humans use to geolocalize images. Leveraging the CLIP model, the method involves generating image embeddings for the queried image and the Guidebook clues. Then, it employs a weakly supervised attention layer to compute a weighted average over the clue embeddings, creating a text-based representation relevant to the image. Throughout the training, the attention layer is weakly supervised with positive examples of clues that match the image's ground truth country. The image embedding and the clue representation are then combined before being fed into a classifier. Remarkably, G^3 significantly outperforms generic geolocation methods in the Country Recognition task, underscoring its effectiveness in this domain.

6.3.2 Visual Place Recognition

A related, yet different, task that has received significant attention, is "Visual Place Recognition" (VPR), where the main goal is to verify if two pictures have been taken in the same location, despite the significant changes in appearance, the viewpoint and other environmental conditions. The classical workflow for "Visual Place Recognition" is based on retrieval methods and consists of three steps. In the first step, an encoding feature vector is obtained from each original input image. Subsequently, a similarity search is performed between the query image and the images in the database. The similarity search leads to the identification of possible candidate images similar to the queried one. The final step consists of finding the nearest image in the queried database and sorting images according to some pre-defined distance measure (for instance the Euclidean distance).

In [121], the authors have proposed a method for large-scale “Visual Place Recognition”, aiming at recognizing the location of a given query photograph. However, the authors considered only two cities: Tokyo and Pittsburgh. In [145], the authors have developed a model to learn the relationship existing between aerial, land cover and ground-level images, in an area of around 1660 km^2. Other works focus only on landmark images, for example in [146,147], or aerial images, as in [12,148]. Another interesting work in the field is [149]. Here the authors have proposed a workflow, named Visual Geo–localization, based on a place recognition system in which a query image is compared to a database of images with the aim of finding images taken at the same location. Interested readers may explore comprehensive surveys on “Visual Place Recognition” in the following references: [9,150–152].

6.3.3 Image Geolocalization datasets

An important aspect to be taken into account in the realm of Image Geolocalization is the availability of reference benchmarking datasets. Developing and testing deep neural networks or even image retrieval systems, in fact, requires the availability of large amounts of data that capture the differences (architectural, stylistic, social, etc. . .) among world countries. The performances of deep neural networks (be they fine-tuned or trained from scratch) largely depend on the availability of massive amounts of data to train them.

A few datasets for “Visual Place Recognition” have been released. For instance, the Pittsburgh250k [153] and the Tokyo 24/7 datasets were used to train and test “Visual Place Recognition” methods [154]. However such datasets are based only on two specific cities (Pittsburgh and Tokyo) and therefore are not relevant for more general tasks, and the Country Recognition tasks in particular. A few other datasets specific to one or few cities have been released, for instance, Oxford [155] and Freiburg [156].

In contrast, there is currently only one available dataset for global-scale Image Geolocalization online. This dataset is derived from a subset of the Yahoo Flickr Creative Commons 100 Million dataset (YFCC100M) [157], specifically introduced for the MediaEval Placing Task 2016 (MP-16) [158]. It comprises approximately five million geo-tagged images from Flickr, and notably, it comes without any usage restrictions. However, the (MP-16) dataset comprises some images that aren’t well-suited for geolocalization purposes, such as portraits or cartoons.

A benchmark dataset that we utilized in our experiments is the Im2GPS dataset [132], along with its extension, Im2GPS3k [127]. Im2GPS was initially introduced in 2008 and consists of 237 images designed for testing purposes. On the other hand, Im2GPS3k, released in 2017, offers a more extensive collection with approximately three thousand images. These datasets hold a significant position as a benchmark for Image Geolocalization research. Table 6.1 summarizing the available Image Geolocalization datasets.

Table 6.1: Available datasets for VPR and Planet-scale Image Geolocalization.

Dataset	Number of images	Intended task
Pittsburgh250k [153]	250,000	Visual Place Recognition
Tokyo 24/7 [154]	1,250	Visual Place Recognition
Oxford [155]	20 million	Visual Place Recognition
Freiburg [156]	30,790	Visual Place Recognition
mp-16 [158]	5 million	Global Image Geolocalization
Im2GPS [132]	237	Global Image Geolocalization
Im2GPS [127]	3,000	Global Image Geolocalization

To address the shortage of public datasets for Image Geolocalization, we have built the VIPPGeo dataset containing 3.8 million geo-tagged images collected from three databases; Flickr [159], Mapillary [160], and Unsplash [161]. We have meticulously crawled these images to ensure comprehensive coverage of 243 countries worldwide. The dataset exclusively consists of urban outdoor scenes, a choice made to highlight architectural and semantic distinctions among countries. To retain only relevant data, we have implemented several filtering strategies, excluding natural scenes, indoor images, grayscale content, and images dominated by irrelevant subjects such as ships or aeroplanes. We will discuss the VIPPGeo dataset in more detail in the next chapter.

Chapter 7

Country Recognition

If I have seen Further Than Others,
It is by Standing on the Shoulders of Giants.
Isaac Newton

Within the confines of this chapter, we address the problem of Country Recognition with two contributions. We introduce the VIPPGeo dataset, containing 3.8 million geo-tagged images. Then, we use the dataset to train a model casting the Country Recognition problem as a classification problem. The experiments we run show that our model provides better results than the current state of the art. Notably, we found that asking the network to identify the country provides better results than estimating the geo-coordinates and then tracing them back to the country where the picture was taken.

7.1 Introduction

Planet-scale geolocalization methodologies primarily approach the task of Image Geolocalization as an inference problem, aiming to estimate the GPS coordinates of the image scene with high accuracy [2,129,137]. As we mentioned in the previous chapter, many of these methods rely on DL networks as a backbone. Although these networks are formulated as classification networks, the final goal often revolves around minimizing the error between estimated and true geo-coordinates, thus qualifying the problem as an inference one.

Our approach to Country Recognition diverges from the inference-driven perspective by reframing the task as a classification problem. Rather than solely estimating the geographic GPS coordinates of an image, our focus is to identify the country where the image was captured. In most cases, in fact, recognizing the country is more important (and possibly easier) than providing a precise estimate of the geo-coordinates of the scene framed by the picture. One may argue that once the geo-coordinates have been estimated, tracing back to the country is a trivial task. This is true only in an ideal case; however, in non-ideal conditions, a system that is asked to minimize the spatial localization error may result in a wrong classification of the country if this allows for the reduction of the localization error. On the other hand, being a typical human construction, countries (usually) define geographical regions with similar human-related features, like historical landmarks, language, architecture, and social habits, and such characteristics may be conveniently used to identify the country a certain image belongs to.

Considering this, our contribution can be summarised as follows:

1. We have built a new dataset, named VIPPGeo, to be used for country classification applications. The dataset contains almost 4 million geo-tagged images obtained by crawling the three public databases of Flickr [159], Mapillary [160] and Unsplash [161]. The images were crawled according to a list of 243 world countries to obtain a comprehensive representation of the whole set of world countries. The photos contain urban outdoor scenes only, due to the relevance of such kind of data for the problem at hand, in particular for detecting the architectural and semantic differences between countries. Data filtering strategies were applied to select and crawl only relevant images, discarding, for instance, natural scene images, indoor images, grey-level images, and images mostly occupied by specific, non-relevant, subjects for instance ships or aeroplanes.

2. We have used the newly constructed dataset to train a classifier based on a Resnet-101 architecture to distinguish among countries. In order to take into account the presence of the very small countries and countries with few sample images in the dataset, we did so by grouping the countries into 61 classes. The results we got by testing the classifier on a test subset extracted from VIPPGeo (and not used for training and validation), show a significant improvement of the performance with respect to a state-of-the-art baseline method [2] recovering the country from the estimated geospatial coordinates [1].

The rest of the chapter is organized as follows. In Section 7.2, we present the VIPPGeo dataset. The developed classification method is described in Section 7.3. In Section 7.4, we describe the experimental setting and the results we got. Finally, we draw our conclusions and outline some perspectives for future research in Section 7.5.

7.2 The VIPPGeo Dataset

In this section, we describe in details the construction and the content of the VIPPGeo dataset. We have built the dataset by using publicly available sources, namely: Flickr, Mapillary and Unsplash. In this way, we gathered photos with different characteristics, taken with different cameras, from a wide range of different photographers and largely diverse views of all world countries. The images were crawled by using the APIs released by each of the three data sources.

Flickr

Flickr [159] is a public repository where users can upload and share their personal photos. It was created in 2002 by a Canadian company. Since 2018, it has been a property of SmugMug. Photos uploaded on the website can have several different types of copyrights. To build our dataset, we have only retrieved photos which are freely usable with no

[1]Our experiments show that the improvement is partly due to the use of VIPPGeo for training, and partly to the choice of classifying the countries directly without passing from the geo-coordinate estimation first.

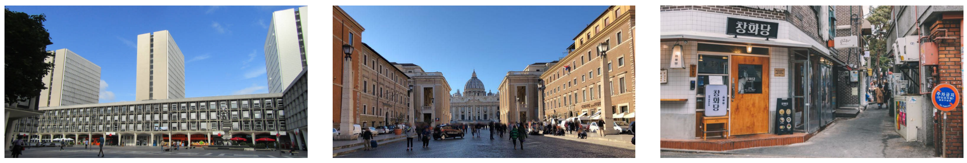
Figure 7.1: Examples of images with Creative Commons licenses retrieved by the Flickr API for the query *'London church'*.

copyright associated. In particular, the images used are all made available under the Creative Commons license.

To build our dataset, we have used the YFCC100M dataset [157] containing about 5 million geo-tagged images. In addition, we have retrieved images from Flickr which are relevant to our task. We have used search queries containing 183 Urban landmarks keywords, as for instance “university”, “museum”, “skyline” etc .... The keywords were paired with all the cities/towns in the world with a population of over 1000 people [162] for a total of 144,563 cities. An example of the queries we have used is *'London church'* (filtering in order to obtain only those images which have a Creative Commons license. See Figure 7.1.). In total, we have sent:

$$183 \text{ keys} \times 144,563 \text{ cites} = 26,455,029 \text{ queries},$$

from which we have collected around one million images. We have merged these images



Flickr - France. Mapillary - Vatican City. Unsplash - S.Korea.

Figure 7.2: Three sample photos taken from Flickr, Mapillary and Unsplash.

with the YFCC100M dataset [157], in this way we obtained around 6 million images.

Mapillary

Mapillary [160] is a web platform allowing crowd-sourcing geo-tagged images. Originally developed by a remote Swedish company, Mapillary AB. It was later acquired by Facebook in 2020. To retrieve the urban images that we have included in our dataset, we have iterated our queries over each city/town in the world with more than 1,000 inhabitants [162]. Then, we requested the Mapillary API to return all images in a bounding box of 10 kilometres around the GPS location of each city. In this way, we obtained a total of 2,528,328 images.

Unsplash

Unsplash [161] is a website devoted to photo-sharing. The site has a growing database of more than 3 million photos. Unsplash has been cited as one of the world's leading photography websites, that provides high-quality images. In our dataset, we have crawled the images from the publicly available 'Full Dataset' database.

All images of the VIPPGeo dataset are stored in JPEG format with a quality factor equal to 75. With regard to the image size, when the smallest dimension of an image was larger than 640, we resized it so that the smallest dimension was equal to 640, otherwise, we stored the images in their original dimension. In Figure 7.2, we show three examples of images taken from Flickr, Mapillary and Unsplash.

7.2.1 Pre-filtering

In order to maximize the significance of the VIPPGeo dataset, we applied several filtering strategies to remove unsuitable or meaningless images. In particular, we have applied the following filters:

- **Remove non–urban images**: We have decided to focus exclusively on urban images. Our original research question, in fact, was to analyse if it is possible to recognize a country given the architectural, language, societal and other human-oriented details contained in the picture. Natural scenery images would not contain

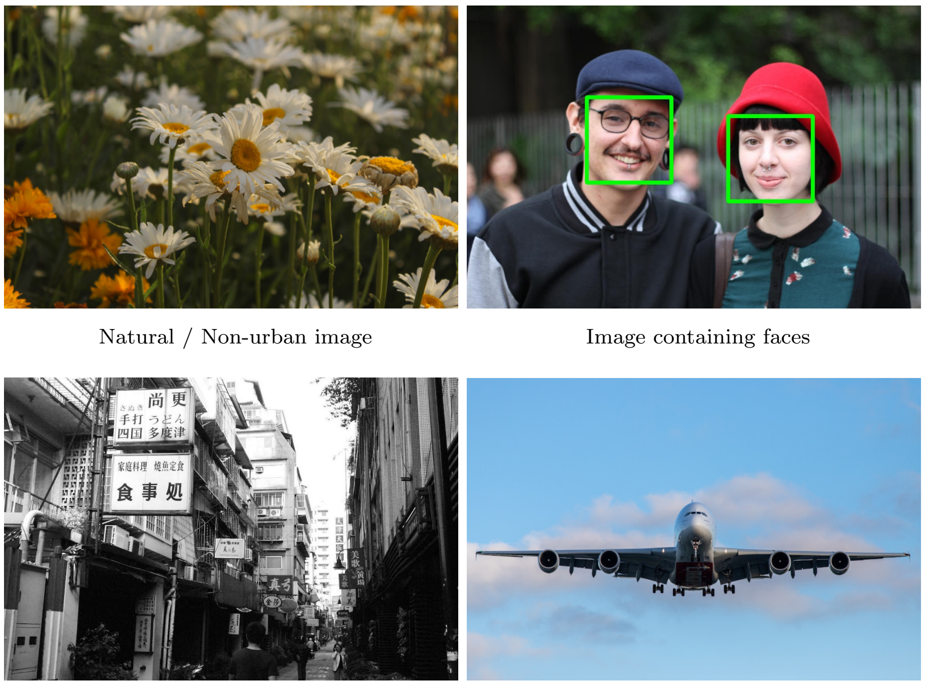


Figure 7.3: Examples of filtered images from the VIPPGeo dataset.

such details and, in general, are not suited for the Country Recognition task. Indoor images might also be misleading, given that they may contain design choices that are not necessarily representative of a specific country or region. In order to implement this filter, we have used the Places365-CNNs model [163]. In particular, Places365 classifies the images according to 365 types of scenery, which can be further categorized as indoor, natural (outdoor natural) and urban (outdoor man-made). To collect the urban images for our dataset, we have used the trained ResNet model of Place2 dataset [163] to get the Top-5 predicted classes. Afterwards, we summed the prediction probabilities of the urban classes contained in the Top-5 list. Finally, we retained only the images with an overall urban probability larger than 0.5.

Since the classification provided by Places365 is not perfect, we have decided to apply some additional filtering rules as discussed in the following.

- **Remove grey-level and outdated images**: We have decided to discard grey-level images. The first reason for doing that was to enforce a certain uniformity across the dataset. This is particularly important for the images obtained from Flickr, which can contain any type of coloured images (also including historical images). A second, more compelling, reason is that the features we are interested in, mostly related to human, societal and architectural factors, vary across history. This, in turn, would change significantly the urban look of a country, making it difficult to distinguish the peculiar features of a country over time. Given that grey-level images *tend* to be old ones, we have decided to discard them. For a similar reason, we have decided to remove images taken before 2012. This was possible by looking at the image timestamp, or by looking at the date when the image was uploaded. In order to filter out grey-level images, we first discarded the images consisting of one colour channel only. For the others, we have computed the standard deviation across the image bands and discarded the images for which the deviation was smaller than a threshold.

- **Remove images portraying *mostly* faces**: Upon inspection of the images we have downloaded, we found that several of them are mostly occupied by one, or more, faces. This effect was particularly relevant for Flickr images. Clearly, these kinds of images are not very relevant for Country Recognition, hence we have decided to discard them. In particular, we have removed the images wherein faces occupy more than 10% of the image area. We did so by using the dlib library [164] to detect and localize faces in the images.

- **Remove, specific, non-relevant, image classes**: We also found that many of the images we have downloaded focus on specific, non-relevant, subjects like ships in the sea, airfields, aeroplanes in the sky, and details of sports stadiums. We can argue that; such kinds of images are not suitable for Country Recognition. For instance, an aircraft flying in the sky belonging to a certain airline could be anywhere in the world and therefore is not representative of a country's architectural or semantic style. The same applies to ships. Additionally, sports stadiums such as soccer fields, raceways, and race circuits are hard to classify, unless they contain very specific iconic spots.

In total, after applying the filtering strategies described above, we reduced the number of images from 12 million to 3,813,651. Some examples of images discarded from VIPPGeo

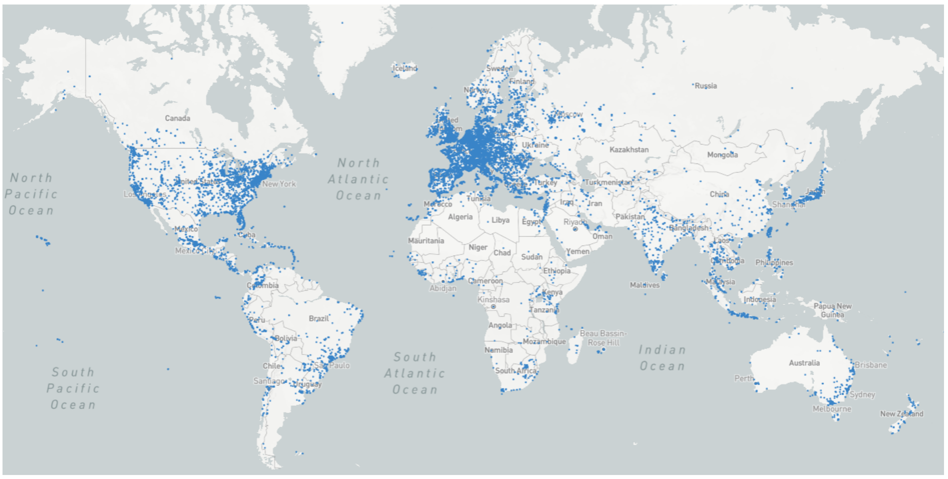


Figure 7.4: The geographical distribution of images in our dataset.

