Deep Learning Person Re-Identification Pipeline for CCTV Systems

Garcia Cebrian, Dani

Award date:
2022

Awarding institution:
Polytechnic University of Madrid

Link to publication

Universidad Politécnica de Madrid

Escuela Técnica Superior de Ingenieros Informáticos

Máster en Data Science

Master in Data Science

Trabajo Fin de Máster
Master Thesis

Deep Learning aplicado a Re-Identificación de Personas en Sistemas de CCTV

Deep Learning Person Re-Identification Pipeline for CCTV Systems

Autor / Author: Daniel García Cebrián

Madrid, Junio/June, Año

Trabajo Fin de Máster
Master Thesis
Máster en Data Science
Master in Data Science

Título: Deep Learning Aplicado a Re-Identificación de Personas para Sistemas de CCTV

Title: Deep Learning Person Re-Identification Pipeline for CCTV Systems

Junio / June, 2022

Autor / Author: Daniel García Cebrián

Tutor *Supervisor:*	*Co-Tutor* *Co-supervisor:*
Javier Segovia Pérez	Alejandro Jaldo Serrano
ETSI Informática Departmento de Lenguajes y Sistemas Infórmaticos e Ingeniería de Software / Department of Informatic Languages and System and Software Engineering Universidad Politécnica de Madrid /Polytechnic University of Madrid	F.F Videosistemas Desarrollo / Development

Acknowledgements

I would like to express my gratitude to Alejandro Jaldo, Raúl Encinas and Jorge Villarán for their help and support during the realization of the present project.

Abstract

Tracking people through an extensive CCTV installation with many cameras on different locations is a complex task and multi-target multi-camera tracking technologies are too resource expensive at the moment to be able to meaningfully utilise them in a reasonable time for security purposes. Instead, solutions based around image retrieval techniques such as person re-identification can be used as a valuable alternative tool for forensic analysis in video surveillance systems. The present work covers the complete development of a person re-identification tool for CCTV systems, from a gallery set generation pipeline to integration into the software suite and using state of the art Deep Learning algorithms. Manually labelled person re-identification datasets can be insufficient in their size and diversity of representations for generalization through Deep Learning models, so a combination of aggregated representations and unsupervised learning techniques used on much wider datasets before fine-tuning of the solution proves to be an excellent method to obtain a more robust model.

Abstract (Spanish)

El seguimiento de personas a través de una extensa red de diferentes cámaras en una instalación de CCTV es una tarea compleja, y las tecnologías existentes de seguimiento de múltiples objetivos en múltiples cámaras son demasiado costosas a nivel de recursos como para permitir el uso de estas con fines de seguridad en un tiempo razonable. En su lugar, las soluciones basadas en técnicas de extracción de imágenes como la reidentificación de personas se pueden usar como una valiosa herramienta alternativa para el análisis forense dentro de sistemas de videovigilancia. El presente trabajo abarca el desarrollo completo de dicha herramienta de reidentificación para sistemas de CCTV, desde la generación del conjunto de imágenes hasta la integración de la herramienta en el paquete de software de CCTV y utilizando algoritmos de Deep Learning de vanguardia. Los conjuntos de datos etiquetados manualmente para su uso en tareas de reidentificación de personas pueden ser insuficientes para obtener modelos Deep Learning capaces de generalizar correctamente, debido al limitado tamaño de los datos y la heterogeneidad de estos. Por ello, en el proyecto se comprueba la efectividad del uso de representaciones agregadas y técnicas de aprendizaje no supervisado con conjuntos de datos más amplios antes de ajustar la solución para obtener modelos más robustos.

Contents

List of Figures

Chapter 1

Introduction

1.1 Context

The present Master Thesis deals with the development and integration of a deep learning re-identification pipeline into a CCTV framework. The work subject of the Thesis has been carried out in collaboration with F.F. Videosistemas, a video surveillance company with an interest in computer vision solutions. There is a general interest in the industry for working re-identification tools for video surveillance systems, as consistent multi-target multi-camera tracking (MTMCT) is still a challenge in the field. Such a solution could be used, for example, to easily find a lost child in a train station through its wide and complex camera system when a known instance of them in the system is available (when the parents arrived to the station with the child). Using the last known time and location of the child as a query image we could easily find them later with the help of a re-identification system.

It is in within this framework that the project is proposed, a system capable of performing on-demand re-identification tasks applying state of the art computer vision techniques and fully integrated into the existing CCTV software environment. The system should be able to receive an image from the video surveillance software and return a collection of suspected instances of the desired person within the CCTV system as browsable events by the security personnel.

It is also of notable importance the performance of the proposed pipeline, which should perform the re-identification task within a reasonable, near real-time timeframe, as to allow the security guards to rapidly respond to possible emergencies.

1.2 Objectives

The general objective of the project is the development and integration of a re-identification solution into an existing CCTV software ecosystem (the Geutebrück software suite).