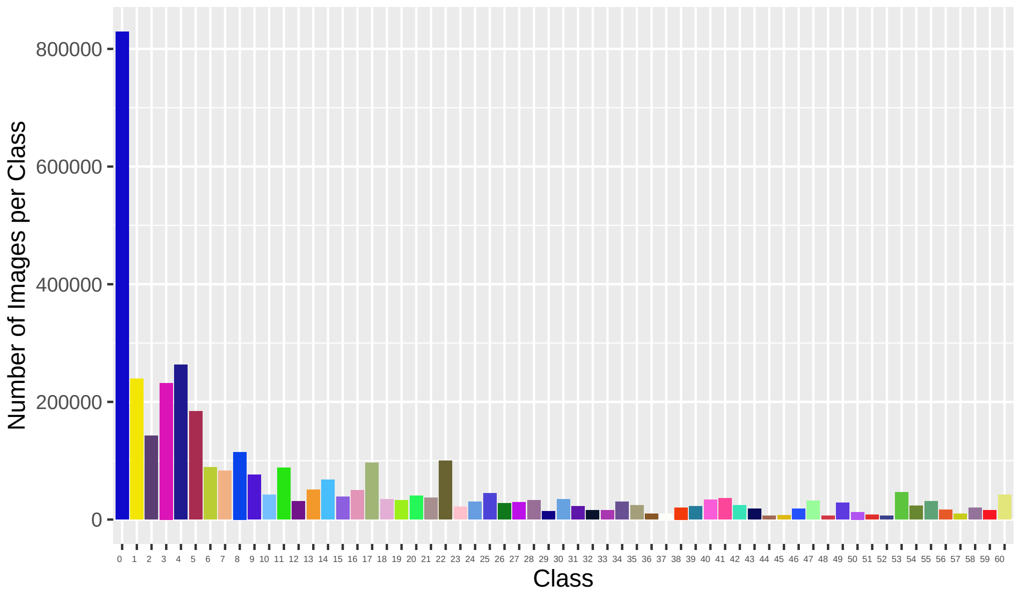


Figure 7.5: Histogram showing the distribution of the number of images for each of the 61 classes. The x axis shows the 61 classes and the y axis shows the number of images in each class.

are reported in Figure 7.3. The geographical distribution of images in our dataset is shown in Figure 7.4. The VIPPGeo dataset is publicly available at the following link: `https://github.com/alamayreh/VIPPGeo_Dataset`.

7.2.2 Dataset composition

The dataset contains a total of 3,813,651 Images. All of the images are geo-tagged with the geo-tagging information available along with the image. For each image in the dataset, we give the GPS location, together with the country. In particular, the dataset contains 1,716,636 images downloaded from Mapillary, 324,018 from Unsplash, and 1,772,997 from Flickr. The images come from a total of 243 different countries, including very small countries, like Vatican City, or the Principality of Monaco.

While the VIPPGeo dataset can be used as is, that is with each image labelled as belonging to one of 243 countries, some of the countries are extremely small ones. For others, the number of images contained in the dataset is very small. For instance, we were able to gather only 231 images from Kenya and 284 images from Libya. For this reason, in our experiments, we found it convenient to group the 243 countries into larger classes consisting of nearby countries. In total, we have grouped the images into 61 classes. Each class includes several countries, which we have decided to merge based on two main properties/characteristics: the number of images and semantic similarities between countries. With regard to the first criterion, we have tried to balance as much as possible the number of images in each of the classes, to avoid possible unbalancing issues

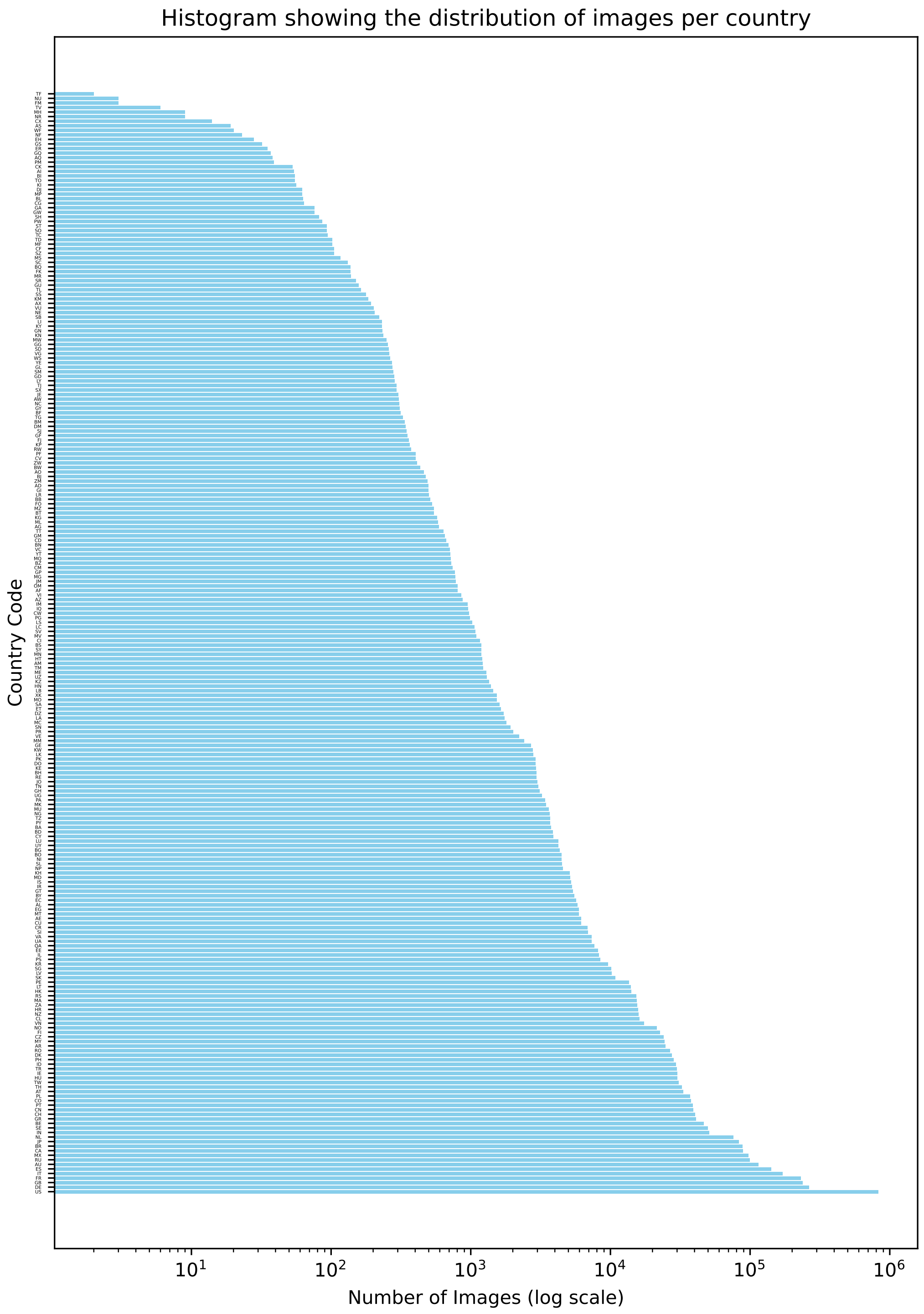


Figure 7.6: Histogram showing the distribution of the number of images per country. The vertical axis shows the 243 countries of the world from which we have crawled the images, while the horizontal axis represents the number of images for each country.

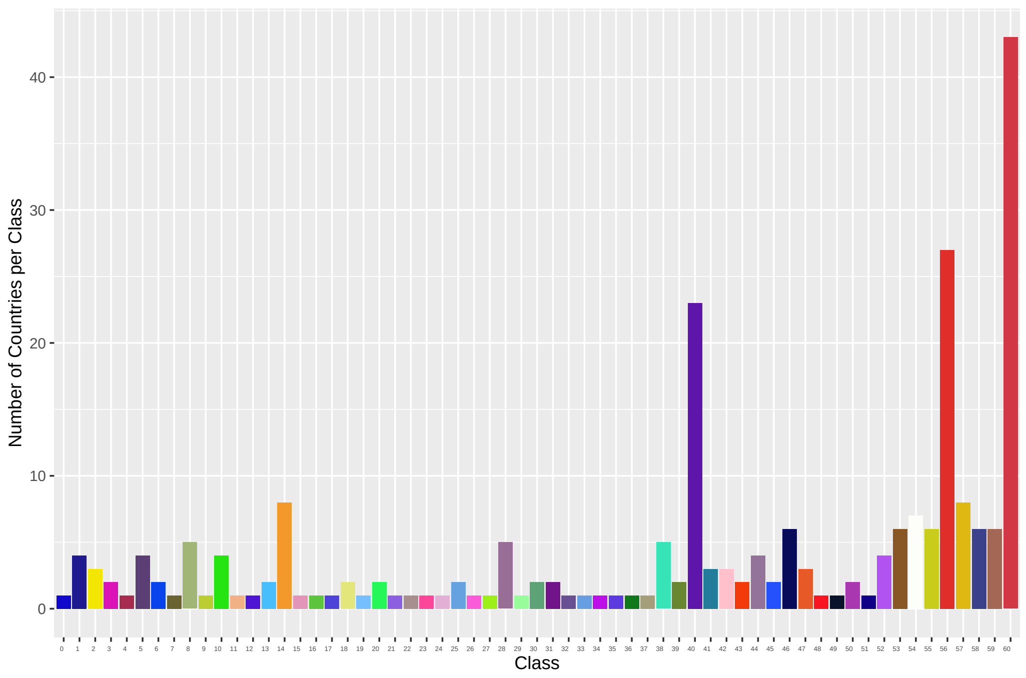


Figure 7.7: Histogram with the distribution of the number of countries for each class. The x axis shows the 61 classes and the y axis reports the number of countries for each class.

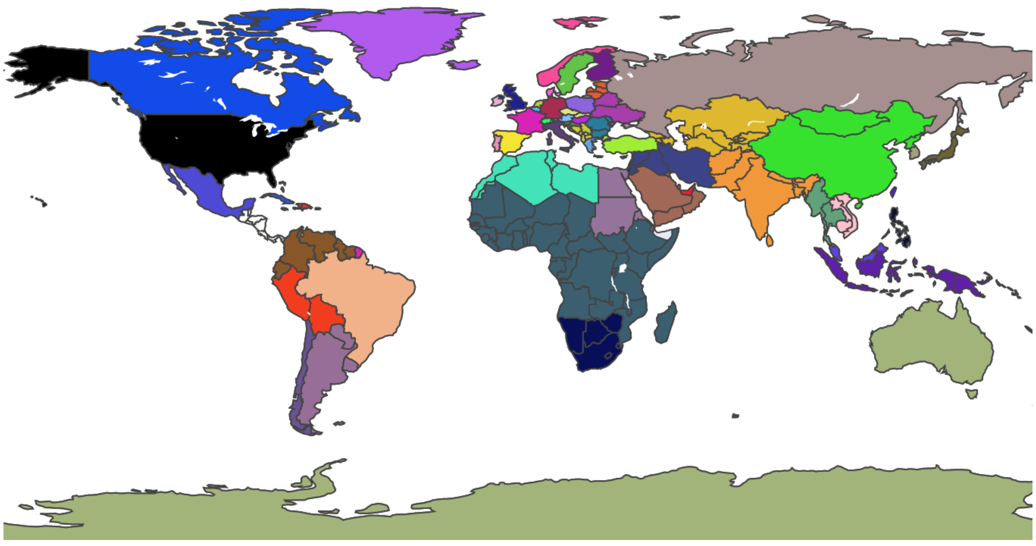


Figure 7.8: World coloured map of the 61 country classes we have used in our experiments.

during training. With regard to the second criterion, instead, we have merged countries according to semantic similarity (for example considering the architectural or linguistic similarities). For instance, according to the above criteria, we have grouped Vatican City with Italy and Egypt with Sudan. It goes without saying, that the way we have grouped the images into 61 countries is somewhat arbitrary and other choices can be made by researchers that will use the VIPPGeo dataset for their studies.

In the end, the class containing the largest number of images is the one with US images, with 829,345 images. The smallest class is a class containing the United Arab Emirates, with only 6,134 images. The histogram with the distribution of the number of images, across the original 243 countries is shown in Figure 7.6, while the distribution over the 61 merged classes is reported in Figure 7.5. As can be seen, grouping the countries into 61 classes results in a more balanced distribution (still far from uniform). The number of countries for each of the 61 classes is reported in Figure 7.7, while the geographical distribution of the countries across the 61 classes is shown in Figure 7.8.

7.3 DNN-based classifier

The country classifier we have built is based on the Resnet–101 architecture [134]. Such architecture has been proven to be extremely successful for the purpose of image classification and has been applied widely in several fields. In Figure 7.9, we show the overall architecture of the network we have used. Given that the size of the network input is fixed (224 × 224, 3-band images), and given that the images to be classified have very diverse dimensions, we devised a strategy, somewhat similar to that used in [2], to analyse the entire image content without changing the aspect ratio of the images, since this could impair the classification. The approach we adopted is sketched in Figure 7.10. The image is first resized in such a way that the lower dimension is equal to 256. Then, we extract 5 crops of size 224 × 224: 4 in the corners and 1 in the centre of the image. The Resnet–101 classifier is then applied to each crop and the 5 results are fused into a single

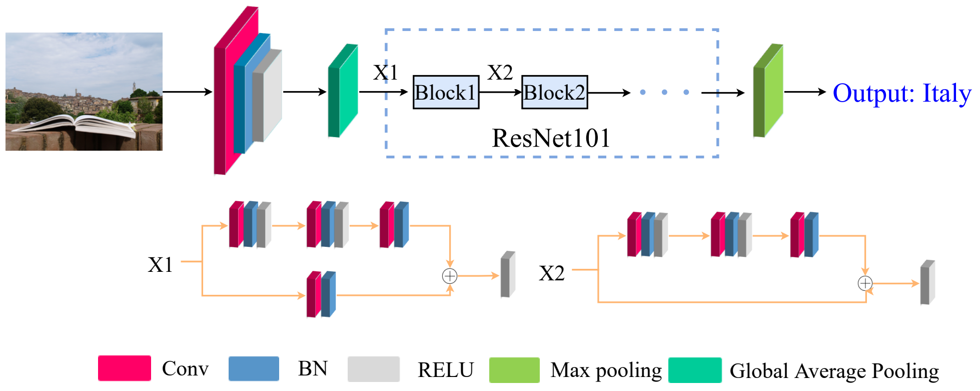


Figure 7.9: The overall architectural workflow. On the top of the image we show the overall workflow, with the input image, the central Resnet-101 architecture. At the bottom of the figure, we display the two types of blocks of Resnet-101.

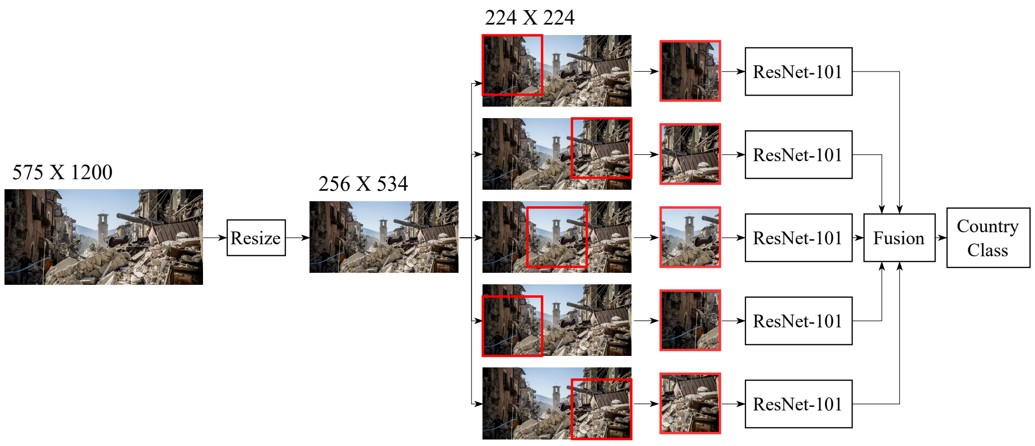


Figure 7.10: The input image is first resized, then split into 5 crops. Each crop is analyzed independently. The final result is obtained by fusing the results obtained on each crop.

score. We have tried several fusing strategies (see Section 7.4.1) and found that the best performance is obtained by averaging the scores obtained on the five crops as follows:

$$Y = \frac{1}{5}\sum_{i=1}^{5} y_i, \tag{7.1}$$

where y_i is the output vector from the network for each crop.

The network analyzing the crops was trained for a total of 25 epochs, starting from a model pre-trained on ImageNet. The batch size was set to 320 images. The network was trained by using cross-entropy loss and the Stochastic Gradient Descend (SGD) optimizer with a learning rate of $1 \cdot 10^{-2}$ a momentum of 0.9, and a weight decay of $1 \cdot 10^{-4}$. All of the experiments have been performed using Pytorch [109] framework on a workstation equipped with one AMD Ryzen 9 3900X 12-Core CPU and one NVIDIA TITAN RTX 24GB.

To avoid overfitting and to facilitate learning robust features, we have applied geometric augmentation. First, we randomly applied horizontal and vertical flipping. After that, we took a random crop covering at least 2/3 of the image area. Then, we resized the crop to 224 $\times$ 224. Finally, The images were normalized to bring the mean pixel values and the standard deviation equal to those of the Imagenet dataset[2].

The training was carried out on the VIPPGeo dataset, which we have split into training, validation and testing subsets, with percentages of 96%, 2% and 2% respectively. Splitting was applied to each country separately.

Even if we did our best to balance the number of images in the 61 country classes, the balance is far from perfect. Therefore, we had to take proper countermeasures to avoid that the network decision being biased towards the most represented countries. We have done so by customizing the cross entropy loss used during training. In particular, we have

[2]This is common practice when using models pre-trained on Imagenet.

followed [165] and weighted the terms in the loss function according to the inverse of the square root of the class frequency, as described in the following equation:

$$Loss = -\sum_{y \in \mathcal{D}_{tr}} \sum_{i=1}^{N} \frac{1}{\sqrt{n_i}} y^i log(\bar{y}^i), \tag{7.2}$$

where N is the number of classes, n_i is the total number of images in class i, y^i is the ground truth of image y, $\bar{y}^i$ is the network score assigned to each class in N for image y, and $\mathcal{D}_{tr}$ is the training set.

7.4 Experiments and Results

In this section, we describe the results of the experiments we have performed to assess the effectiveness of the classifier trained on the VIPPGeo dataset. We also report some examples to get some hints about the plausibility of the analysis carried out by the classifier.

The second row of Table 7.1 shows the accuracy of our network when tested over the images taken from the Im2GPS [132], the Im2GPS-3k dataset [127], and the test subset of VIPPGeo (consisting of 76,269 images). The same filtering strategies described in Section 7.2.1 were applied to the images in the Im2GPS and Im2GPS-3k datasets. In this way, the 270 images in Im2GPS were lowered down to 102 and the 3,000 images in Im2gps3K were reduced to 1,001. All test images represent urban views in order to have a fair comparison with the VIPPGeo dataset. Top-1, 3, 5, 10 and balanced accuracies [166] are reported in the table. Results regarding Top-3 accuracy and balanced accuracy are particularly good, taking into account the difficulty of the task and the diversity of the conditions under which the network was tested. Top-1 accuracy is also very good, considering a large number of classes.