1.2.1 Individual objectives

The general objective of the project can be further broken down into the following individual objectives:

1. Perform person identification from several video feeds at once
2. Maintain consistent person tracking within each individual camera
3. Obtain quality image crops from selected person images
4. Extract the core representation features from person images
5. Find images from the existing backlog similar to a query image
6. Provide an easy to use tool integrated with the existing software for image querying
7. Allow the validation and supervision of the re-identification task
8. Integration of the re-identification results into the CCTV software

1.3 Justification

Deep Learning Computer Vision is one of the hottest fields of Artificial Intelligence right now, advancing at giant steps every other day with uses ranging from autonomous cars to supervision of tasks in logistic centers. One of the areas in which it is thoroughly used is in the field of video surveillance, in which **F.F. Videosistemas** finds itself, with examples such as people counting in venues, intrusion detection, car plate detection and traffic law infringement detection or gun and robbery detection. There are many legitimate uses for computer vision solutions for video surveillance in the name of security, but there are also major public concerns about the intrusiveness and availability of such systems. An example of a controversial use of computer vision is that of face recognition. Face recognition systems are rightfully deemed too intrusive for most video surveillance uses and banned for privacy-related reasons in the European Union under the General Data Protection Regulation. Indiscriminately storing the linked identity and facial features of every person in a public place is simply too intrusive for it's use in CCTV, so in this project a less intrusive solution based on on-demand image comparison. The intended usage of the solution is more akin to **forensic analysis**, with the deep learning re-identification step being only performed on demand by an authorized security person (such as a trained and licensed security guard), and should not be abused for intrusive video surveillance purposes.

1.4 Document Structure

- **Theoretical Framework**: Chapter in which the field of study of the work is contextualized, as well as the minimal knowledge needed to understand the proposed work.
- **State of the Art**: Contextualization of the state of the field and related works and new advancements relevant to the proposed work.
- **Method**: This chapter describes the work carried out for the project.
- **Results, Conclusions and Future Work**: Includes a reasoning on the results of the proposed work, as well as findings from it and a set of possible future directions for the project.

Chapter 2

Theoretical Framework

The field of Computer Science corresponding to the present work is that of Computer Vision, a subset of Machine Learning problems relate to images, an area of Artificial Intelligence. In order to better understand the work, it is necessary to first have a firm grasp on the basics of what Machine Learning (and subsequently Deep Learning) is and the basics of Computer Vision staples such as Convolutional Neural Networks (CNN)[5].

2.1 Machine Learning

Machine Learning can be described as the science of getting a computer to perform tasks it was not directly programmed for, and doing so as a human being would, learning from experience and improving the performance autonomously over time. This kind of behaviour is achieved with machine learning algorithms, that receive data as an input, perform a task on it and learn from the results.

Machine Learning problems can be classically classified into 3 major groups of learning behaviour. The first of these would be **supervised learning**, in which a machine learning model is tasked with the transformation of a set of input data into an output (or outputs) with knowledge beforehand of the desired output. Then the output of the algorithm is compared to the known desired one and a metric of its performance is used (a *loss function*, as could be Mean Squared Error) comparing both, and the algorithm is corrected according to it. This means, during supervised learning the algorithm is trained with real, validated examples of the task to perform. The second major group is that of **unsupervised learning**, which describes a set of problems for which a definite answer may not be available (no real samples as in supervised learning) and the core of the task is to describe or extract information from a set of data. The simplest example of an unsupervised task is clustering, in which an algorithm is tasked with the classification of data points into unknown groups of similar samples. The third major group is **reinforcement learning**, which consists in a series of problems in which an agent has to act within an environment that interacts with it and is influenced by its actions. The agent is then awarded a *fitness score* through the results of the its action within the environment, describing how good or bad of a job the agent is doing. In reinforcement learning there is not definite set of input data, but rather the input is the result of the actions of the agent, and it must learn to operate in the environment through the action-reaction loop.

The problem of re-identification, subject of the project, usually falls under the umbrella of supervised learning, as the algorithm is trained with labeled images of known identities, but such a complex problem often consists of several simpler components, some of which can use unsupervised techniques[6].

2.2 Deep Learning

Deep learning is a branch of Machine Learning based upon the use of artificial neurons and general purpose structures of them called Neural Networks. Deep learning Neural Networks are widely used in a multitude of fields, from medical science, to natural language processing or computer vision, and are often preferred over classic Machine Learning solutions for more complex problems or abstractions.

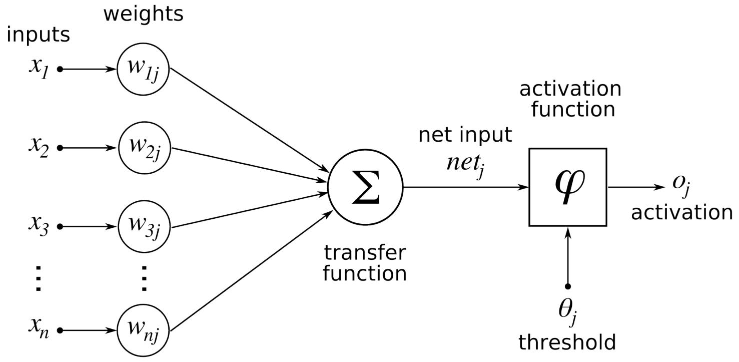


Figure 2.1: Artificial network schema[1]

2.2.1 Deep Neural Networks

A Neural Network is a structure of multiple layers of interwoven artificial Neural Networks. Neural Networks try to find a correct mathematical transformation of the input data into the desired output. The neurons on the first layer (input layer) receive the data, transform it and pass it forward to the next layers (hidden layers), the result of the transformation of the last layers conforms the output of the network, this is called a *forward pass*. After the *loss* of the output is computed, the network is fitted to it through *backpropagation* of the gradient and the weights of the artificial neurons of each layer are updated. The general structure of a simple Feedforward Neural Network and that of an artificial neural network can be seen in Figures 2.2 and 2.1.