Test sets	*Im2gps - (102 images)*					*Im2gps3k - (1001 images)*					*VIPPGeo test set - (76269 images)*				
	Top1	Top3	Top5	Top10	Bal	Top1	Top3	Top5	Top10	Bal	Top1	Top3	Top5	Top10	Bal
GeoEstimation SOTA [2]	0.80	0.88	0.88	0.91	0.75	0.66	0.76	0.80	0.85	0.51	0.41	0.52	0.57	0.65	0.31
Developed method	**0.80**	**0.91**	**0.97**	**0.98**	**0.80**	**0.67**	**0.83**	**0.88**	**0.94**	**0.56**	**0.66**	**0.81**	**0.87**	**0.93**	**0.60**

Table 7.1: Accuracy of our method and the method in [2] for country classification.

We have compared the results of our method with those achieved by the state-of-the-art method in [2]. Such a method returns the estimation of the GPS location where the image has been shot. For a fair comparison, then, we have transformed the geo-coordinates into a country class by mapping the output GPS location returned by [2] to the corresponding country and then to the country class. As shown in the table, [2] performs well on the Img2ps–102 and the Img2ps3k–1001 datasets, but does not generalize well to the images in the VIPPGeo dataset. On the contrary, our method provides comparable or even better performance than [2] even when tested on images belonging to the datasets used to train [2].

Test sets	*Imgps - (102 images)*					*Imgps3k - (1001 images)*					*VIPPGeo test set - (76269 images)*				
	Top1	Top3	Top5	Top10	Bal	Top1	Top3	Top5	Top10	Bal	Top1	Top3	Top5	Top10	Bal
GeoEstimation-Trained on VIPPGeo	0.75	0.82	0.84	0.88	0.72	0.60	0.72	0.77	0.82	0.50	0.63	0.73	0.76	0.81	0.55

Table 7.2: Accuracy of the method proposed in [2] trained on VIPPGeo dataset.

As a further investigation, we have also trained the network described in [2] on the VIPPGeo dataset obtaining the results reported in Table 7.2. As we can see, by training the system with the large number of images made available by VIPPGeo, we can boost the performance of [2] significantly, thus confirming the importance of the work we made to build the VIPPGeo dataset. Still, upon comparison with the results in Table 7.1, the superior performance of our newly developed method is confirmed.

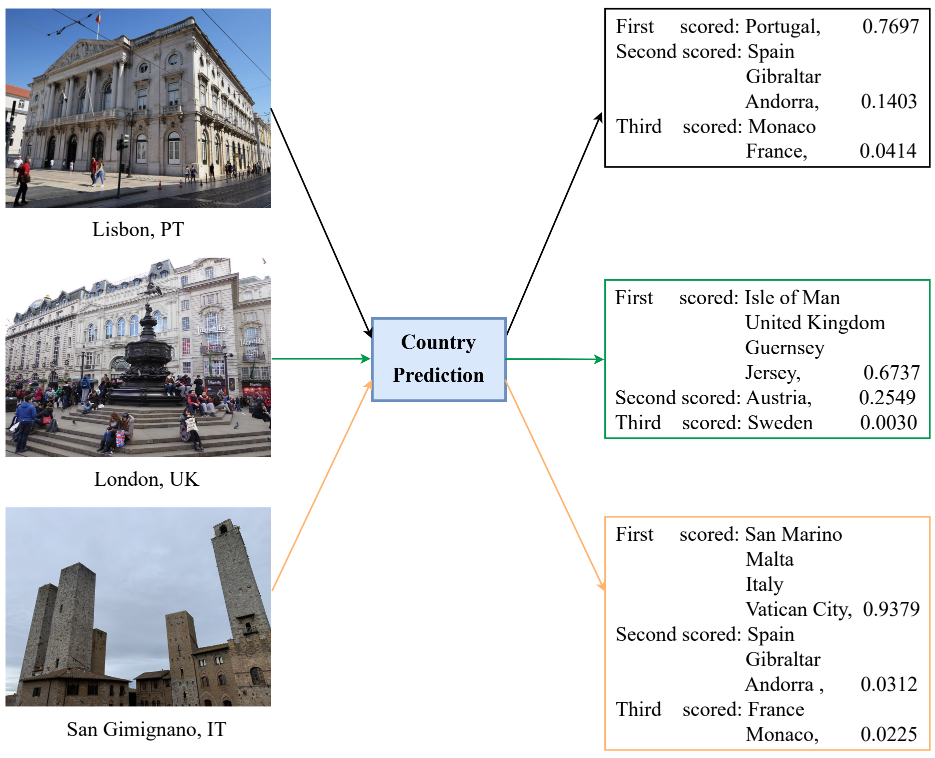

Lisbon, PT
London, UK
San Gimignano, IT
Country Prediction
First scored: Portugal, 0.7697
Second scored: Spain Gibraltar Andorra, 0.1403
Third scored: Monaco France, 0.0414
First scored: Isle of Man United Kingdom Guernsey Jersey, 0.6737
Second scored: Austria, 0.2549
Third scored: Sweden 0.0030
First scored: San Marino Malta Italy Vatican City, 0.9379
Second scored: Spain Gibraltar Andorra , 0.0312
Third scored: France Monaco, 0.0225

Figure 7.11: Some examples of predictions made by our system.

Figure 7.11 shows some examples of input images and the corresponding classes predicted by our network.

7.4.1 Ablation study

To understand the impact of the way we fuse the results obtained on the five crops on the final accuracy, we have carried out some experiments by adopting different fusion strategies. Specifically, we have considered two alternative ways to take into account

Test sets	*Imgps - (102 images)*					*Imgps3k - (1001 images)*					*VIPPGeo test set - (76269 images)*				
	Top1	Top3	Top5	Top10	Bal	Top1	Top3	Top5	Top10	Bal	Top1	Top3	Top5	Top10	Bal
Upper left crop	0.73	0.87	0.93	0.96	0.69	0.59	0.78	0.85	0.92	0.50	0.57	0.76	0.83	0.90	0.50
Upper right crop	0.73	0.89	0.94	0.96	0.74	0.60	0.78	0.84	0.91	0.50	0.57	0.75	0.82	0.90	0.49
Lower left crop	0.69	0.81	0.90	0.98	0.64	0.58	0.76	0.84	0.91	0.49	0.60	0.77	0.83	0.91	0.53
Lower right crop	0.68	0.86	0.89	0.96	0.67	0.59	0.77	0.84	0.91	0.47	0.59	0.77	0.83	0.91	0.52
Central crop	0.72	0.89	0.95	0.98	0.69	0.60	0.79	0.84	0.91	0.51	0.60	0.78	0.84	0.91	0.54
Max-based fusion	0.75	0.90	**0.97**	0.98	0.78	0.66	0.82	**0.88**	**0.94**	0.55	0.65	0.80	0.86	0.92	0.59
Resize to 224	0.72	**0.92**	0.95	**0.99**	0.65	0.62	0.81	0.87	0.93	0.51	0.64	0.80	0.86	0.92	**0.60**
Developed method - average	**0.80**	0.91	**0.97**	0.98	**0.80**	**0.67**	**0.83**	**0.88**	**0.94**	**0.56**	**0.66**	**0.81**	**0.87**	**0.93**	**0.60**

Table 7.3: Accuracy when different strategies are used to feed the Resnet-101 classifier with images of fixed size (224 × 224).

the entire image content, despite the fixed size of the network input. The first strategy takes the maximum output resulting from the five crops and the second resizes the input image to 224 × 224 (thus modifying the aspect ratio) and classifies the image directly in one single step. The last three rows of Table 7.3 show the results we got. As we can see, averaging over the five crops gives slightly better performance compared to the other strategies, even if the advantage is a minor one.

We have also verified that fusing the results of different crops is indeed helpful. We have done so by looking at the accuracy obtained by classifying the images based on the result of one single crop. The results we got, reported in the first rows of Table 7.3, show clearly that classifying the images based on the partial content contained in one single crop provides significantly worse results.

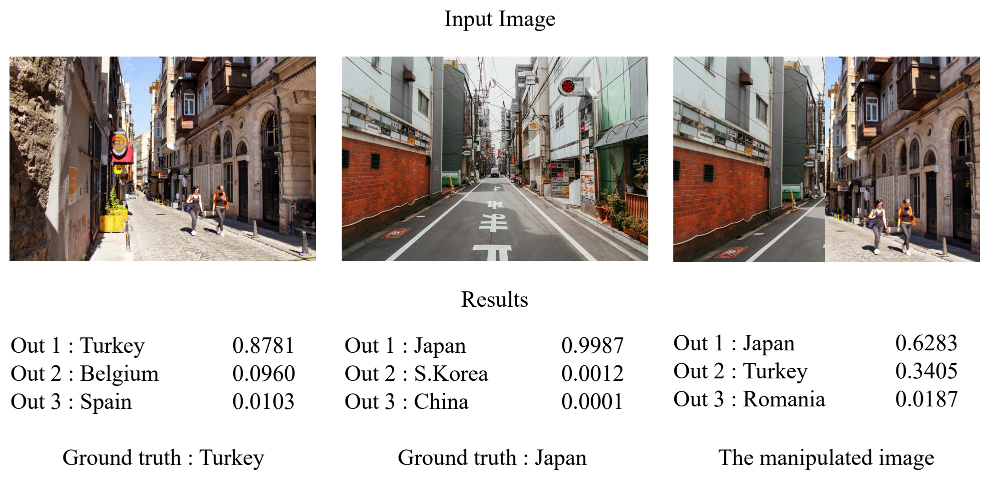
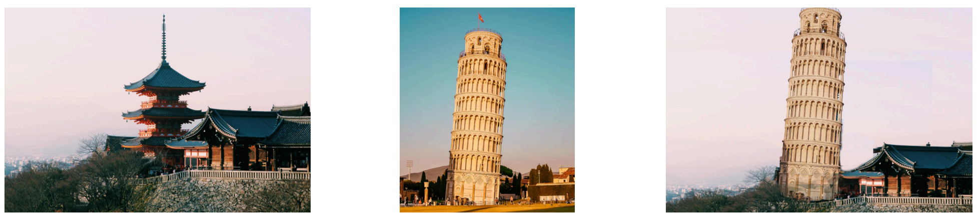


Figure 7.12: An example showing the result provided by the network when two images representing different countries are put together to form a composite image.

7.4.2 Plausibility analysis

To get some insights about the plausibility of the analysis carried out by the network, and make sure that the classification is based on the semantic content of the images, we have fed the network with images wherein an iconic monument of one country is pasted into a picture taken from a different country. We have also fed the network with images composed of two sub-images taken from different countries. In these cases, we expect the network to be confused, but still capable of assigning the highest probabilities to the two countries that contributed to forming the image. This is indeed the case in many situations, as shown in the examples reported in Figures 7.12 and 7.13. In all of the cases, the countries which contributed to forming the picture are ranked in the first position.

Input Image



Results

Top 1 : Japan	0.9907	Top 1 : Italy	0.9441	Top 1 : Italy	0.6137
Top 2 : S.Korea	0.0055	Top 2 : Germany	0.0168	Top 2 : S.Korea	0.1868
Top 3 : China	0.0015	Top 3 : Austria	0.0114	Top 3 : Japan	0.0686
Ground truth : Japan		Ground truth : Italy		The manipulated image	

Figure 7.13: Insertion of an iconic monument of one country into a picture representing a different country. The network identifies the two contributing countries in the first positions.

7.5 Conclusions

In this chapter, we have presented two contributions toward the development of an automatic method for Country Recognition. Firstly, we present a new benchmark dataset, which we named VIPPGeo. VIPPGeo is composed of almost 4 million urban images, crawled from 3 public sources (Flickr, Unsplash and Mapillary). The dataset was collected to overcome the lack of benchmarks on which DL methods can be trained. This, in fact, represents one of the main initial issues regarding the possibility of developing such automatic methodologies. Secondly, we present a new classification method for automatic Country Recognition. Differently from previous state-of-the-art methods, we have cast the problem as a classification one, instead of the more “classical” approach in which the GPS coordinates of the image are first estimated, and then used to trace back to the country where the image was taken. In particular, we have formalized the geolocalization Country Recognition problem as a multi-class classification task, where each class is formed by a group of countries sharing similar geographic and architectural similarities. The DL model we have developed is based on the ResNet–101 architecture. Moreover, we have performed ablation studies, in order to evaluate the impact of the fusion of the 5 crops' decisions on the final performance. The plausibility of the network behaviour was also investigated, testing the network with modified input images, where iconic monuments were used as input to the network, as well as landmarks added to input pictures. Our experiments have shown promising performance, both on the new VIPPGeo dataset and other benchmark datasets, outperforming state-of-the-art results.

Chapter 8

City Verification

Machines Take Me by Surprise with Great Frequency.
Alan Turing

This chapter addresses the problem of recognizing the city where an image has been shot. Due to the vast number of cities in the world, we cast the problem as a verification problem, whereby the system has to decide whether a certain image has been taken in a given city or not. In particular, we present a system that given a query image and a small set of images taken in a target city, decides if the query image has been shot in the target city or not. To allow the system to handle the case of images taken in cities that have not been used during training, we use a Siamese network based on a Vision Transformer as a backbone. The experiments we have carried out prove the validity of the proposed system which outperforms solutions based on state-of-the-art techniques, even in the challenging case of images shot in different cities of the same country.

8.1 Introduction

In the previous chapter, we focused on the identification of the country of an image. In this chapter, our attention shifts towards a distinct challenge: identifying the specific city where an image was captured. On the one hand, recognizing the country where an image was taken often lacks the required precision for precise localization. On the other hand, estimating GPS geo-coordinates proves to be an especially challenging task. However, identifying the city of origin of an image is often more crucial and adequate for specific tasks compared to the precise estimation of GPS coordinates or the Country Recognition task.

As a matter of fact, city recognition is an extremely challenging task, due to the huge number of cities worldwide and the high similarities between many urban environments belonging to the same country. To address this challenge, we shift from the inference and classification approaches adopted in the literature so far and set the city recognition problem as a verification task. In particular, we present a system based on a Siamese network, whose goal is to evaluate if a source image has been taken in a target city, assuming that a small set of images of the target city is available.

The developed system employs a Siamese network with a Vision Transformers (ViT) backbone. The ViT architecture has been proven to be extremely successful for image classification and has been widely applied in several fields. Our system takes as inputs both a query image and a small set of pictures taken in a target city and decides if the query image has been shot in the target city or not. The system is complemented in both the training and test phases by the GeoVIPP country classifier described in the previous chapter, to rule out the possibility that images taken in different countries are judged to belong to the same city. Moreover, we have utilized the Places365-CNNs model [163] to measure the semantic similarity between the input image and the reference images of the target city, weighting more heavily the similarity (or dissimilarity) between images having similar semantic content.

To train and test our verification system, we have constructed a dataset extracted from the VIPPGeo dataset introduced in the previous chapter. Overall, the City Verification dataset we have built consists of $120,909$ high-quality urban images from 19 cities all around the world.

We have used the dataset to run several experiments and demonstrate the effectiveness of our system for both closed and open set scenarios. In the former case, the images used in the testing phase belong to cities that were also used in the training phase. In the open set scenario, instead, the to-be-verified images do not belong to any of the cities used in the training phase. In both cases, we have also considered the difficult case of images taken in cities of the same country. We have compared the performances of our system with those of a verification system, built on top of a geolocalization network, providing an estimate of the image geo-coordinates and using them to decide if the query image belongs to the target city or not. The results of the experiments show that our system outperforms the system based on state-of-the-art geo-coordinates estimation in both the closed and the open set scenarios.

The rest of the chapter is organized as follows. In Section 8.2, we briefly introduce ViT and Siamese Neural Networks. In Section 8.3, we present the City Verification dataset and the developed ViT-based Siamese architecture. In Section 8.4, we describe the experimental setting and the results we got. Finally, we draw our conclusions and outline some perspectives for future research in Section 8.6.

8.2 ViT and Siamese Neural Networks

In this section, we briefly review ViT and Siamese Neural Networks.

8.2.1 Vision Transformers

Transformers were first described in the work by Vaswani et al. [167] based on attention mechanism in natural language processing tasks, e.g., machine translation and question answering. The basic building block of a transformer consists of the multi-head self-attention mechanism that exploits a deep relationship among the elements of embedding words. The ViTs, a variant of transformer targeting computer vision tasks, was first presented in [74] for image classification, by taking a sequence of image blocks as input. Thanks to their outstanding performance, more and more researchers are proposing transformer-based models for improving a wide range of visual tasks, including object detection [168], semantic segmentation [169, 170], image processing [171], and video understanding [172]. Traditional CNNs have gradually been substituted by transformers as the preferred model in the field of computer vision, with several models proposed such as Swinformer, BERT [173], and BEVT [174].

In contrast to RNN, transformers are able to focus on the whole sequence, not focusing mainly on short-term dependencies. Moreover, transformers are purely based on the attention mechanisms and their uniqueness consists in an implementation which is optimized for parallelization purposes [175]. As opposed to other approaches, like hard attention [176], which is stochastic in nature and needs Monte Carlo sampling for attention location sampling, transformers scale well to high-complexity models and large-scale datasets. Additionally, pre-trained transformers trained using pretext tasks on large-scale (unlabelled) datasets [167,173] are adopted as the starting point of the training procedure, thus significantly reducing the cost of manual annotations.

8.2.2 Siamese Networks

The Siamese network framework was first proposed by Bromley et al. [131] in 1993 for verification tasks. A basic Siamese network adopts two subnetworks with shared weights as feature extractors. The final decision, then, is made by comparing the outputs of the two subnetworks [177, 178]. Learning knowledge by comparing the features extracted by the two branches instead of directly using labels gives the possibility to learn with unlabeled data and plays an important role in overcoming the limited label issue in real-life applications. In recent years, Siamese architectures attracted increasing attention in addressing various matching problems, such as object tracking [179], image matching [180], image identification [181] and image change detection [182, 183].

In this work, we have combined a ViT model with a Siamese architecture for the City Verification task. Specifically, we have constructed a two-branch network, with ViT as the backbone. The network is then followed by fully connected layers with ReLU activation functions. The final layer of the network uses a Sigmoid activation function to make the final decision.

8.3 Methodology

In this section, we describe the City Verification dataset and the developed City Verification system.

8.3.1 City Verification Dataset

The City Verification dataset was constructed starting from the VIPPGeo dataset. It consists of 120, 909 images from 19 cities worldwide. To obtain the images of each city, we have retrieved all the images shot within a circular area around the city centre. The diameter of the circular area varies from 5 Km to 10 Km, depending on the availability of images in the VIPPGeo dataset for that particular city. We have used a flexible diameter length to ensure that a fixed number of images are collected for each city of the dataset. The City Verification dataset is partitioned into two sets: closed and open sets. The closed set was utilized for training and testing the developed framework, while the open

Table 8.1: Number of images in the City Verification Dataset.

	Training	Validation	Siamese-testing	Verification
Closed Set	100,000	10,000	10,000	1,010
Open Set	0	0	0	1,010

set was employed to evaluate the generalization capability of the system. Notably, the system has not encountered images from the open set during training. We have collected $12,000$ images for each city class in the closed set and 101 images for each city class in the open set.