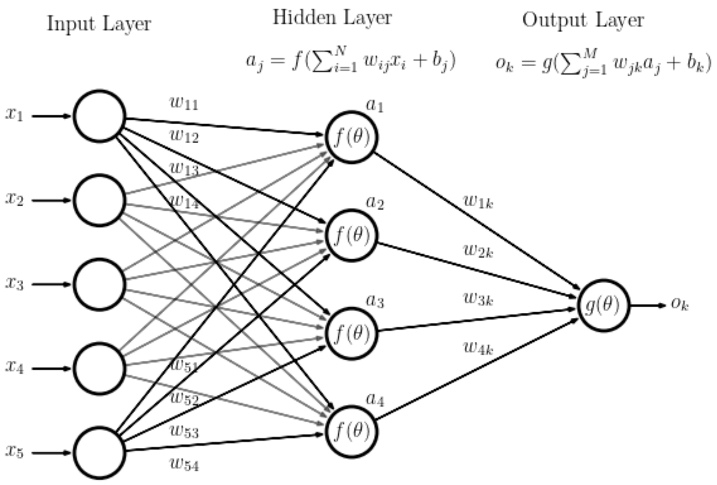


Figure 2.2: Diagram of a Deep Neural Network

2.2.2 Convolutional Neural Networks (CNN)

Convolutional Neural Networks [5] (CNNs) are a type of Neural Networks of widely adopted use in the field of computer vision (or any other field strong spatial relations within the data). CNN make use of convolution algorithms over the input data, which must take the shape of a n-dimensional tensor. CNNs commonly use 2 different types of layers, convolutional layers and *pooling* layers. In convolutional layers, matrix-like structures called *filters* move across all possible positions over the input data (or skip some if stride is used) and compute the output of that position as an operation between the data of the input position and the weights of the filter (also referredto as *kernel*) as can be seen on Figure 2.3. The end result is an output of a similar, albeit slightly smaller shape to the input (if no padding is used) for each used in the layer, which are then stacked giving rise to an output volume. The other kind of layers commonly used, usually between convolutional layers, are pooling layers. In this layers the same principle is applied but with a simpler rule, for example *maxpooling* reduces to the maximum value for each position. The general purpose of these layers is to reduce the dimensionality of the output and concentrate the information into smaller representations (*subsampling*).

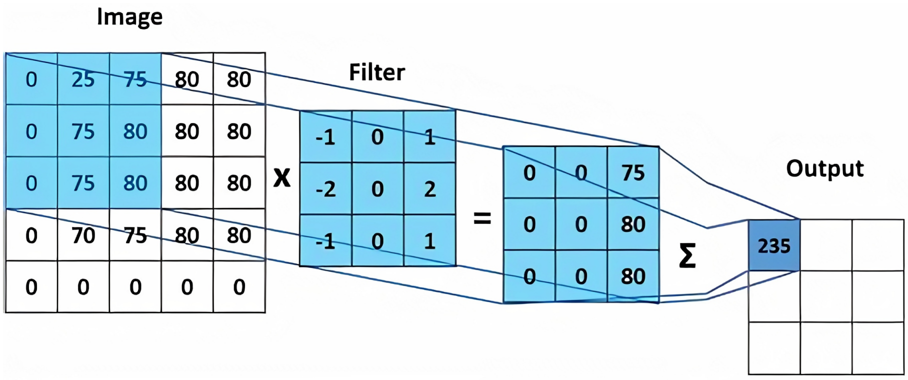


Figure 2.3: Application of a convolutional filter or kernel matrix

2.2.3 Structure of general purpose CNNs

A Convolutional Neural Network usually does not only consist of convolutional and pooling layers, these are often used to obtain an encoded representation of the input data (*an embedding*) that is later flattened into a 1-dimension tensor that can be then used by another kind of NN layer, such as a Feedforward Neural Network (FNN) as in Figure 2.4.

In the field of computer vision, such structure of a convolutional part and a FNN part is widely prevalent, and these two parts are referred to as the **backbone** of the network and the **projection head**. The goal of the backbone is to extract spatial and structural features out of the input data and then the head will learn to perform a task with those learnt features (such as classification).