The closed set portion of the City Verification dataset consists of 10 city classes, with two pairs of cities belonging to the same country. The cities are Amsterdam, Barcelona, Berlin, London, NewYork, Los Angeles, Rome, Milan, Paris and Tokyo. This subset contains a total of $120,000$ images, $100,000$ out of which were used to train the ViT Siamese network, $10,000$ for validation, $10,000$ for testing, and $1,010$ images for verification. The verification subset dimension is reduced by applying restrictive filtering on the test set and we have used it to evaluate the performance of the overall City Verification system, which includes not only the Siamese network but also other auxiliary components. In contrast, we have used the testing subset exclusively to assess the performance of the Siamese network. Further details are given in the following sections.

The open set portion of the dataset consists of 10 city classes, including 9 new cities that were not presented in the closed set, and one city shared with the closed set. The cities are Amman, Istanbul, Mexico City, Singapore, Quebec, Vancouver, Florence, Rome, Rio de Janeiro and Delhi. The open set contains two pairs of cities coming from the same country, with each class containing 101 images. The number of images in the open set is much smaller than that of the closed set since these images were not used for training. The open set was used exclusively for the verification task. Table 8.1 summarizes the characteristics of the City Verification dataset.

8.3.2 City Verification System

Figure 8.1 shows the overall architecture of the City Verification system. The core of the system is formed by a Siamese Network with a ViT backbone. The other building blocks include the country classifier (GeoVIPP) introduced in the previous chapter and the Places365-CNNs model [163] to measure the semantic similarity between images. The verification pipeline of our system is outlined in the following:

1. Given an input image (hereafter referred to as query image) and a claim on the city where the image has been taken (hereafter referred to as claimed city), the image is passed through the GeoVIPP country classifier to obtain the Top-2 country predictions.

2. If the country of the claimed city does not appear in the Top-2 country predictions, the image city claim is not verified, and the image is not passed to the subsequent

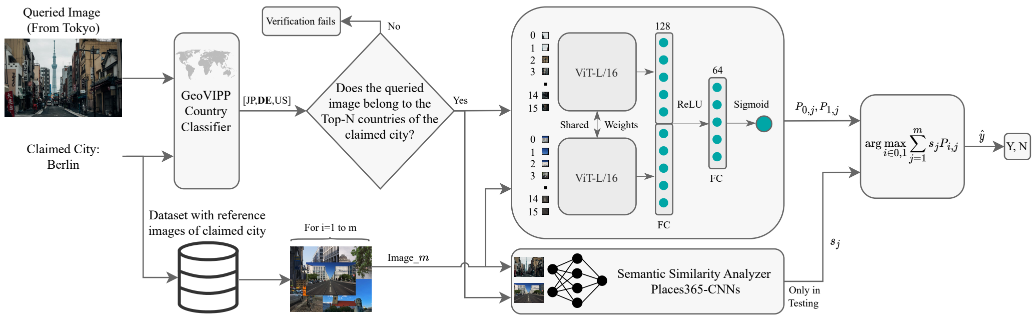

Queried Image (From Tokyo)
Claimed City: Berlin
GeoVIPP Country Classifier
[JP,**DE**,US]
Verification fails
No
Does the queried image belong to the Top-N countries of the claimed city?
Yes
Dataset with reference images of claimed city
For i=1 to m
Image_m
0 1 2 3 14 15
ViT-L/16
Shared Weights
ViT-L/16
128
FC
ReLU
64
FC
Sigmoid
$P_{0,j}, P_{1,j}$
$\arg\max_{i\in 0,1} \sum_{j=1}^{m} s_j P_{i,j}$
$\hat{y}$
Y, N
s_j
Semantic Similarity Analyzer Places365-CNNs
Only in Testing

Figure 8.1: The figure illustrates the overall workflow of the developed architecture. For more verification examples refer to Section 8.5. The code implementing the City Verification systems is publicly available at the following link: `https://github.com/alamayreh/city_verifier`.

steps of the system.

3. If the country of the claimed city appears in the Top-2 country predictions, the image is paired with m reference images taken in the claimed city for verification.

4. The image pairs (composed by the query image and the m reference images from the claimed city) are fed to the Siamese network.

5. The semantic similarity between the two images in each pair is calculated using the Places365-CNNs model [163] (refer to Section 8.3.2 for the details).

6. The Siamese network outputs a probability between 0 and 1 for each image pair, with 0 meaning that the images belong to the same city, and 1 that they belong to different cities.

7. The scores given by the Siamese network are weighted according to the image similarities evaluated by the Places365-CNN network and summed together. Eventually, the weighted score is thresholded to verify if the query city has been shot in the claimed city or not.

In the following sections, all the components of the verification system are described in detail, starting with the Siamese network which represents the backbone of the system.

ViT-based Siamese Network

For each branch of the Siamese network, we have used the ViT-L/16 variant of the Vision Transformer model provided by [74]. We have set the last layer of the ViT model to have 64 output units. Afterwards, we concatenated the output of both networks to form a single-layer feed-forward network with a size of 128 units. The concatenated output is then passed through a Rectified Linear Unit (ReLU) activation function. Then, the output is forwarded to a layer with 64 units, and finally to a Sigmoid activation function.

A Sigmoid output equal to or close to 0 indicates that the image pairs come from the same city, while a value close to 1 indicates that the image pairs come from different cities.

Given that the size of the ViT input is fixed (224×224, $3-\textit{band}$ images with 16 patches), and given that the images to be verified have very different dimensions, we have adopted a strategy to analyse the entire image content without changing the aspect ratio of the images, since this could affect the performance of the system. During the validation, testing and verification phases, the query image is first resized in such a way that the lower dimension (either width or height) is equal to 256. Then, a center crop of size 224×224 is taken from the resized image. The pre-processing step ensures that all images fed to the Siamese network are consistent in size and orientation.

Furthermore, during the training phase, we applied geometric augmentation to prevent overfitting and promote the learning of robust features. The images are first resized so that the lower dimension is equal to 256. Then, we randomly applied horizontal and vertical flipping to the training images. Subsequently, we took a random crop covering at least 2/3 of the image area. Eventually, we resized the crop to 224×224.

In all phases (i.e., training, validation, testing, and verification), the images were normalized to bring the mean pixel values and the standard deviation to be equal to those of the Imagenet dataset[1].

ViT Siamese Training

Based on the number of images per class in the City Verification dataset, which is equal to $10,000$ images, we can form a very large number of possible image pairs that we can use to train the Siamese network. Specifically, the total number of image pairs that we can form is in the order of 5×10^9.

To account for this large number of image pairs and to force the Siamese network to learn robust and discriminative features, without being biased towards any specific city class, we have adopted a proper sampling strategy to be used during training. The strategy ensures that for each training epoch, we have a balanced and representative number of positive and negative image pairs. In addition, the sampling strategy ensures that the negative pairs are evenly distributed across different cities, preventing any bias towards specific cities. During each epoch, a total of 16384 image pairs are sampled to train the Siamese network. The large number of image pairs sampled at each epoch provides a diverse and representative sample of the City Verification dataset.

We have trained the Siamese network for 50 epochs, starting from a ViT model pre-trained on ImageNet [108]. The batch size was set to 32 image pairs. The network was trained using Binary Cross Entropy loss (BCE) and the Stochastic Gradient Descend (SGD) optimizer with a learning rate of $1 \cdot 10^{-2}$ a momentum of 0.9, and a weight decay of $1 \cdot 10^{-4}$. We used the loss function value on the validation set to select the best model for testing and verification. All the experiments have been performed using PyTorch [109] as a framework on a workstation equipped with one Intel(R) Xeon(R) E5-2620 24-Core CPU and four NVIDIA Quadro M6000 12GB GPU.

[1]This is common practice when using models pre-trained on Imagenet.

GeoVIPP Country Classifier

We have exploited the GeoVIPP classifier during training and testing. During training, while sampling the negative pairs to be used within each epoch, we have imposed a specific country-plausibility condition to hold for at least 50% of the pairs. The condition can be expressed as follows. Suppose we have sampled Image A and Image B as candidate negative pairs. Before feeding the pair to train the Siamese network, we pass each image through the GeoVIPP country classifier, and we consider the negative image pair *(A, B)* valid if at least one of the Top-2 country predictions of Image A is equal to one of the Top-2 country predictions of Image B. Adding this condition helps to ensure that the negative pairs used to train the Siamese network consist of challenging pairs to verify, in the sense that these images are coming from different cities but according to the country classifier, these images may potentially be from the same country. In this way, we also mimic the operating conditions enforced at test time, when the verification automatically fails if the country of the claimed city is not in the Top-2 countries estimated by the GeoVIPP country classifier on the query image.

In the testing phase, the role of the GeoVIPP country classifier is to let the verification fail if the country of the claimed city does not appear in the Top-2 country predictions made by GeoVIPP on the query image.

Semantic Similarity Analyzer

As shown in Figure 8.1, the final decision is made by accumulating the output of the Siamese network on all the image pairs formed by the query image and the reference images of the claimed city. The output of the Siamese network is weighted according to the semantic similarity between the two images in the pair. The rationale behind this choice is that the output of the Siamese network is more reliable when the input images represent similar scenes. To do so, we have exploited the Places365-CNNs classification model [163]. Given an image, Places365 classifies it into one of 365 scenery categories, for example, house, street, river, etc. We have reduced the output vector dimensions of the Places365 classification system from 365 to 16 by utilizing the scene hierarchy matrix provided in the same work. The hierarchy categories contain 16 major classes including shopping and dining, workplace (office building, factory, lab, etc.), transportation (vehicle interiors, stations, etc.), sports and leisure, home or hotel, cultural (art, education, religion, military, law, politics, etc.), (water, ice, snow), (mountains, hills, desert, sky), (forest, field, jungle), (man-made elements), transportation (roads, parking, bridges, boats, airports, etc.), cultural or historical building/place (military, religious), (sports fields, parks, leisure spaces), industrial and construction, (houses, cabins, gardens, and farms), and (commercial buildings, shops, markets, cities, and towns).

The reduction results in a 16-long vector, whose components give the probability that the scene shown in the input image belongs to the 16 hierarchy scene classes considered by Places365-CNN. Given such a vector, the semantic similarity s between the images of each image pair is computed as follows:

$$s = \frac{u \cdot v}{\|u\|_2 \|v\|_2} \tag{8.1}$$

where u and v are the output probability vectors provided by Places365 model for each image. Then, we used the similarity to weigh the output of the Siamese network. In particular, the final decision is made as:

$$\hat{y} = \arg\max_{i\in 0,1} \sum_{j=1}^{m} s_j P_{i,j} \tag{8.2}$$

where s is the semantic similarity between the two images in each pair, m is the total number of image pairs, P_0 is the probability that the images in the pair come from the same city, and P_1 that the images come from a different city. P_1 is equal to the Sigmoid output of the Siamese Network, while P_0 is equal to $(1 - P_1)$.

8.4 Experiments and Results

We have conducted several experiments to evaluate the performance of the City Verifier. We have considered different scenarios, including open and closed datasets, to assess the effectiveness of the system under various conditions. We have also conducted an ablation study to understand the impact of the GeoVIPP classifier on the accuracy. Finally, we compared our results with a verification system built on top of a state-of-the-art Image Geolocalization system.

8.4.1 Siamese Network Training

To start with, we have trained the Siamese ViT model using image pairs sampled from the $100,000$ images of the training set. The model is continuously evaluated on $64,000$ image pairs sampled from the $10,000$ images of the validation set. We have selected the best model based on the lowest loss value achieved by the trained models on the validation set. Then, the best model is used in the test and verification phases.

Upon training, we evaluated the performance of the Siamese ViT model on $64,000$ image pairs sampled from the $10,000$ images of the Siamese-testing set (see Table 8.1 for the dataset details). In the following, we present the results of the evaluation summarizing them in a confusion matrix (see Figure 8.2). The matrix indicates that the Siamese Network has an adequate capability to detect if two images belong to the same city. Even if the performance of the Siamese network on single image pairs may appear insufficient, this is good enough to let the overall verification system work, when the Siamese network is applied to all the images in the reference dataset of the claimed city, and when it is used in conjunction with the country classifier which contributes significantly to discard negative image pairs, when they belong to cities of different countries. This aligns with our goal, to maximize the Siamese network's ability to detect images from the same city and to rely on the GeoVIPP classifier to discard images that do not belong to the country of the claimed city.

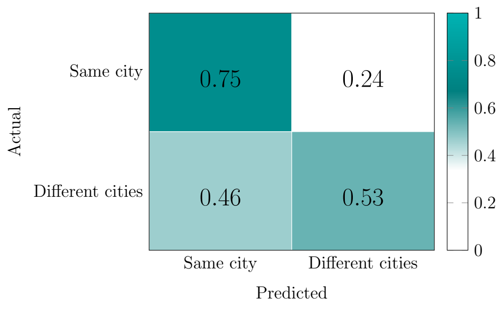


Figure 8.2: Confusion matrix showing the performance of the Siamese network in determining positive and negative pairs on 64, 000 image pairs sampled from the Siamese-testing set.

8.4.2 Verification Results - Closed Set

With regard to the verification task, which we remind is the final goal of our system, the results we got on images representing cities belonging to the closed set are shown in Table 8.2. The experiments have been conducted on the verification set, including 101 images per city class. As mentioned in the procedure outlined in Section 8.3.2, we paired each image from the claimed city with the 101 images of the reference dataset of the claimed city. The claimed cities are given in the first column of the table, while the ground truth of the query cities is shown in the first row of the table.

For cases where the claimed city and the ground truth of the queried cities are the same (on the diagonal), we held out one image from the 101 images of the claimed city

Table 8.2: Results on the closed Set. In each cell, the value on the right shows the total number of images that have successfully passed the GeoVIPP classifier test. The value on the left indicates the fraction of images that passed the overall verification procedure. The overall accuracy is 0.82 for positive pairs (True Positive rate) and 0.96 for the negative pairs (off-diagonal elements, True Negative rate). The blue cells refer to images from the same cities, while the grey cells indicate the results obtained on different cities of the same country.

City	Amsterdam	Barcelona	Berlin	London	New York	L.Angeles	Rome	Milan	Paris	Tokyo
Amsterdam	0.89 - 98	0.00 - 00	0.02 - 15	0.07 - 24	0.00 - 04	0.00 - 04	0.00 - 01	0.00 - 01	0.01 - 09	0.00 - 00
Barcelona	0.00 - 01	0.83 - 90	0.03 - 06	0.04 - 05	0.01 - 17	0.14 - 17	0.00 - 19	0.15 - 19	0.21 - 26	0.00 - 02
Berlin	0.00 - 04	0.00 - 03	0.89 - 93	0.04 - 07	0.00 - 07	0.04 - 07	0.00 - 02	0.01 - 02	0.11 - 12	0.00 - 03
London	0.04 - 07	0.01 - 02	0.03 - 08	0.71 - 96	0.06 - 24	0.13 - 24	0.00 - 03	0.02 - 03	0.15 - 19	0.01 - 03
NewYork	0.00 - 01	0.00 - 00	0.00 - 05	0.00 - 06	0.91 - 99	0.04 - 99	0.00 - 01	0.00 - 01	0.00 - 02	0.03 - 04
L.Angeles	0.00 - 02	0.12 - 14	0.00 - 01	0.00 - 01	0.10 - 88	0.74 - 88	0.00 - 01	0.00 - 01	0.00 - 02	0.03 - 06
Rome	0.00 - 00	0.05 - 32	0.00 - 01	0.00 - 02	0.00 - 02	0.00 - 02	0.85 - 98	0.09 - 98	0.00 - 23	0.00 - 00
Milan	0.00 - 03	0.15 - 19	0.07 - 11	0.04 - 06	0.03 - 13	0.08 - 13	0.00 - 78	0.70 - 78	0.10 - 12	0.00 - 01
Paris	0.00 - 01	0.16 - 18	0.08 - 10	0.18 - 22	0.00 - 05	0.02 - 05	0.00 - 07	0.06 - 07	0.95 - 99	0.00 - 00
Tokyo	0.00 - 01	0.00 - 02	0.00 - 04	0.01 - 04	0.13 - 17	0.01 - 17	0.00 - 00	0.00 - 00	0.00 - 01	0.80 - 96

and paired it with the remaining 100 images. Then, we passed the pairs to the verification system. The process is repeated for each of the 101 images. The fraction of images that passed the entire verification process is reported in the table.

Each cell of the table reports two values. The value on the right shows the total number of images that have successfully passed the GeoVIPP classifier test, that is, the claimed country belongs to the Top-2 countries predicted by the classifier on the query image. The number on the left, instead, shows the overall verification accuracy, that is the fraction of images that passed the verification procedure. The blue cells (on the diagonal) represent the verification results when the claimed city is correct, while the grey cells indicate results when the query and claimed cities are different, but from the same country.

As shown by the table, the verification system demonstrates very good performance, even in the difficult case when the claimed and query cities belong to the same country [2]. As an overall performance metric, we have considered the True Positive (TP) and False Positive (FP) rates, defined as:

$$TP = \frac{1}{C}\sum_{i=1}^{C} a_{i,i} \tag{8.3}$$

$$FP = \frac{1}{C(C-1)}\sum_{i=1}^{C}\sum_{j=1,j\neq i}^{C} a_{i,j}, \tag{8.4}$$

where C is the total number of cities and $a_{i,j}$ are the left values reported in Table 8.2. In particular, TP in Equation (8.3) reports the accuracy of the verification when the query city corresponds to the claimed one (the larger the better), while FP in Equation (8.4) gives the probability that a false claim is verified (the lower the better). The values of TP and FP calculated from the table are 0.82 and 0.03, respectively.

8.4.3 Verification Results - Open Set

We have also evaluated the performance of the verification in an open set scenario where both the query and claimed cities do not belong to the set of cities used during training. The open set portion of the dataset consists of 10 cities, each city class containing 101 images. The only exception to the open set rule is represented by the images of Rome, given that Rome was also included in the closed set portion of the dataset. The reason for this exception is that we wanted to evaluate the accuracy of the system in a mixed scenario where the query (res. claimed) city has been seen during training and the claimed (res. query) city has not.

As illustrated in Table 8.3, the city verifier maintains very good performance also in the open set scenario, especially for cities belonging to different countries. A certain performance drop can be observed when the claimed and query cities are different but belong to the same country. However, even in this difficult case, the verifier maintains a certain capability to recognize if the query image depicts the claimed city or not. Applying Equations (8.3) and (8.4) to Table 8.3,. we now get $TP = 0.72$ and $FP = 0.02$.

[2] In this case, in fact, the country classifier is of no help.

Table 8.3: Verification accuracy results in the open set scenario. In each cell, the value on the right shows the total number of images that have successfully passed the GeoVIPP classifier test. The value on the left indicates the fraction of images that passed the overall verification procedure. The overall accuracy is 0.71 for positive pairs (True Positive rate) and 0.98 for the negative pairs (off-diagonal elements, True Negative rate). The blue cells refer to images from the same cities, while the grey cells indicate the results obtained in different cities of the same country.