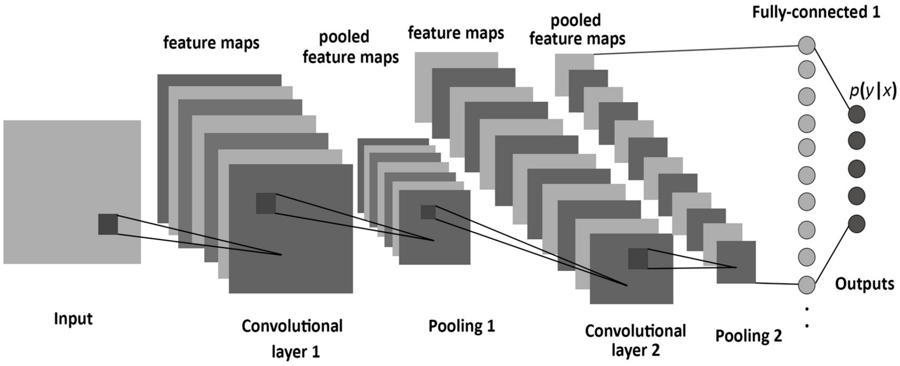


Figure 2.4: Diagram of a Convolutional Neural Network

2.2.4 Common CNN architectures

Although a Convolutional Neural Network can theoretically take almost any form with any number of layers, but there is no point in reinventing the wheel and there are several popular CNN architectures that have already proven their worth in a wide array of uses. As per the Universal Approximation Theorem, given enough complexity a Feedforward network consisting of only one layer is capable of representing any possible function[7]. That would require an enormous layer of many neurons, which proves to be hard to train and prone to overfitting. Multiple layer networks are used and, there has been a push for years now in research towards the use of deeper and deeper architectures, consisting of many layers. Three of the most popular architectures are discussed here.

VGGNet

Visual Geometry Group Networks[8] (VGGNets) is a family of standard Deep Convolutional Neural Networks with an architecture 16 or 19 layers deep (VGG16 or VGG19 respectively). In the original paper, the so-called by the authors ”very deep nets” at the time were proposed as the result of the study of increasingly larger Convolutional Networks for large-scale image recognition, and it was first tested against the ImageNet open classification dataset[9], obtaining top-5 results. The main idea of VGGNets is the use of smaller filters (3x3) across a deeper than usual network at the time. The original proposed architecture consists of a backbone of 13 convolutional and pooling layers, and a head of 3 fully-connected layers. It originally takes as an input an image with a 224x224 resolution and outputs into an array of size

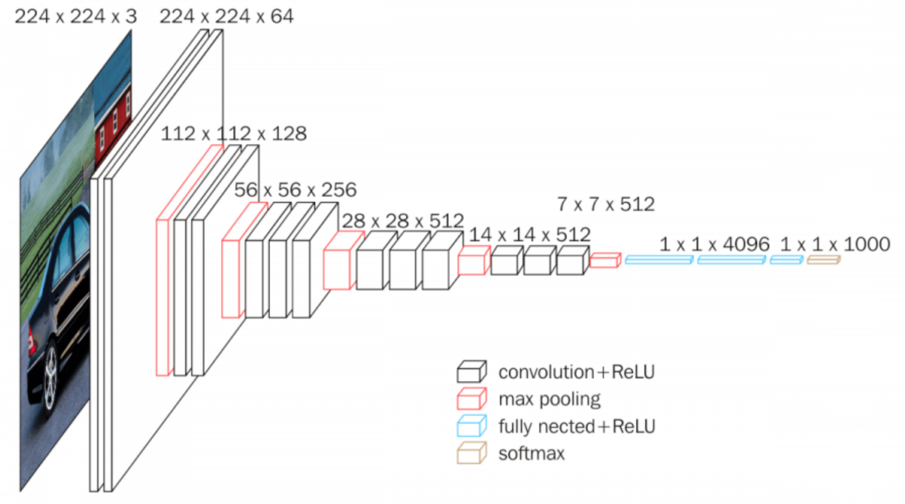


Figure 2.5: Architecture of VGG16

1000, one prediction for every class of the ImageNet dataset.

ResNet

Residual Neural Networks[10] (ResNets) were born as a direct result of the common trend of stacking more and more layers for deeper networks. When increasing the depth of a network, a problem that rapidly arises is that of a *vanishing gradient*, where the repeated multiplication of the gradient during the backpropagation step between layers may result in a gradient infinitely approximating zero and its performance rapidly degrades. ResNets are set to minimize this problem through the use of ”identity shortcut connections”, connections that directly skip several layers and act as ”gradient highways” (Figure 2.6). Since the original paper, there have been several reinterpretations of the concept of the residual block and architecture improvements which has given rise the variants of the architecture, such as ResNeXt[11] os DenseNets[12], but the core remains the same.