City	Amman	Istanbul	Mexico.C	Singapore	Quebec	Vancouver	Florence	Rome	R.Janeiro	Delhi
Amman	0.79 - 97	0.02 - 03	0.00 - 00	0.00 - 00	0.00 - 01	0.00 - 01	0.00 - 04	0.02 - 04	0.00 - 01	0.03 - 05
Istanbul	0.06 - 07	0.79 - 92	0.00 - 01	0.00 - 00	0.00 - 00	0.00 - 00	0.11 - 13	0.00 - 13	0.00 - 01	0.03 - 04
Mexico.C	0.00 - 00	0.00 - 00	0.55 - 71	0.00 - 00	0.06 - 07	0.00 - 07	0.02 - 04	0.00 - 04	0.01 - 06	0.00 - 01
Singapore	0.00 - 00	0.00 - 00	0.00 - 00	0.66-100	0.00 - 00	0.00 - 00	0.00 - 00	0.00 - 00	0.00 - 02	0.01 - 04
Quebec	0.00 - 00	0.00 - 00	0.00 - 01	0.00 - 00	0.70 - 88	0.20 - 88	0.00 - 00	0.00 - 00	0.00 - 00	0.00 - 01
Vancouver	0.00 - 00	0.00 - 00	0.00 - 00	0.00 - 01	0.38 - 86	0.64 - 86	0.00 - 00	0.00 - 00	0.00 - 02	0.00 - 00
Florence	0.00 - 01	0.00 - 01	0.00 - 00	0.00 - 00	0.00 - 00	0.00 - 00	0.77 - 99	0.21 - 99	0.00 - 00	0.01 - 02
Rome	0.00 - 00	0.00 - 02	0.00 - 01	0.00 - 00	0.00 - 00	0.00 - 00	0.20 - 98	0.85 - 98	0.00 - 00	0.00 - 02
R.Janeiro	0.00 - 00	0.00 - 01	0.04 - 08	0.00 - 00	0.00 - 00	0.00 - 00	0.00 - 01	0.00 - 01	0.57 - 81	0.03 - 05
Delhi	0.06 - 09	0.01 - 02	0.00 - 01	0.00 - 00	0.00 - 00	0.00 - 00	0.00 - 01	0.00 - 01	0.00 - 00	0.78 - 99

8.4.4 Ablation Study

As a further investigation, we have conducted an ablation study by removing the GeoVIPP country classifier from the developed City Verification framework. Moreover, we have systematically tuned the GeoVIPP country classifier at different stages of the verification process. Then, we have evaluated its impact on the overall performance. By carefully analyzing the results, we were able to gain insights into the contribution and importance of the GeoVIPP classifier in the developed City Verification framework.

Table 8.4: Impact of GeoVIPP country classifier on verification accuracy at different stages of the verification pipeline.

GeoVIPP		No use	Sampling	Verification	Sampling and Verification
Closed Set	(TP)	0.9079	0.8782	0.8544	**0.8287**
	(FP)	0.2639	0.3410	0.0391	**0.0388**
Open Set	(TP)	0.7772	0.7851	0.7118	**0.7128**
	(FP)	0.3785	0.4832	0.0242	**0.0209**

We have started by removing the GeoVIPP classifier from the system. Then, we measured the verification accuracy on the open and closed City Verification datasets using Equations (8.3) and (8.4). The results are reported in Table 8.4. The column labelled *No use* refers to a case where the GeoVIPP classifier is not used at all, while the *Sampling* column shows the results when the GeoVIPP is only used in the sampling stage of pairs during training of the Siamese network. The *Verification* column represents the case where the GeoVIPP is not used during training, but only during verification. Finally, the last column refers to the case where the GeoVIPP is used in all stages. Upon inspection of the table, we can see that incorporating the GeoVIPP country classifier at different stages of the City Verification leads to a performance increase in both open and closed sets scenarios.

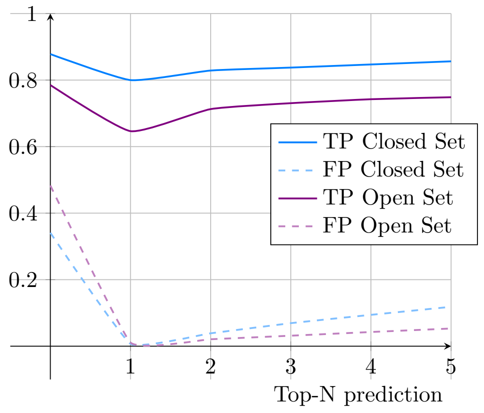


Figure 8.3: Verification accuracy using different Top-N predictions by GeoVIPP country classifier

We have also conducted an experiment to investigate the effect of changing the Top-N prediction of the GeoVIPP classifier used in the verification process. Increasing the value of N means allowing more images to be passed to the City Verification system, with an expected positive effect on the True Positive rate (diagonal values in Tables 8.2 and 8.3) and a negative effect on the False Positive rate (off-diagonal values in Tables 8.2 and 8.3). As we can see from Figure 8.3, choosing N involves finding a trade-off between TP and FP. According to our experiments, a good trade-off is obtained by letting $N = 2$.

8.4.5 Comparisons with the State-of-the-Art

As a last set of experiments, we have compared the results of our system with those obtained by relying on the state-of-the-art ISN method described in [2] and already used as a baseline in chapter 7.

As we said, ISN provides an estimation of the GPS location where the image was captured. To turn the system into a city verifier, we have defined circles around the city centres with different diameters, precisely 25 Km, 50 Km, and Flexible. A city claim is positively verified if the GPS estimation of the query city falls within the predefined radius of the claimed city. The Flexible diameter was adjusted until we achieved the same False Positive rate FP of our system. This yielded a radius equal to 463 Km and 711 Km for the open and closed set cases, respectively. In particular, we have used the Haversine[3] formula to measure the distance between the city centre coordinates and the

[3]The Haversine formula determines the geodesic distance between two points on a sphere given their longitudes and latitudes.

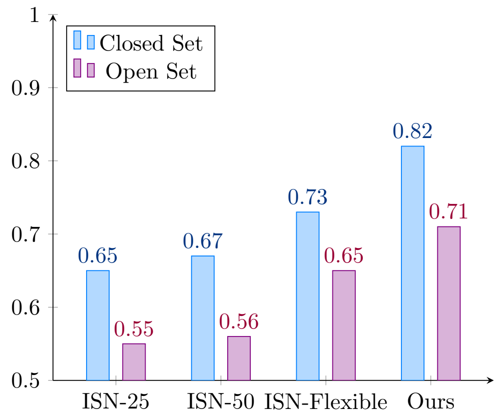


Figure 8.4: True Positive rate comparison with SOTA [2] for various values of proximity circle diameters (from left to right, 25 Km, 50 Km, and Flexible).

estimated geo-coordinates [2]. By using the procedure outlined in Section 8.3.2, we have constructed tables similar to Table 8.2 and Table 8.3 for both the closed and open sets. Then, we applied Equation (8.3) and Equation (8.4) to calculate TP and FP.

Figure 8.4 illustrates the TP rates obtained when the claimed and the query cities are the same. As we can see from the figure, our City Verification system outperforms ISN even when we relax the diameter constraint and enlarge the predefined circle around the city centres. In Figure 8.4, the slightly higher verification accuracy obtained on the closed set compared to the open set case, can be explained by the fact that some cities in the open set have fewer images in the original dataset on which ISN has been trained [2]. With regard to the False Positive rate, we got the results reported in Table 8.5. As shown in the table, both our system and ISN have achieved good performance.

Table 8.5: False Positive rate comparison with SOTA [2] (Equation (8.4)).

	ISN-25	ISN-50	ISN-Flexible	Ours
Closed Set	0.0049	0.0051	0.0388	0.0388
Open Set	0.0009	0.0009	0.0209	0.0209

8.5 Verification Examples

In this section, we illustrate more verification examples for the developed framework. In Figure 8.5, we demonstrate the verification of an image claimed to be from Berlin, but which was instead shot in Tokyo.

In Figure 8.5, we show the voting process using five pairs of images. We report both the semantic similarity s and the Sigmoid output. A higher similarity score means that the content of the images is similar, while a high Sigmoid score indicates that the image

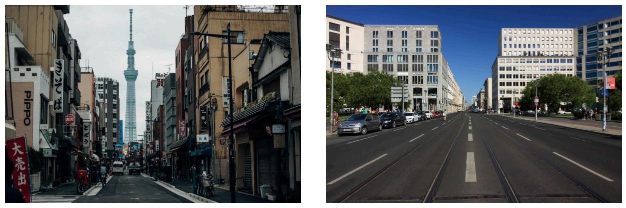
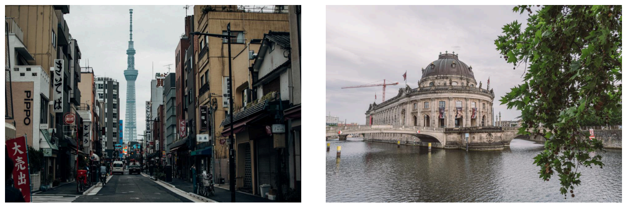
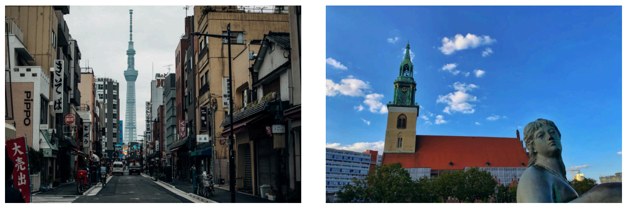
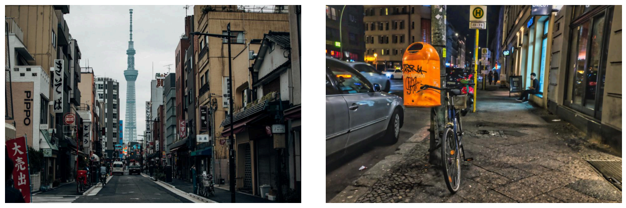
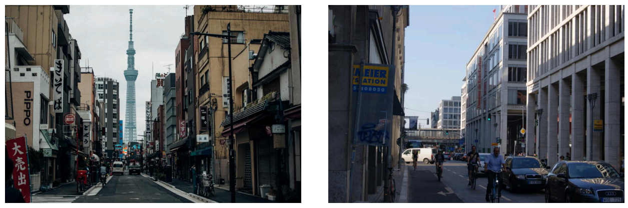
Image Pairs	Similarity	Sigmoid
	0.98	0.96
	0.12	0.90
	0.20	0.76
	0.99	0.32
	0.97	0.88

Figure 8.5: The image on the left is from Tokyo which has been claimed to be from Berlin. The 5 images on the right are from Berlin.

pairs are coming from different cities. For the first, fourth, and fifth pairs, which all depict street views, the similarity score is high. In contrast, the similarity score is low for the second and third pairs, which means these pairs will have less voting power in the final decision. Nevertheless, the Siamese network accurately identifies four image pairs as coming from different cities but misses the fourth pair.

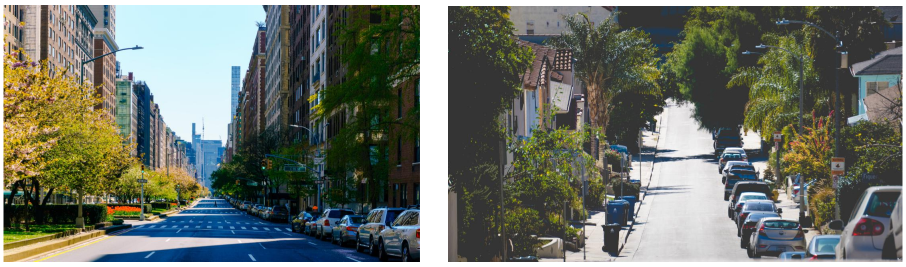

New York. Los Angeles.

Figure 8.6: Similarity: 0.78, Sigmoid output: 0.84

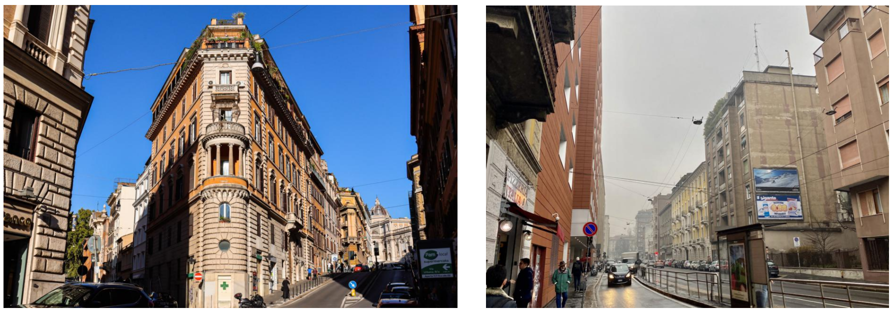

Rome. Milan.

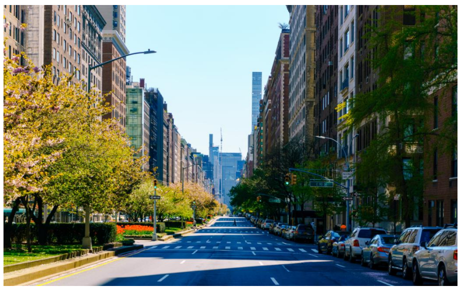
Figure 8.7: Similarity: 0.99, Sigmoid output: 0.60

8.5.1 Closed set

In this section, we illustrate different image pairs sampled from the closed verification dataset. Figure 8.6 and 8.7 show the case when the image pair images are taken in different cities of the same country. Figures 8.8, 8.9, 8.10, and 8.11 show pairs from the same cities in the same countries. Figures 8.12 and 8.13 show image pairs between various cities.

8.5.2 Open Set

In this section, we illustrate various image pairs sampled from the open verification dataset. From Figure 8.14 to 8.25, we show different image scenes including street views, panoramic images, and general urban images. The Figures depict different scenarios, including image pairs from the same cities in the same country, as well as pairs of images from different cities worldwide.


New York.

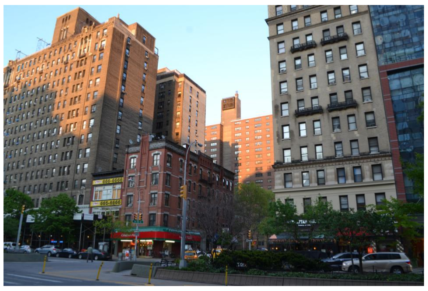
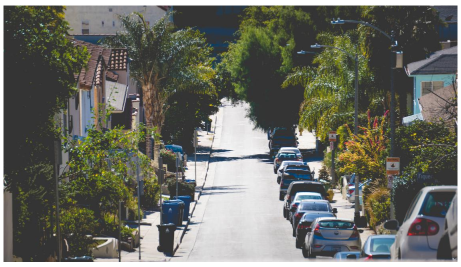

New York.

Figure 8.8: Similarity: 0.99, Sigmoid output: 0.09


Los Angeles.

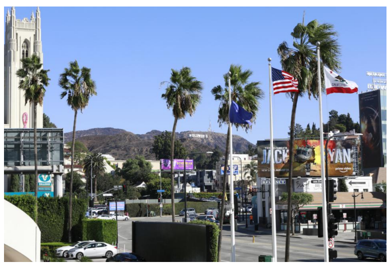
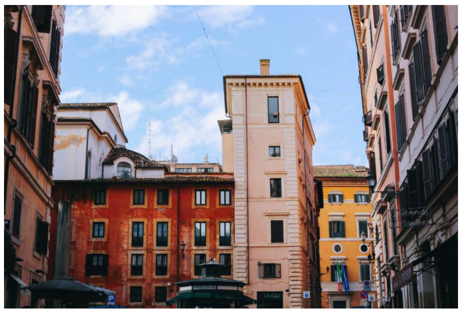

Los Angeles.

Figure 8.9: Similarity: 0.72, Sigmoid output: 0.43


Rome.

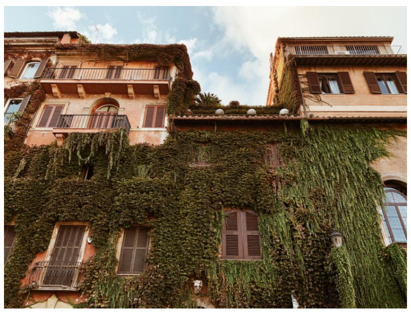
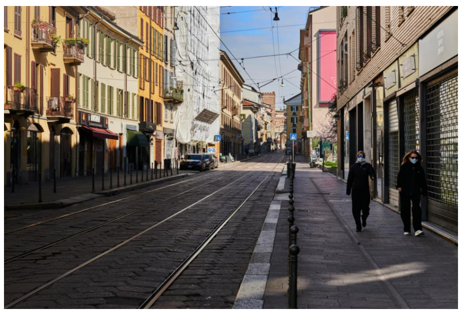
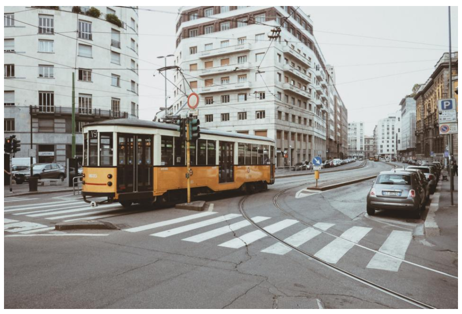

Rome.

Figure 8.10: Similarity: 0.84, Sigmoid output: 0.38

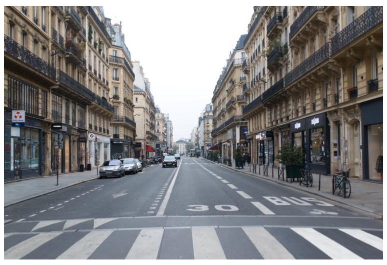


Milan.

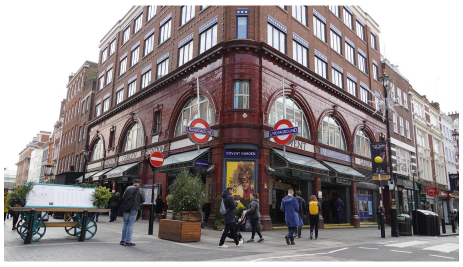


Milan.

Figure 8.11: Similarity: 0.76, Sigmoid output: 0.34

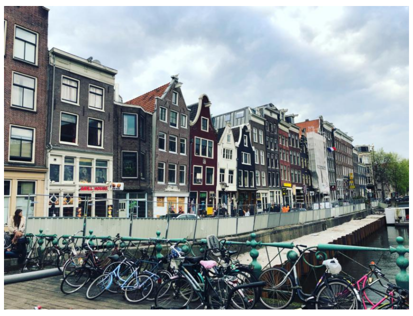


Paris.