EfficientNet

When the need arises for yet more complex networks, existing structures and architectures are usually scaled bigger to match our needs, as could be extending the ResNet architecture from a ResNet50 (50 layers) to a ResNet100 (100 layers). For now, the scaling through deeper networks has been mentioned, but that is not the only way a Convolutional Neural Network can be scaled. It can also be made wider

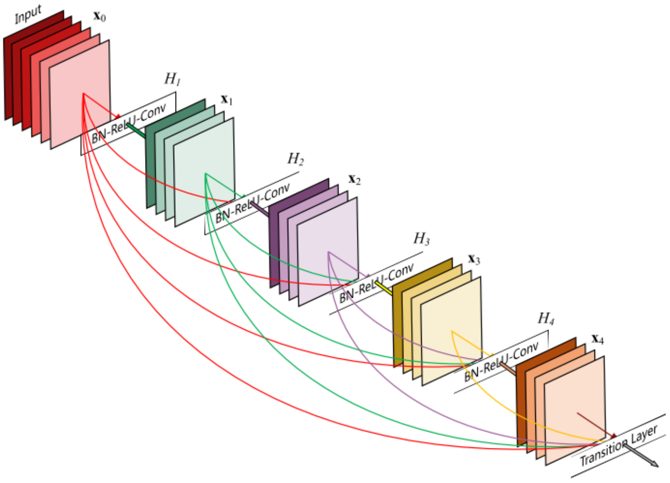


Figure 2.6: ResNet residual connection scheme

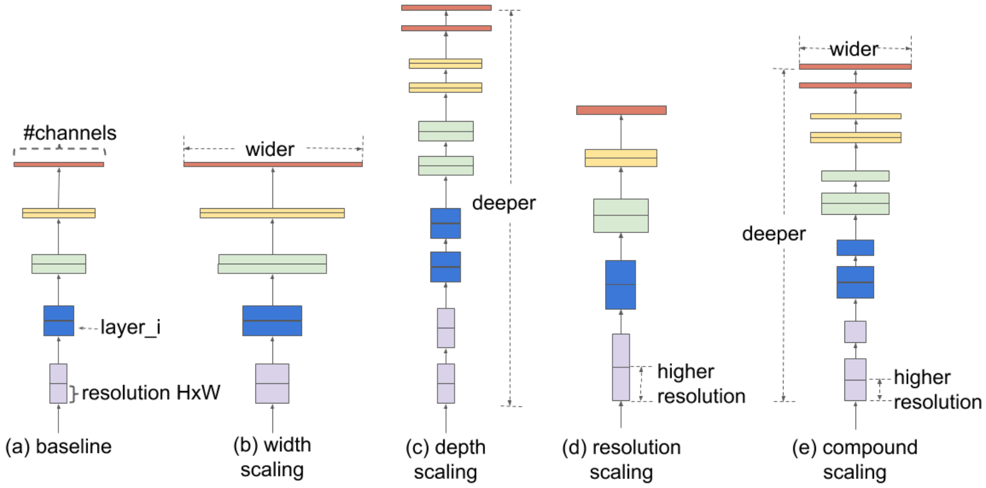


Figure 2.7: Different network scaling methods

instead of deeper, with more filters or neurons per layer, or the input data can be made larger as to take higher resolution images. It is in the constant search for balance of depth, width and input of larger networks that EfficientNets[13] were proposed. EfficientNets were designed with compound scaling of nets in mind (Figure 2.7). The main idea compound scaling is that, if the input data is bigger, then the network will need more layers to be able to extract information from it and more channels to represent the more detailed patterns of the bigger image.

Chapter 3

State of the Art

All across the field of Deep Learning, there has been a push for larger and more deeper networks that better capture complex problems tuned by large companies for general purpose tasks. This trend can be seen on its highest expression on the field of Natural Language Processing, where unreasonably big general purpose models such as OpenAi's GPT-3[14] or Google's PaLM[15] with billions of parameters are taking the lead. Although this gigantic architectures are essentially impossible to train or host for a simple user or researcher , some of them are open to the community and provide the opportunity to use the networks pre-trained by a larger company and apply them for their own purposes, be it either as they come or via fine-tuning or transfer-learning techniques. Although not to that extreme, this trend has also permeated into the Computer Vision field too. Since the ImageNet movement in 2014, most Computer Visions projects make use of open models pre-trained on general purpose image datasets (like ImageNet or COCO[16]) and apply them to their own uses. On the specific subject of person re-identification, there also exists large open datasets available for use. The most prevalent in the area are the DukeMTMC[17], Market1501[18], CUHK03[19] and MSMT17[20] datasets.

Although not the main focus of work of this Master's Thesis, the present work touches upon several different specific topics within the Computer Vision field and makes use of them for the re-identification pipeline. Due to that this chapter will also briefly summarise the SOTA of related works used on the project.

3.1 Human detection and YOLOv5

YOLOv5[21] is the latest generation of a family of general purpose object detection architectures and CNN models by Ultralytics. YOLOv5 is trained on Microsoft's Common Objects in Context[16] (COCO) dataset. The main idea of a YOLO model is that of "**Y**ou **O**nly **L**ook **O**nce", predicting both the class of an object and it's bounding box in only one run of the image. While other object detection models such a R-CNN[22, 23] or RetinaNet[24] provide highly accurate object detections and bounding boxes through repeated passes over the input image, YOLOv5 trades detection accuracy to solve the main problem of mentioned models, the problem of speed for real-time object detection. For its predictions, the YOLO algorithm divides the input image into a grid, usually of 19x19 cells, and for each cell predicts the probability that the cell contains an object of a certain class, the center point of the anchor box for the class in the image, and the width and height of the k possible bounding boxes. After that is done, redundant bounding boxes for each object are eliminated through a non-max suppression step[25] and the output predictions are finalised.