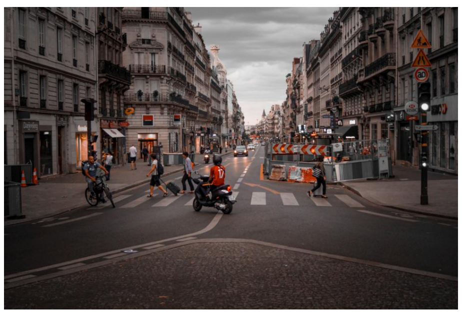


London.

Figure 8.12: Similarity: 0.95, Sigmoid output: 0.60

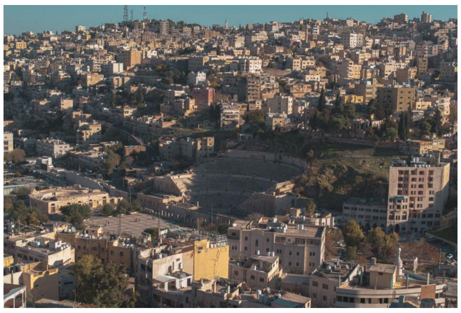


Amsterdam.



Barcelona.

Figure 8.13: Similarity: 0.82, Sigmoid output: 0.71


Amman.

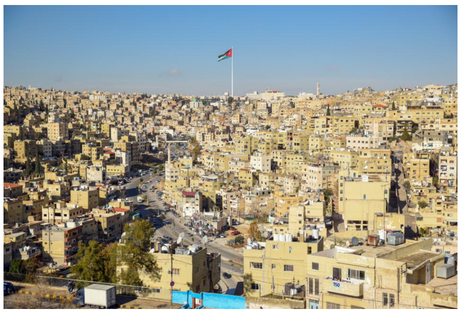
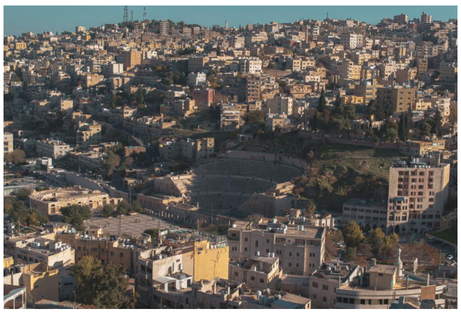

Amman.

Figure 8.14: Similarity: 0.99, Sigmoid output: 0.33


Amman.

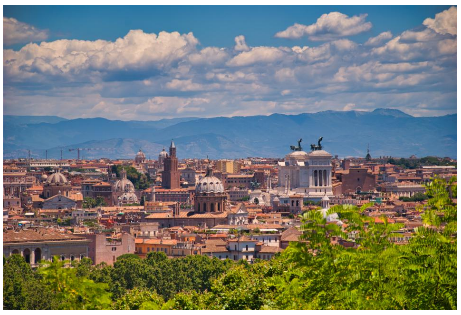
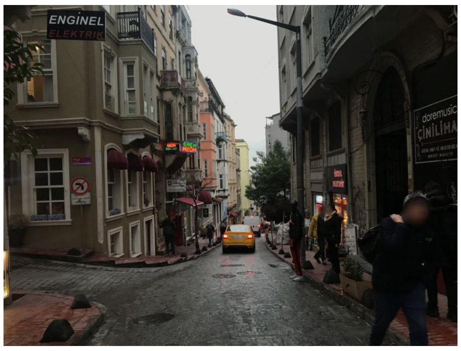

Roma.

Figure 8.15: Similarity: 0.99, Sigmoid output: 0.65



Istanbul.

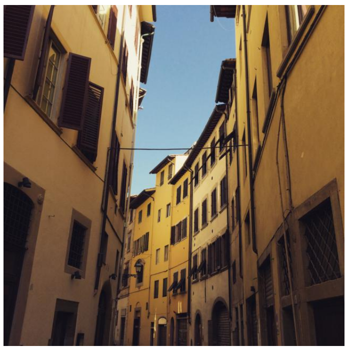
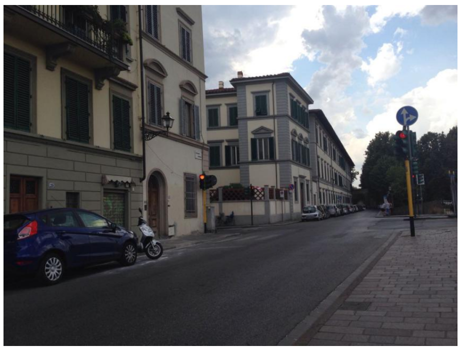

Florence.

Figure 8.16: Similarity: 0.97, Sigmoid output: 0.94


Florence.

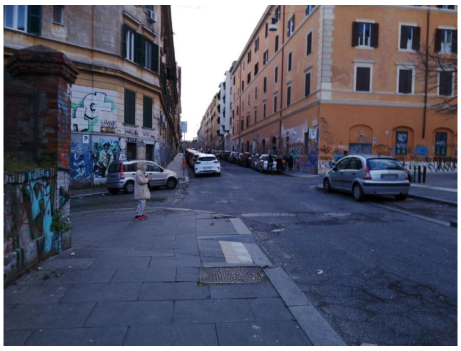
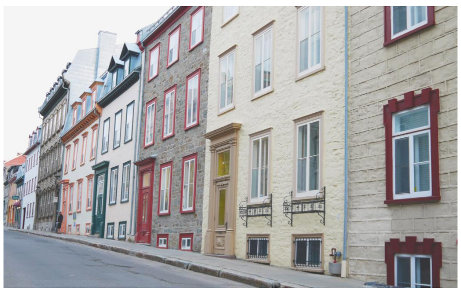

Rome.

Figure 8.17: Similarity: 0.98, Sigmoid output: 0.99


Quebec.

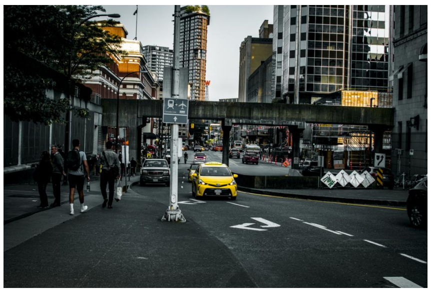
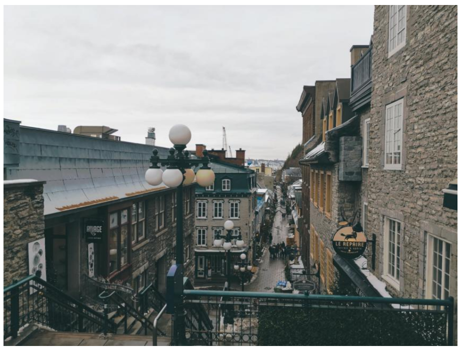

Vancouver.

Figure 8.18: Similarity: 0.96, Sigmoid output: 0.93


Quebec.

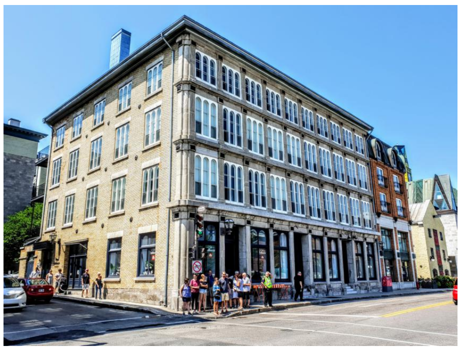
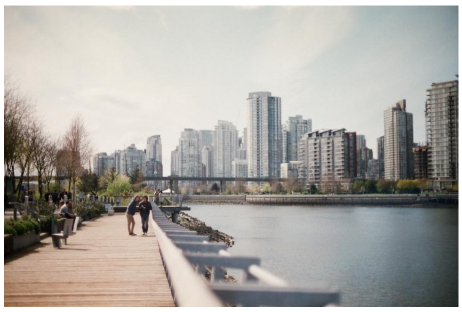

Quebec.

Figure 8.19: Similarity: 0.68, Sigmoid output: 0.45


Vancouver.

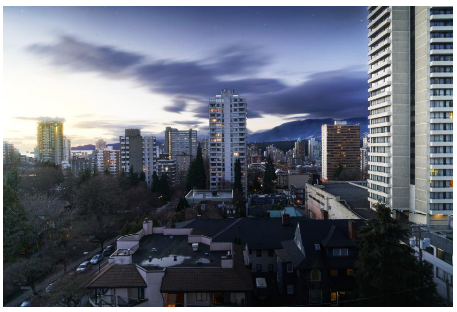
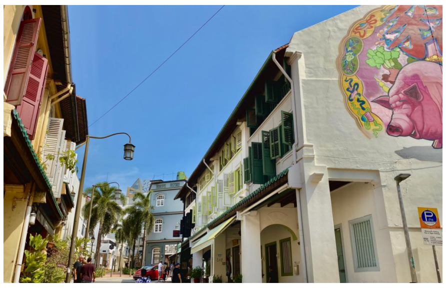

Vancouver.

Figure 8.20: Similarity: 0.10, Sigmoid output: 0.43


Singapore.

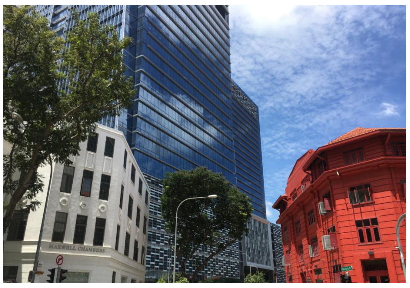
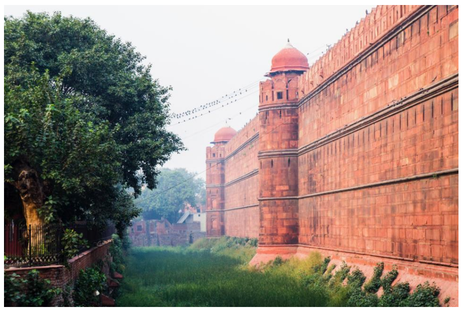

Singapore.

Figure 8.21: Similarity: 0.99, Sigmoid output: 0.43


Delhi.

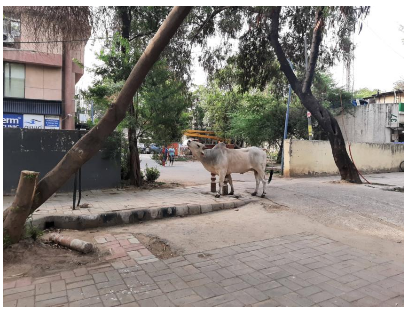
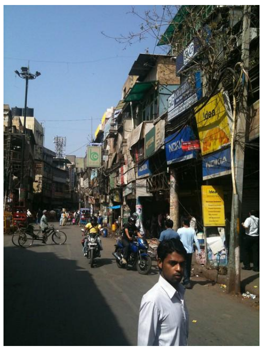

Delhi.

Figure 8.22: Similarity: 0.53, Sigmoid output: 0.41


Delhi.

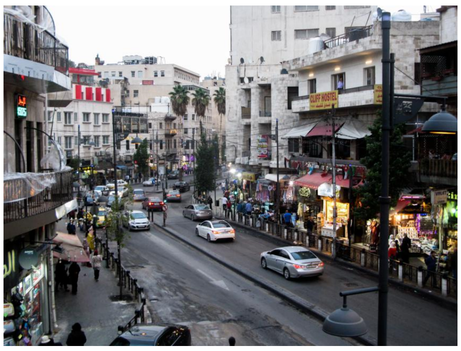
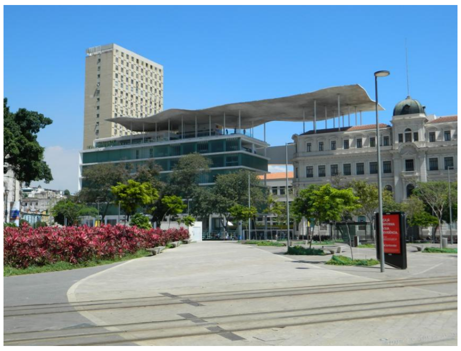

Amman.

Figure 8.23: Similarity: 0.97, Sigmoid output: 0.79


Rio de Janeiro.

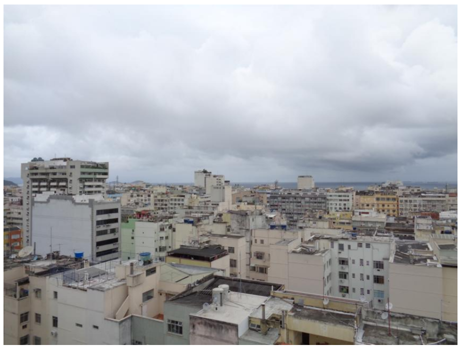
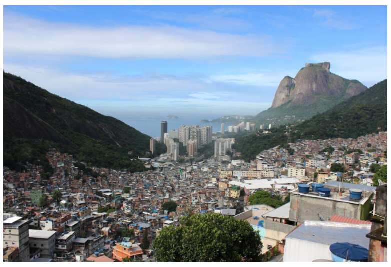

Rio de Janeiro.

Figure 8.24: Similarity: 0.74, Sigmoid output: 0.30


Rio de Janeiro.

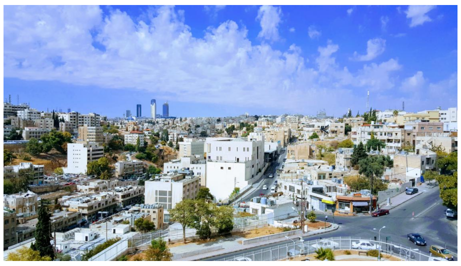

Amman.

Figure 8.25: Similarity: 0.88, Sigmoid output: 0.44

8.6 Conclusion

In this chapter, we have introduced the City Verification task, a new instance in the class of Image Geolocalization problems, and we present our developed system designed to address it. The system uses a Siamese network backboned with Vision Transformer, coupled with a country classifier and Semantic similarity analyser. Given a query image and a small set of images taken in a target city, the system accurately determines whether the query image was taken in the target city or not.

The City Verification problem involves two key factors: i) the vast number of cities worldwide, and ii) the variability in scenes from the same city. Our Siamese-based verifier effectively addresses both challenges. Firstly, it is not necessary to retrain the system to handle images belonging to cities that have not been used in the training set. Secondly, our system gives more voting power to the reference images that are semantically similar to the query image. Moreover, our system compares the query image with several reference images of the claimed city. We argue that if the number of reference images is large enough, then the verifier can handle the variability of images from the same city properly.

Chapter 9
Conclusion

Nothing Comes for Free.
You Can't Have it All.
Do not Take Things Personally.
Omran Alamayreh

Throughout the first part of the thesis, we focused on the detectability of synthetic non-facial deepfake videos. Our investigation involved the creation of two datasets and the development of various detectors tailored for this purpose. Additionally, we developed a detector capable of identifying deepfake videos in scenarios with limited data availability. In the second part, we discussed our contributions to the field of Image Geolocalization, with a particular focus on recognizing cities and countries from images. In this chapter, we recapitulate the main contributions of the thesis, discuss important open issues, and highlight potential directions for future research.

9.1 Summary and Final Remarks

When we initiated our research activity, numerous methodologies and solutions had been published in the field of facial deepfake generation and detection, with only a few works tackling the generation of non-facial deepfake videos, and almost no work addressing the detection of such videos. Consequently, the first part of the thesis is focused on exploring the detectability of this particular type of fake videos.

We have split our study into two primary sections: the generation and the detection of non-facial fake videos. In the generation part, we have created a dataset of fake street videos, employing vid2vid synthesis tools, which we called the DeepStreets dataset. In addition, we have constructed a dataset of human-body fake reenactment videos utilizing the "Everybody Dance Now" framework. We referred to it as the FakeDance dataset. Both datasets have been made accessible online. In the detection part, our focus was on assessing the detectability of these novel fake videos (DeepStreets and FakeDance) using data-driven methods and feature-based methods. Our findings revealed distinct strengths and specific characteristics within each detection approach, showcasing their effectiveness in various operational scenarios. However, we observed that the generalization capacity of both methods heavily relies on specific experimental conditions.

One of the primary challenges in detecting deepfake videos is the availability of training datasets, particularly for DL-based methods that demand extensive data for training. Furthermore, real-world scenarios typically offer a limited number of similar deepfake video samples for forensic analysis. Motivated by the above, we have developed a lightweight Deepfake detector that can work in data scarcity conditions, when only very few examples of fake videos are available to the forensic analyst. The detector relies on a simple footprint obtained from the motion prediction modes within video sequences. The extracted footprint is then utilized to train a simple linear SVM classifier. Our experiments conducted on three diverse datasets demonstrate the effectiveness of our detector, which showcases remarkable performance while requiring minimal video samples for training.

In the landscape of deepfake detection, certain patterns are emerging. First, the quality of deepfake videos is continually evolving, becoming more and more realistic over time. We already surpassed the point where human eyes can differentiate between real and fake content. Second, the deepfake detection problem is an eternal battle between the attacker and the defender. The attacker will consistently come up with new generative models that generate increasingly realistic deepfakes compared to their predecessors. The defender will try to develop methods to detect the new deepfakes content. The prospect of a one-size-fits-all deepfake detector seems unattainable, at least in the near future. Consequently, this raises substantial concerns regarding the authenticity and trustworthiness of visual media content.

We strongly believe the scientific community bears a huge responsibility in safeguarding our digital landscape from deepfakes. Some important measures that need to be taken include:

- Publicly sharing generative models, including code and pre-trained models.
- Constructing large datasets that contain diverse deepfake videos from the latest generative models.
- Raising public awareness about the phenomenon of deepfake.

Wishing for a world where we can still believe what we see.

In the second part of the thesis, we focused on the task of Image Geolocalization aimed at revealing the country and the city where a photo was captured. To accomplish that, we have collected a dataset of 3.8 million geo-tagged images by crawling the web. The images contain urban outdoor scenes only, due to the relevance of such kinds of images for detecting the architectural and semantic differences between countries and cities. The dataset is named VIPPGeo and is available online.

Leveraging the VIPPGeo dataset, we have addressed the Country Recognition task by training a DL model. We have cast the task as a classification problem. In particular, our experiments revealed that directly asking the model to identify the country from a photo yielded superior results compared to estimate GPS geo-coordinates and subsequently deducing the country from the GPS estimation. These findings showcase the efficiency of our model, outperforming the current state-of-the-art approaches in this domain.

In addition to the Country Recognition, we have addressed the problem of recognizing the city where a photo has been captured. Due to the vast number of cities in the world and the considerable variability in scenes originating from the same city, we depart from the conventional inference and classification methods employed thus far and reframe the city recognition problem as a verification task. Specifically, we have developed a system based on a Siamese network designed to determine whether a photo was captured in a particular city or not, with the assumption that a small set of images from that city is available. Moreover, we have designed our system to assign greater voting power to reference images that exhibit semantic similarity to the query photo, allowing for more accurate verification. Our system can handle the case of photos taken in cities that have not been used during training. The experiments we run prove the validity of the developed system which outperforms solutions based on state-of-the-art techniques, even in the challenging case of photos shot in different cities of the same country.