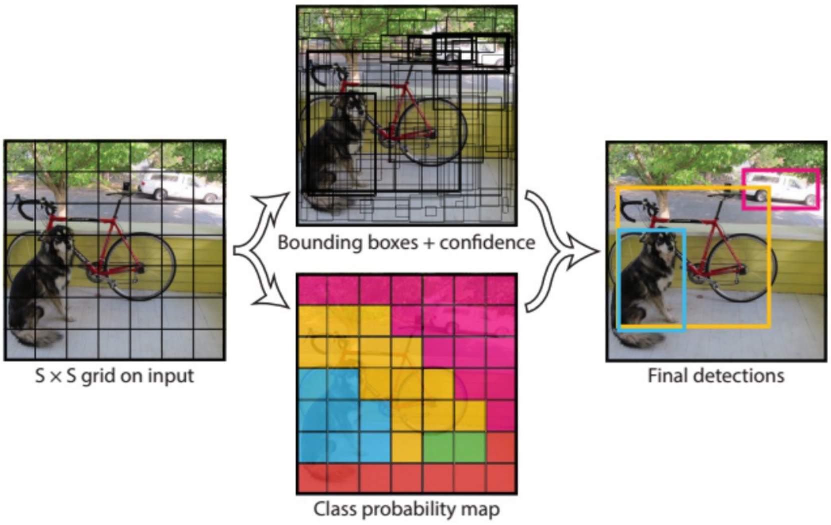


Figure 3.1: YOLO algorithm detection scheme (from YOLOv2 paper[2])

The detection speed of the YOLO family of algorithms has made them a staple for real-time object detection, specially on lower-end hardware or if the number of frames per second is too large (as in multi-camera setups).

3.2 Pose estimation and High-Resolution Networks

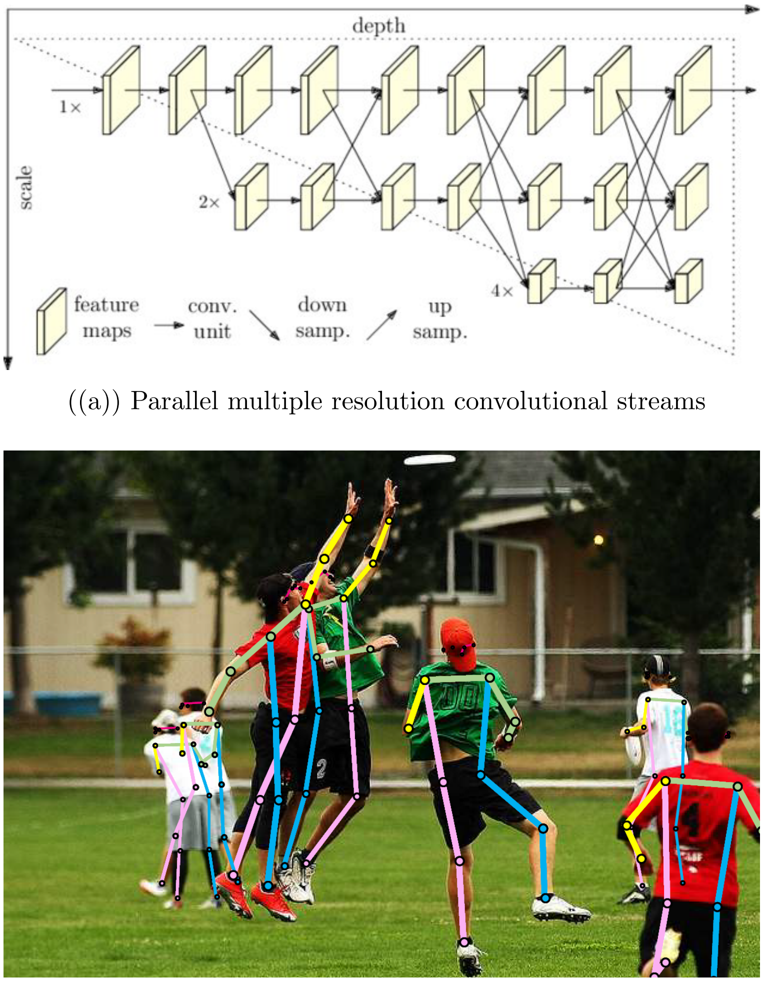
High-Resolution Network[27] (HRNet) is a state of the art algorithm developed by Microsoft's research team designed from scratch to make use of high resolution convolutional streams for precise representations, in contrast to many other ConvNets that use relatively low resolution representations designed for image classification. The architecture of an HRNet starts with a high-resolution stream of convolutional layers, which is gradually forked into lower resolution ones while keeping the first one, connecting them at intervals allowing the fusing and exhange of information across the parallel streams of different representations (Figure 3.2(a)) . While HRNets were also first proposed for semantic segmentation and object detection, in the context of the present work we are interested in the use of HRNets for human pose estimation (or keypoint detection, as it may be known). Human pose estimation is usually a two step process, the first one is that of human detection and retrieval of detection limits (bounding boxes) from the input image, then on the areas of the image with people on them the task of the model is to estimate the location of keypoints of the human body such as eyes, mouth, knees or elbows in a regression problem. In the end, the output of the net is the location of all those keypoints of the human body that allow to build a model of the skeleton or position of the person on the image (Figure 3.2(b))