We believe that solving the problem of Image Geolocalization, covering tasks like Country Recognition, City Verification, and even pinpointing exact GPS coordinates from photos, is achievable to some extent. Despite its complexity, recent advancements in AI, particularly the emergence of very large models and the abundance of large Image datasets make the resolution of this challenge plausible if a proper GPU infrastructure is available. This was not feasible in the past. Furthermore, we anticipate that in some cases, this problem will become as straightforward as recognizing handwritten digits in the near future.

The ability to create a model capable of accurately determining the precise GPS location of images worldwide raises significant concerns regarding individuals' privacy and security. This technology could potentially be misused, revealing the geographic locations of individuals in photos without their consent for malicious purposes. We believe that it is the responsibility of the scientific community to mitigate the impact of this issue. For instance, rather than developing models that pinpoint exact GPS locations, they could return a broader areas like countries, regions or cities. Additionally, there is a need for methods that safeguard individuals' privacy by introducing noise to deceive Image Geolocalization models.

9.2 Open Issues

The work of this thesis paves the way for several future research directions, some of which are described below.

With regard to non-facial deepfake videos, future research could be devoted to create a dataset containing content going beyond street videos and human reenactment videos. This may include F1 race videos, tennis match videos, and many more. Another direction for future research is to investigate the generalization capability of the forgery detectors to classes of non-facial synthetic videos that are not included in the training set. Moreover, the investigation of an advanced fusion strategy for handcrafted and DL models for the task of detecting non-facial fake videos could also be a promising research direction. Finally, it would be interesting to investigate the robustness of the detectors we have developed against adversarial attacks aimed at detection failure.

Concerning the lightweight deepfake detector, several future research directions could be explored. For instance, improving the performance in the case of manipulated videos (non-fully synthetic), where only a region of the frames is modified, like in the case of face swapping case, by confining the extraction of the footprint to the foreground region.

Moreover, a comprehensive assessment of the generalization capability of the developed detector based on video coding analysis could be performed. Finally, the robustness of the detector against video re-compression and in general intentional attacks, e.g., adversarial attacks, is also worth investigating.

With regard to Image Geolocalization, a first future perspective consists of extending the VIPPGeo dataset to group world countries into smaller classes. Furthermore, enhancing the efficacy of our Country Recognition system could involve leveraging distinct country-specific features, such as the presence of text within images or different driving behaviours (left or right-handed). This improvement could be achieved through the use of large pre-trained models like CLIP [141] or Large ViTs [167]. Additionally, adding tools to explain which aspects of the images the Country Recognition framework focuses on, would significantly improve its usability.

While our City Verification system shows promising results, we acknowledge that the experiments we have carried out can only prove the plausibility of the arguments our system relies on. Thus, future research should focus on improving the generality and scalability of our approach. Additionally, optimizing the choice of the images in the reference dataset and adopting more sophisticated fusion strategies could further enhance the system's performance. Similar to Country Recognition, an explainability study could prove beneficial to understanding the specific image areas our system focuses on when making a decision.

Bibliography

[1] F. Chollet, “Xception: Deep learning with depthwise separable convolutions,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1251–1258.

[2] E. Muller-Budack, K. Pustu-Iren, and R. Ewerth, “Geolocation estimation of photos using a hierarchical model and scene classification,” in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 563–579.

[3] O. Alamayreh and M. Barni, “Detection of gan-synthesized street videos,” in *2021 29th European Signal Processing Conference (EUSIPCO)*. IEEE, 2021, pp. 811–815.

[4] L. Verdoliva, “Media forensics and deepfakes: an overview,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 14, no. 5, pp. 910–932, 2020.

[5] “Semantic forensics,” https://www.darpa.mil/program/semantic-forensics.

[6] T. T. Nguyen, C. M. Nguyen, D. T. Nguyen, D. T. Nguyen, and S. Nahavandi, “Deep learning for deepfakes creation and detection: A survey,” *arXiv preprint arXiv:1909.11573*, 2019.

[7] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, G. Liu, A. Tao, J. Kautz, and B. Catanzaro, “Video-to-video synthesis,” *arXiv preprint arXiv:1808.06601*, 2018.

[8] C. Chan, S. Ginosar, T. Zhou, and A. A. Efros, “Everybody dance now,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 5933–5942.

[9] S. Garg, T. Fischer, and M. Milford, “Where is your place, visual place recognition?” *arXiv preprint arXiv:2103.06443*, 2021.

[10] X. Zhang, L. Wang, and Y. Su, “Visual place recognition: A survey from deep learning perspective,” *Pattern Recognition*, vol. 113, p. 107760, 2021.

[11] A. Ghai, P. Kumar, and S. Gupta, “A deep-learning-based image forgery detection framework for controlling the spread of misinformation,” *Information Technology & People*, 2021.

[12] S. Workman, R. Souvenir, and N. Jacobs, “Wide-area image geolocalization with aerial reference imagery,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 3961–3969.

[13] I. Amerini, A. Anagnostopoulos, L. Maiano, L. R. Celsi *et al.*, “Deep learning for multimedia forensics,” *Foundations and Trends® in Computer Graphics and Vision*, vol. 12, no. 4, pp. 309–457, 2021.

[14] A. C. Popescu and H. Farid, “Statistical tools for digital forensics,” in *International workshop on information hiding.* Springer, 2004, pp. 128–147.

[15] J. Fridrich, D. Soukal, J. Lukas *et al.*, “Detection of copy-move forgery in digital images,” in *Proceedings of digital forensic research workshop*, vol. 3, no. 2. Cleveland, OH, 2003, pp. 652–63.

[16] M. C. Stamm, M. Wu, and K. R. Liu, “Information forensics: An overview of the first decade,” *IEEE access*, vol. 1, pp. 167–200, 2013.

[17] G. E. Hinton and R. Zemel, “Autoencoders, minimum description length and helmholtz free energy,” *Advances in neural information processing systems*, vol. 6, 1993.

[18] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, “Extracting and composing robust features with denoising autoencoders,” in *Proceedings of the 25th international conference on Machine learning*, 2008, pp. 1096–1103.

[19] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial networks,” *arXiv preprint arXiv:1406.2661*, 2014.

[20] J. Ho, A. Jain, and P. Abbeel, “Denoising diffusion probabilistic models,” *Advances in neural information processing systems*, vol. 33, pp. 6840–6851, 2020.

[21] O. Alamayreh, C. Fascella, S. Mandelli, B. Tondi, P. Bestagini, and M. Barni, “Just dance: Detection of human body reenactment fake videos,” 2022.

[22] R. Salloum, Y. Ren, and C.-C. J. Kuo, “Image splicing localization using a multi-task fully convolutional network (mfcn),” *Journal of Visual Communication and Image Representation*, vol. 51, pp. 201–209, 2018.

[23] Y. Wu, W. Abd-Almageed, and P. Natarajan, “Busternet: Detecting copy-move image forgery with source/target localization,” in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 168–184.

[24] B. Bayar and M. C. Stamm, “On the robustness of constrained convolutional neural networks to jpeg post-compression for image resampling detection,” in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* IEEE, 2017, pp. 2152–2156.

[25] C. Long, E. Smith, A. Basharat, and A. Hoogs, “A c3d-based convolutional neural network for frame dropping detection in a single video shot,” in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).* IEEE, 2017, pp. 1898–1906.

[26] S. Verde, L. Bondi, P. Bestagini, S. Milani, G. Calvagno, and S. Tubaro, “Video codec forensics based on convolutional neural networks,” in *2018 25th IEEE International Conference on Image Processing (ICIP).* IEEE, 2018, pp. 530–534.

[27] S.-H. Nam, J. Park, D. Kim, I.-J. Yu, T.-Y. Kim, and H.-K. Lee, “Two-stream network for detecting double compression of h. 264 videos,” in *2019 IEEE International Conference on Image Processing (ICIP).* IEEE, 2019, pp. 111–115.

[28] D. Cozzolino, J. Thies, A. Rössler, C. Riess, M. Nießner, and L. Verdoliva, “Forensictransfer: Weakly-supervised domain adaptation for forgery detection,” *arXiv preprint arXiv:1812.02510*, 2018.

[29] Y. Wu, W. AbdAlmageed, and P. Natarajan, “Mantra-net: Manipulation tracing network for detection and localization of image forgeries with anomalous features,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 9543–9552.

[30] D. Cozzolino and L. Verdoliva, “Noiseprint: A cnn-based camera model fingerprint,” *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 144–159, 2019.

[31] S. Mandelli, D. Cozzolino, P. Bestagini, L. Verdoliva, and S. Tubaro, “Cnn-based fast source device identification,” *IEEE Signal Processing Letters*, vol. 27, pp. 1285–1289, 2020.

[32] A. Ferreira, H. Chen, B. Li, and J. Huang, “An inception-based data-driven ensemble approach to camera model identification,” in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2018, pp. 1–7.

[33] I. Petrov, D. Gao, N. Chervoniy, K. Liu, S. Marangonda, C. Umé, J. Jiang, L. RP, S. Zhang, P. Wu *et al.*, “Deepfacelab: A simple, flexible and extensible face swapping framework,” *arXiv preprint arXiv:2005.05535*, 2020.

[34] GitHub, “Faceswap-gan,” https://github.com/shaoanlu/faceswap-GAN.

[35] VGGFace, https://github.com/rcmalli/kerasvggface.

[36] DFaker, https://github.com/dfaker/df.

[37] J. Thies, M. Zollhöfer, M. Stamminger, C. Theobalt, and M. Nießner, “Face2face: Real-time face capture and reenactment of RGB videos,” *CoRR*, vol. abs/2007.14808, 2020. [Online]. Available: https://arxiv.org/abs/2007.14808

[38] J. Thies, M. Elgharib, A. Tewari, C. Theobalt, and M. Nießner, “Neural voice puppetry: Audio-driven facial reenactment,” *CoRR*, vol. abs/1912.05566, 2019. [Online]. Available: http://arxiv.org/abs/1912.05566

[39] L. Yang, Z. Zhang, Y. Song, S. Hong, R. Xu, Y. Zhao, W. Zhang, B. Cui, and M.-H. Yang, “Diffusion models: A comprehensive survey of methods and applications,” *ACM Computing Surveys*, vol. 56, no. 4, pp. 1–39, 2023.

[40] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, “Hierarchical text-conditional image generation with clip latents,” *arXiv preprint arXiv:2204.06125*, vol. 1, no. 2, p. 3, 2022.

[41] C. Saharia, W. Chan, S. Saxena, L. Li, J. Whang, E. L. Denton, K. Ghasemipour, R. Gontijo Lopes, B. Karagol Ayan, T. Salimans *et al.*, “Photorealistic text-to-image diffusion models with deep language understanding,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 36 479–36 494, 2022.

[42] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer, “High-resolution image synthesis with latent diffusion models,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 10 684–10 695.

[43] A. Nichol, P. Dhariwal, A. Ramesh, P. Shyam, P. Mishkin, B. McGrew, I. Sutskever, and M. Chen, “Glide: Towards photorealistic image generation and editing with text-guided diffusion models,” *arXiv preprint arXiv:2112.10741*, 2021.

[44] A. Rossler, D. Cozzolino, L. Verdoliva, C. Riess, J. Thies, and M. Nießner, “Faceforensics++: Learning to detect manipulated facial images,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 1–11.

[45] B. Dolhansky, R. Howes, B. Pflaum, N. Baram, and C. C. Ferrer, “The deepfake detection challenge (dfdc) preview dataset,” *arXiv preprint arXiv:1910.08854*, 2019.

[46] Y. Li, X. Yang, P. Sun, H. Qi, and S. Lyu, “Celeb-df: A large-scale challenging dataset for deepfake forensics,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3207–3216.

[47] L. Jiang, R. Li, W. Wu, C. Qian, and C. C. Loy, "Deeperforensics-1.0: A large-scale dataset for real-world face forgery detection," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 2889–2898.

[48] B. Zi, M. Chang, J. Chen, X. Ma, and Y.-G. Jiang, "Wilddeepfake: A challenging real-world dataset for deepfake detection," in *Proceedings of the 28th ACM international conference on multimedia*, 2020, pp. 2382–2390.

[49] P. Kwon, J. You, G. Nam, S. Park, and G. Chae, "Kodf: A large-scale korean deepfake detection dataset," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2021, pp. 10 744–10 753.

[50] F. Matern, C. Riess, and M. Stamminger, "Exploiting visual artifacts to expose deepfakes and face manipulations," in *2019 IEEE Winter Applications of Computer Vision Workshops (WACVW)*. IEEE, 2019, pp. 83–92.

[51] U. A. Ciftci, I. Demir, and L. Yin, "Fakecatcher: Detection of synthetic portrait videos using biological signals," *IEEE transactions on pattern analysis and machine intelligence*, 2020.

[52] S. Fernandes, S. Raj, E. Ortiz, I. Vintila, M. Salter, G. Urosevic, and S. Jha, "Predicting heart rate variations of deepfake videos using neural ode," in *Proceedings of the IEEE/CVF international conference on computer vision workshops*, 2019, pp. 0–0.

[53] Y. Li, M.-C. Chang, and S. Lyu, "In ictu oculi: Exposing ai created fake videos by detecting eye blinking," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2018, pp. 1–7.

[54] Y. Li and S. Lyu, "Exposing deepfake videos by detecting face warping artifacts," *arXiv preprint arXiv:1811.00656*, 2018.

[55] X. Yang, Y. Li, H. Qi, and S. Lyu, "Exposing gan-synthesized faces using landmark locations," in *Proceedings of the ACM workshop on information hiding and multimedia security*, 2019, pp. 113–118.

[56] X. Yang, Y. Li, and S. Lyu, "Exposing deep fakes using inconsistent head poses," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019, pp. 8261–8265.

[57] V. Conotter, E. Bodnari, G. Boato, and H. Farid, "Physiologically-based detection of computer generated faces in video," in *2014 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2014, pp. 248–252.

[58] S. Agarwal, H. Farid, Y. Gu, M. He, K. Nagano, and H. Li, "Protecting world leaders against deep fakes." in *CVPR workshops*, vol. 1, 2019, p. 38.

[59] R. Tolosana, R. Vera-Rodriguez, J. Fierrez, A. Morales, and J. Ortega-Garcia, "Deepfakes and beyond: A survey of face manipulation and fake detection," *Information Fusion*, vol. 64, pp. 131–148, 2020.

[60] D. Afchar, V. Nozick, J. Yamagishi, and I. Echizen, "Mesonet: a compact facial video forgery detection network," in *2018 IEEE International Workshop on Information Forensics and Security (WIFS)*. IEEE, 2018, pp. 1–7.

[61] H. H. Nguyen, J. Yamagishi, and I. Echizen, "Use of a capsule network to detect fake images and videos," *ArXiv*, vol. abs/1910.12467, 2019.

[62] N. Bonettini, E. D. Cannas, S. Mandelli, L. Bondi, P. Bestagini, and S. Tubaro, "Video face manipulation detection through ensemble of CNNs," in *International Conference on Pattern Recognition (ICPR)*, 2021.

[63] S. Seferbekov, "Deepfake detection (dfdc) solution," https://github.com/selimsef/dfdc_deepfake_challenge, 2021.

[64] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 4700–4708.

[65] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.

[66] A. Graves, S. Fernández, and J. Schmidhuber, "Multi-dimensional recurrent neural networks," in *International conference on artificial neural networks.* Springer, 2007, pp. 549–558.

[67] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[68] D. Güera and E. J. Delp, "Deepfake video detection using recurrent neural networks," in *2018 15th IEEE International Conference on Advanced Video and Dignal Based Surveillance (AVSS).* IEEE, 2018, pp. 1–6.

[69] S. J. Sohrawardi, A. Chintha, B. Thai, S. Seng, A. Hickerson, R. Ptucha, and M. Wright, "Poster: Towards robust open-world detection of deepfakes," in *Proceedings of the 2019 ACM SIGSAC conference on computer and communications security*, 2019, pp. 2613–2615.

[70] E. Sabir, J. Cheng, A. Jaiswal, W. AbdAlmageed, I. Masi, and P. Natarajan, "Recurrent convolutional strategies for face manipulation detection in videos," *Interfaces (GUI)*, vol. 3, no. 1, pp. 80–87, 2019.

[71] D. Lin, B. Tondi, B. Li, and M. Barni, "Exploiting temporal information to prevent the transferability of adversarial examples against deep fake detectors," in *2022 IEEE International Joint Conference on Biometrics (IJCB).* IEEE, 2022, pp. 1–8.

[72] I. Amerini, L. Galteri, R. Caldelli, and A. Del Bimbo, "Deepfake video detection through optical flow based cnn," in *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, 2019, pp. 0–0.

[73] I. Amerini and R. Caldelli, "Exploiting prediction error inconsistencies through LSTM-based classifiers to detect deepfake videos," in *Proceedings of the 2020 ACM Workshop on Information Hiding and Multimedia Security*, 2020, pp. 97–102.

[74] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *arXiv preprint arXiv:2010.11929*, 2020.

[75] A. Khormali and J.-S. Yuan, "Dfdt: an end-to-end deepfake detection framework using vision transformer," *Applied Sciences*, vol. 12, no. 6, p. 2953, 2022.

[76] X. Dong, J. Bao, D. Chen, T. Zhang, W. Zhang, N. Yu, D. Chen, F. Wen, and B. Guo, "Protecting celebrities from deepfake with identity consistency transformer," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 9468–9478.

[77] J. Wang, Z. Wu, W. Ouyang, X. Han, J. Chen, Y.-G. Jiang, and S.-N. Li, "M2tr: Multi-modal multi-scale transformers for deepfake detection," in *Proceedings of the 2022 international conference on multimedia retrieval*, 2022, pp. 615–623.

[78] D. A. Coccomini, N. Messina, C. Gennaro, and F. Falchi, "Combining efficientnet and vision transformers for video deepfake detection," in *International conference on image analysis and processing.* Springer, 2022, pp. 219–229.

[79] B. Kaddar, S. A. Fezza, W. Hamidouche, Z. Akhtar, and A. Hadid, "Hcit: Deepfake video detection using a hybrid model of cnn features and vision transformer," in *2021 International Conference on Visual Communications and Image Processing (VCIP).* IEEE, 2021, pp. 1–5.

[80] S. Ganguly, A. Ganguly, S. Mohiuddin, S. Malakar, and R. Sarkar, "Vixnet: Vision transformer with xception network for deepfakes based video and image forgery detection," *Expert Systems with Applications*, vol. 210, p. 118423, 2022.

[81] S. Milani, M. Fontani, P. Bestagini, M. Barni, A. Piva, M. Tagliasacchi, and S. Tubaro, "An overview on video forensics," *APSIPA Transactions on Signal and Information Processing*, vol. 1, 2012.

[82] D. Vazquez-Padin, M. Fontani, T. Bianchi, P. Comesaña, A. Piva, and M. Barni, "Detection of video double encoding with gop size estimation," in *2012 IEEE International Workshop on Information Forensics and Security (WIFS).* IEEE, 2012, pp. 151–156.

[83] H. Yao, S. Song, C. Qin, Z. Tang, and X. Liu, "Detection of double-compressed h. 264/avc video incorporating the features of the string of data bits and skip macroblocks," *Symmetry*, vol. 9, no. 12, p. 313, 2017.

[84] A. Costanzo and M. Barni, "Detection of double avc/hevc encoding," in *2016 24th European Signal Processing Conference (EUSIPCO).* IEEE, 2016, pp. 2245–2249.

[85] Y. Yu, H. Yao, R. Ni, and Y. Zhao, "Detection of fake high definition for hevc videos based on prediction mode feature," *Signal Processing*, vol. 166, p. 107269, 2020.

[86] Y. Nirkin, Y. Keller, and T. Hassner, "Fsgan: Subject agnostic face swapping and reenactment," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[87] J. Thies, M. Zollhofer, M. Stamminger, C. Theobalt, and M. Nießner, "Face2face: Real-time face capture and reenactment of rgb videos," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2387–2395.

[88] A. Mallya, T.-C. Wang, K. Sapra, and M.-Y. Liu, "World-consistent video-to-video synthesis," *arXiv preprint arXiv:2007.08509*, 2020.

[89] T.-C. Wang, M.-Y. Liu, A. Tao, G. Liu, J. Kautz, and B. Catanzaro, "Few-shot video-to-video synthesis," *arXiv preprint arXiv:1910.12713*, 2019.

[90] T. Xue, J. Wu, K. L. Bouman, and W. T. Freeman, "Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks," *arXiv preprint arXiv:1607.02586*, 2016.

[91] C. Vondrick, H. Pirsiavash, and A. Torralba, "Generating videos with scene dynamics," *arXiv preprint arXiv:1609.02612*, 2016.

[92] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 8798–8807.

[93] D. Chen, J. Liao, L. Yuan, N. Yu, and G. Hua, "Coherent online video style transfer," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 1105–1114.

[94] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The cityscapes dataset for semantic urban scene understanding," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 3213–3223.

[95] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The kitti dataset," *International Journal of Robotics Research (IJRR)*, 2013.

[96] Y. Zhu, K. Sapra, F. A. Reda, K. J. Shih, S. Newsam, A. Tao, and B. Catanzaro, "Improving semantic segmentation via video propagation and label relaxation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8856–8865.

[97] GitHub, "Opensfm," https://github.com/mapillary/OpenSfM.

[98] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," *arXiv preprint arXiv:1912.01703*, 2019.

[99] H. Farid, "Seeing is not believing," *IEEE Spectrum*, vol. 46, no. 8, pp. 44–51, 2009.

[100] Z. Cao, G. Hidalgo, T. Simon, S.-E. Wei, and Y. Sheikh, "Openpose: realtime multi-person 2d pose estimation using part affinity fields," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 43, no. 1, pp. 172–186, 2019.

[101] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional gans," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[102] S. Tomar, "Converting video formats with ffmpeg," *Linux Journal*, vol. 2006, no. 146, p. 10, 2006.

[103] W. Zhou, B. Alan C., and L. Ligang, "Why is image quality assessment so difficult?" in *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2002.

[104] Q.-T. Phan, D.-T. Dang-Nguyen, G. Boato, and F. G. De Natale, "Face spoofing detection using ldp-top," in *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2016, pp. 404–408.

[105] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "Flownet 2.0: Evolution of optical flow estimation with deep networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2462–2470.

[106] T. Jabid, M. H. Kabir, and O. Chae, "Local directional pattern (ldp) for face recognition," in *2010 digest of technical papers International Conference on Consumer Electronics (ICCE)*. IEEE, 2010, pp. 329–330.

[107] M. Tan and Q. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *International Conference on Machine Learning*. PMLR, 2019, pp. 6105–6114.

[108] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*. Ieee, 2009, pp. 248–255.

[109] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, 2019.

[110] S. Mandelli, D. Cozzolino, E. D. Cannas, J. P. Cardenuto, D. Moreira, P. Bestagini, W. J. Scheirer, A. Rocha, L. Verdoliva, S. Tubaro, and E. J. Delp, "Forensic analysis of synthetically generated western blot images," *IEEE Access*, pp. 1–1, 2022.

[111] S.-Y. Wang, O. Wang, R. Zhang, A. Owens, and A. A. Efros, "Cnn-generated images are surprisingly easy to spot... for now," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 8695–8704.

[112] S. Mandelli, N. Bonettini, P. Bestagini, and S. Tubaro, "Detecting gan-generated images by orthogonal training of multiple cnns," in *2022 IEEE International Conference on Image Processing (ICIP).* IEEE, 2022, pp. 3091–3095.

[113] B. Liu, F. Yang, X. Bi, B. Xiao, W. Li, and X. Gao, "Detecting generated images by real images," in *European Conference on Computer Vision.* Springer, 2022, pp. 95–110.

[114] G. Fox, W. Liu, H. Kim, H.-P. Seidel, M. Elgharib, and C. Theobalt, "Videoforensicshq: Detecting high-quality manipulated face videos," in *2021 IEEE International Conference on Multimedia and Expo (ICME).* IEEE, 2021, pp. 1–6.

[115] M. Du, S. Pentyala, Y. Li, and X. Hu, "Towards generalizable deepfake detection with locality-aware autoencoder," in *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, 2020, pp. 325–334.

[116] Y. Wang, Q. Yao, J. T. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," *ACM Computing Surveys (CSUR)*, vol. 53, no. 3, pp. 1–34, 2020.

[117] S. Tariq, S. Lee, and S. S. Woo, "A convolutional LSTM based residual network for deepfake video detection," *arXiv preprint arXiv:2009.07480*, 2020.

[118] I. E. Richardson, *The H. 264 advanced video compression standard.* John Wiley & Sons, 2011.

[119] V. Sze, M. Budagavi, and G. J. Sullivan, "High efficiency video coding (hevc)," in *Integrated circuit and systems, algorithms and architectures.* Springer, 2014, vol. 39, p. 40.

[120] H.-S. Chen, M. Rouhsedaghat, H. Ghani, S. Hu, S. You, and C.-C. J. Kuo, "Defakehop: A light-weight high-performance deepfake detector," in *2021 IEEE International Conference on Multimedia and Expo (ICME).* IEEE, 2021, pp. 1–6.

[121] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic, "Netvlad: Cnn architecture for weakly supervised place recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 5297–5307.

[122] H. J. Kim, E. Dunn, and J.-M. Frahm, "Predicting good features for image geo-localization using per-bundle vlad," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1170–1178.

[123] H. Jin Kim, E. Dunn, and J.-M. Frahm, "Learned contextual feature reweighting for image geo-localization," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 2136–2145.

[124] F. Radenović, G. Tolias, and O. Chum, "Cnn image retrieval learns from bow: Unsupervised fine-tuning with hard examples," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part I 14.* Springer, 2016, pp. 3–20.

[125] G. Schindler, M. Brown, and R. Szeliski, "City-scale location recognition," in *2007 IEEE Conference on Computer Vision and Pattern Recognition.* IEEE, 2007, pp. 1–7.

[126] E. Zemene, Y. T. Tesfaye, H. Idrees, A. Prati, M. Pelillo, and M. Shah, "Large-scale image geo-localization using dominant sets," *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 1, pp. 148–161, 2018.

[127] N. Vo, N. Jacobs, and J. Hays, "Revisiting im2gps in the deep learning era," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2621–2630.

[128] S. Pramanick, E. M. Nowara, J. Gleason, C. D. Castillo, and R. Chellappa, "Where in the world is this image? transformer-based geo-localization in the wild," in *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part XXXVIII.* Springer, 2022, pp. 196–215.

[129] T. Weyand, I. Kostrikov, and J. Philbin, "Planet-photo geolocation with convolutional neural networks," in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VIII 14.* Springer, 2016, pp. 37–55.

[130] P. H. Seo, T. Weyand, J. Sim, and B. Han, "Cplanet: Enhancing image geolocalization by combinatorial partitioning of maps," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 536–551.

[131] J. Bromley, I. Guyon, Y. LeCun, E. Säckinger, and R. Shah, "Signature verification using a" siamese" time delay neural network," *Advances in neural information processing systems*, vol. 6, 1993.

[132] J. Hays and A. A. Efros, "Im2gps: estimating geographic information from a single image," in *2008 ieee conference on computer vision and pattern recognition.* IEEE, 2008, pp. 1–8.

[133] J. Hays and A. A. Efros, "Large-scale image geolocalization," *Multimodal location estimation of videos and images*, pp. 41–62, 2015.

[134] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[135] M. Izbicki, E. E. Papalexakis, and V. J. Tsotras, "Exploiting the earth's spherical geometry to geolocate images," in *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2019, Würzburg, Germany, September 16–20, 2019, Proceedings, Part II.* Springer, 2020, pp. 3–19.

[136] G. Kordopatis-Zilos, P. Galopoulos, S. Papadopoulos, and I. Kompatsiaris, "Leveraging efficientnet and contrastive learning for accurate global-scale location estimation," in *Proceedings of the 2021 International Conference on Multimedia Retrieval*, 2021, pp. 155–163.

[137] A. Panagiotopoulos, G. Kordopatis-Zilos, and S. Papadopoulos, "Leveraging selective prediction for reliable image geolocation," in *MultiMedia Modeling: 28th International Conference, MMM 2022, Phu Quoc, Vietnam, June 6–10, 2022, Proceedings, Part II.* Springer, 2022, pp. 369–381.

[138] R. El-Yaniv *et al.*, "On the foundations of noise-free selective classification." *Journal of Machine Learning Research*, vol. 11, no. 5, 2010.

[139] B. Clark, A. Kerrigan, P. P. Kulkarni, V. V. Cepeda, and M. Shah, "Where we are and what we're looking at: Query based worldwide image geo-localization using hierarchies and scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 23 182–23 190.

[140] V. V. Cepeda, G. K. Nayak, and M. Shah, “Geoclip: Clip-inspired alignment between locations and images for effective worldwide geo-localization,” *arXiv preprint arXiv:2309.16020*, 2023.

[141] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, “Learning transferable visual models from natural language supervision,” in *International conference on machine learning.* PMLR, 2021, pp. 8748–8763.

[142] J. Theiner, E. Müller-Budack, and R. Ewerth, “Interpretable semantic photo geolocation,” in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2022, pp. 750–760.

[143] “Openstreetmap contributors. planet dump retrieved from https://planet.osm.org. https://www.openstreetmap.org,” 2017.

[144] G. Luo, G. Biamby, T. Darrell, D. Fried, and A. Rohrbach, “G^ 3: Geolocation via guidebook grounding,” *arXiv preprint arXiv:2211.15521*, 2022.

[145] T.-Y. Lin, S. Belongie, and J. Hays, “Cross-view image geolocalization,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 891–898.

[146] A. Boiarov and E. Tyantov, “Large scale landmark recognition via deep metric learning,” in *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, 2019, pp. 169–178.

[147] T. Weyand and B. Leibe, “Visual landmark recognition from internet photo collections: A large scale evaluation,” *Computer Vision and Image Understanding*, vol. 135, pp. 1–15, 2015.

[148] A. Boguszewski, D. Batorski, N. Ziemba-Jankowska, T. Dziedzic, and A. Zambrzycka, “Landcover. ai: Dataset for automatic mapping of buildings, woodlands, water and roads from aerial imagery,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 1102–1110.

[149] G. Berton, R. Mereu, G. Trivigno, C. Masone, G. Csurka, T. Sattler, and B. Caputo, “Deep visual geo-localization benchmark,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 5396–5407.

[150] S. Schubert, P. Neubert, S. Garg, M. Milford, and T. Fischer, “Visual place recognition: A tutorial,” *arXiv preprint arXiv:2303.03281*, 2023.

[151] C. Masone and B. Caputo, “A survey on deep visual place recognition,” *IEEE Access*, vol. 9, pp. 19 516–19 547, 2021.

[152] S. Lowry, N. Sünderhauf, P. Newman, J. J. Leonard, D. Cox, P. Corke, and M. J. Milford, “Visual place recognition: A survey,” *ieee transactions on robotics*, vol. 32, no. 1, pp. 1–19, 2015.

[153] A. Torii, J. Sivic, T. Pajdla, and M. Okutomi, “Visual place recognition with repetitive structures,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2013, pp. 883–890.

[154] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla, “24/7 place recognition by view synthesis,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1808–1817.

[155] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, “1 year, 1000 km: The oxford robotcar dataset,” *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2017.

[156] T. Naseer, L. Spinello, W. Burgard, and C. Stachniss, “Robust visual robot localization across seasons using network flows,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 28, no. 1, 2014.

[157] B. Thomee, D. A. Shamma, G. Friedland, B. Elizalde, K. Ni, D. Poland, D. Borth, and L.-J. Li, “Yfcc100m: The new data in multimedia research,” *Communications of the ACM*, vol. 59, no. 2, pp. 64–73, 2016.

[158] M. Larson, M. Soleymani, G. Gravier, B. Ionescu, and G. J. Jones, “The benchmarking initiative for multimedia evaluation: Mediaeval 2016,” *IEEE MultiMedia*, vol. 24, no. 1, pp. 93–96, 2017.

[159] “Flickr website,” flickr.com.

[160] “Mapillary website,” mapillary.com.

[161] “Unsplash website,” unsplash.com.

[162] “Geonames - all cities with a population over 1000,” https://pypi.org/project/reverse_geocoder.

[163] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba, “Places: A 10 million image database for scene recognition,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 6, pp. 1452–1464, 2017.

[164] D. E. King, “Dlib-ml: A machine learning toolkit,” *The Journal of Machine Learning Research*, vol. 10, pp. 1755–1758, 2009.

[165] Y. Cui, M. Jia, T.-Y. Lin, Y. Song, and S. Belongie, “Class-balanced loss based on effective number of samples,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2019, pp. 9268–9277.

[166] K. H. Brodersen, C. S. Ong, K. E. Stephan, and J. M. Buhmann, “The balanced accuracy and its posterior distribution,” in *2010 20th international conference on pattern recognition.* IEEE, 2010, pp. 3121–3124.

[167] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017.

[168] Z. Zhang, X. Lu, G. Cao, Y. Yang, L. Jiao, and F. Liu, “Vit-yolo: Transformer-based yolo for object detection,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 2799–2808.

[169] R. Strudel, R. Garcia, I. Laptev, and C. Schmid, “Segmenter: Transformer for semantic segmentation,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 7262–7272.

[170] B. Zhang, Z. Tian, Q. Tang, X. Chu, X. Wei, C. Shen *et al.*, “Segvit: Semantic segmentation with plain vision transformers,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 4971–4982, 2022.

[171] H. Chen, Y. Wang, T. Guo, C. Xu, Y. Deng, Z. Liu, S. Ma, C. Xu, C. Xu, and W. Gao, “Pre-trained image processing transformer,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2021, pp. 12 299–12 310.

[172] A. Arnab, M. Dehghani, G. Heigold, C. Sun, M. Lucic, and C. Schmid, “Vivit: A video vision transformer,” in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 6836–6846.

[173] J. D. M.-W. C. Kenton and L. K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” in *Proceedings of NAACL-HLT*, 2019, pp. 4171–4186.

[174] R. Wang, D. Chen, Z. Wu, Y. Chen, X. Dai, M. Liu, Y.-G. Jiang, L. Zhou, and L. Yuan, “Bevt: Bert pretraining of video transformers,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 14 733–14 743.

[175] S. Chaudhari, V. Mithal, G. Polatkan, and R. Ramanath, “An attentive survey of attention models,” *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 12, no. 5, pp. 1–32, 2021.

[176] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, “Show and tell: A neural image caption generator,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3156–3164.

[177] J. Wang, B. Tondi, and M. Barni, “An eyes-based siamese neural network for the detection of gan-generated face images,” *Front. Sig. Proc. 2: 918725. doi: 10.3389/frsip*, 2022.

[178] Y. Li, C. P. Chen, and T. Zhang, “A survey on siamese network: Methodologies, applications, and opportunities,” *IEEE Transactions on Artificial Intelligence*, vol. 3, no. 6, pp. 994–1014, 2022.

[179] Q. Guo, W. Feng, C. Zhou, R. Huang, L. Wan, and S. Wang, “Learning dynamic siamese network for visual object tracking,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 1763–1771.

[180] I. Melekhov, J. Kannala, and E. Rahtu, “Siamese network features for image matching,” in *2016 23rd international conference on pattern recognition (ICPR)*. IEEE, 2016, pp. 378–383.

[181] R. R. Varior, B. Shuai, J. Lu, D. Xu, and G. Wang, “A siamese long short-term memory architecture for human re-identification,” in *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14*. Springer, 2016, pp. 135–153.

[182] W. G. C. Bandara and V. M. Patel, “A transformer-based siamese network for change detection,” in *IGARSS 2022-2022 IEEE International Geoscience and Remote Sensing Symposium*. IEEE, 2022, pp. 207–210.

[183] P. Yuan, Q. Zhao, X. Zhao, X. Wang, X. Long, and Y. Zheng, “A transformer-based siamese network and an open optical dataset for semantic change detection of remote sensing images,” *International Journal of Digital Earth*, vol. 15, no. 1, 2022.

The recent development of Artificial Intelligence tools, which enables non-expert users to generate deepfake videos of extraordinary quality is raising increasing concerns about the credibility and the authenticity of digital media. The ability to create such videos, coupled with the widespread use of social media raises enormous concerns about the authenticity and credibility of what we see. The consequences of these fake media could be devastating to our society. In addition to deepfake manipulation, a significant challenge persists within the realm of fake media diffusion, the deceptive geographical recontextualization of photos. This deceptive practice involves the misrepresentation of images in news posts or articles, where an image depicting a war, flood, protest, or earthquake, is claimed to be captured in one specific location in the world, but it has been taken from an entirely different country or city. This manipulation fuels the dissemination of misinformation campaigns by exploiting the powerful impact of images in shaping public perception and belief. In response to these challenges, the thesis focuses on the development of AI-based algorithms. These algorithms aim to detect deepfake videos, a critical step in safeguarding the credibility of digital media. Additionally, the thesis addresses the deceptive recontextualization of photos by unveiling the true geographical locations where images were captured in the world. By tackling these issues, the thesis aims to contribute to the protection of our digital world from the adverse effects of misinformation campaigns and fake media diffusion.

UNIVERSITÀ
DI SIENA
1240