3.3 Instance retrieval

The problem of instance retrieval is that of, given an object from a query image, matching the query object to objects represented inside of images in a gallery set. The first obvious usage of instance retrieval is in **person re-identification**, but the same problem can be applied to vehicle or fashion re-identification, copyright infringement detection or clinical diagnosis.

In order to be able to match objects in an image to others we first have to transform them into similar and comparable representations. It is here where most instance retrieval solutions make use of Deep Learning techniques, in order to compute the information of an image by transforming it into n-sized vector representations of the encoded information (the *embedding*).

The main idea is to create such a transformation that creates similar vector rep-



((a)) Parallel multiple resolution convolutional streams

((b)) Pose estimation output sample [26]

Figure 3.2: HRNetwork architecture and pose estimation

resentations from images containing similar objects, and dissimilar vector representations for dissimilar objects. If the representation task was performed successfully, we can then compute the similarity of the query object and the objects in the gallery set by comparing their vector representations in the n-dimensional space, with the most common similarity method being the computation of the cosine of the angle formed by the vectors (similar vectors will point in roughly the same direction in the n-dimensional space). When the time of image retrieval comes, the embedding of the query object is compared against all other embeddings in the gallery set and the most similar ones are returned.

3.3.1 Triplet Loss Function

While a cosine similarity will provide us with a metric of the similarity of two given samples with a trained network, for the purposes of training the model we still need to use a proper, derivable loss function. For the task of image comparison there exist several possible loss functions, such as contrastive loss[28] or circle loss[29], the most popular one is the Triplet Loss function[30], with usage not limited to Computer Vision but also in the training of siamese BERT[31] models for natural language processing[32].

The main idea behind the Triplet Loss is to build triplets of samples consisting of an anchor image A, a positive sample P (an image similar to the anchor image) and a negative sample N (an image dissimilar to the anchor) and score the similarity between the samples and the anchor. The objective of the Triplet Loss function is to minimise the distance between the representation of the anchor A and the positive sample P while maximizing distance with the negative sample N (Figure 3.3).

In detail, the Triplet Loss function is formulated as follows:

$$\mathcal{L}_{Triplet} = [||f(A) - f(P)||_2^2 - ||f(A) - f(N)||_2^2 + \alpha]_+ \tag{3.1}$$

where $[z]_+ = max(z, 0)$, f is the embedding function and α is a margin parameter.

3.3.2 Self-supervised learning, SimCLR and MoCo

Obtaining enough labelled data for training a supervised algorithm in the context of person re-identification can be difficult, but there are some techniques to leverage the power of unsupervised training for contrastive learning[33]. The most important

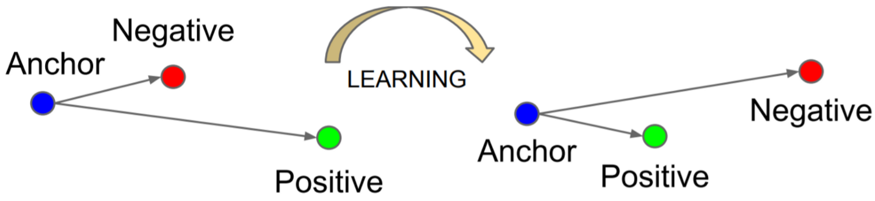


Figure 3.3: Visual contextualisation of Triplet Loss function

of them make use of **"self-supervised" learning**, an algorithm that creates its own positive labels from unlabelled samples generally via data augmentation. The state of the art in self-supervised training for person re-identification comes from SimCLR[34, 35] and MoCo[36, 3, 37].

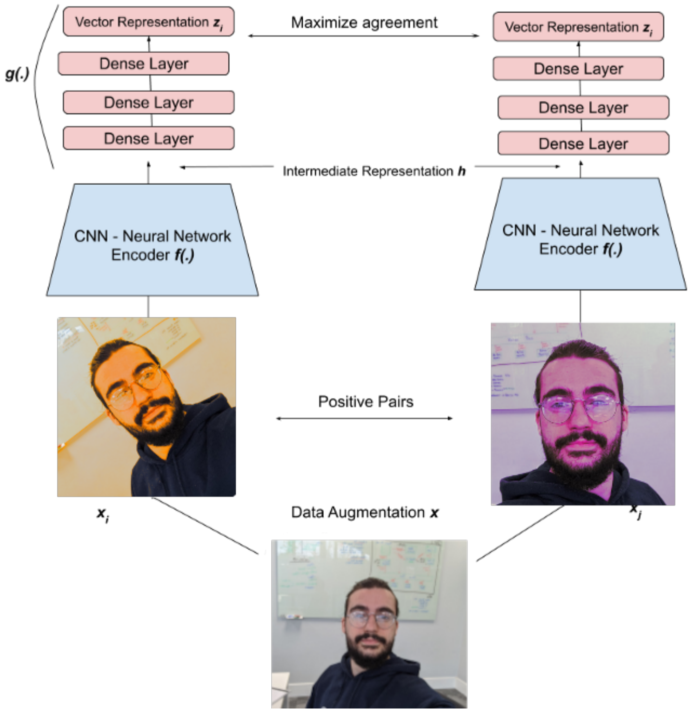

g(.)
Vector Representation z_i
Maximize agreement
Vector Representation z_j
Dense Layer
Dense Layer
Dense Layer
Dense Layer
Dense Layer
Dense Layer
Intermediate Representation h
CNN - Neural Network Encoder $f(.)$
CNN - Neural Network Encoder $f(.)$
Positive Pairs
x_i
Data Augmentation x
x_j

Figure 3.4: SimCLR v2 training step

SimCLR is a framework developed by Google for self supervised contrastive learning. The main training loop of the SimCLR framework can be explained in three steps. The first one is, for each sample in the dataset, obtain two different similar images through the random combination of data augmentation techniques (for example, recoloring+resizing, cropping+rotating, etc.). The second step is obtaining the embeddings of the augmented samples, and the last step is to maximize the similarity of the embeddings of the pair by minimizing a contrastive loss function (Normalized Temperature-Scaled Cross-Entropy Loss or "NT-Xent"[38] is usually used by SimCLR). An schema of the SimCLR self-supervised training step can be found in Figure 3.4.

The Momentum Contrast (MoCo) framework developed by Facebook builds upon the first iteration of SimCLR and implements the use of a dictionary of embeddings of the dataset, then using a queue of mini batches of the dictionary during training to compare to query image to. MoCo uses a momentum encoder to represent the gallery set queue and updates it by enqueing and dequeing minibatches, but the backpropagation is ran through the simple encoder rather than the momentum encoder. A simple overview of the idea is shown in Figure 3.5.

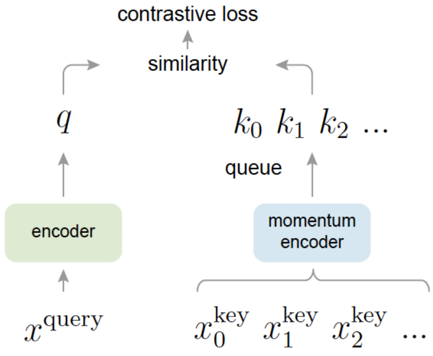


Figure 3.5: Momentum Encoder scheme (from Moco v2 paper[3])

The latest state of the art solution to use this kind of self-supervised learning techniques for person re-identification is the publication "Unsupervised Pre-training for Person Re-identification"[6], in which a backbone was built through a fully unsupervised pipeline (using MoCo v2). The solution also includes the generation of a new dataset for re-identification from images and videos from the Internet called the LUPerson dataset. The LUPerson dataset claims to be the biggest unlabelled dataset for person re-identification with over 4 million images of over 200,000 identities obtained crawling from around 750 streetview Youtube videos of the top 100 biggest cities in the world.

3.3.3 Centroid Triplet Loss

While the Triplet Loss Function has proven to be mostly superior to other approaches for re-identification tasks since it's introduction in 2018, it still suffers from some problems[39]. Some problems of the Triplet Loss function come as a result of using hard negative mining[40, 41]. Hard negative mining is a useful training technique consisting in creating training batches made up of only very negative samples, in the case of contrastive learning that would be done in a batch of only informative triplets for optimizing the generalization of trained features. Though useful, when using Triplet Loss the technique can lead to bad local minima and is usually very computationally expensive as distance needs to be computed between all samples in the batch to find the "hardest" negatives. Apart from that, Triplet Loss can also be heavily affected by outliers and noisy labels, due to the nature of point-to-point loss functions[42].

As a way to minimise these problems, the authors of "On the Unreasonable Effectiveness of Centroids in Image Retrieval"[4] propose the usage of centroid representation of same-identity samples when comparing representation distances by the usage of a Centroid Triplet Loss (CTL) function. Centroids are an aggregation of objects into a single representation, in the case of person re-identification each unique identity of the training set should aggregate into a single centroid, providing a more robust representation of said identity and reducing the computation and memory usage requirements by reducing the gallery set when querying. The proposed Centroid Triplet Loss function is formulated as:

$$\mathcal{L}_{Triplet} = [||f(A) - c_p||_2^2 - ||f(A) - c_n||_2^2 + \alpha_c]_+ \tag{3.2}$$

where $[z]_+ = max(z, 0)$, f is the embedding function, α_c is a margin parameter and c_p, c_n are the centroids of positive and negative samples respectively.

Models using the CTL approach (as the one on Figure 3.6) have proven capable of outperforming outperform previous methods, achieving rank-1 results on the DukeMTMC dataset (at the time of publication, in) while lowering the usage of computational resources.

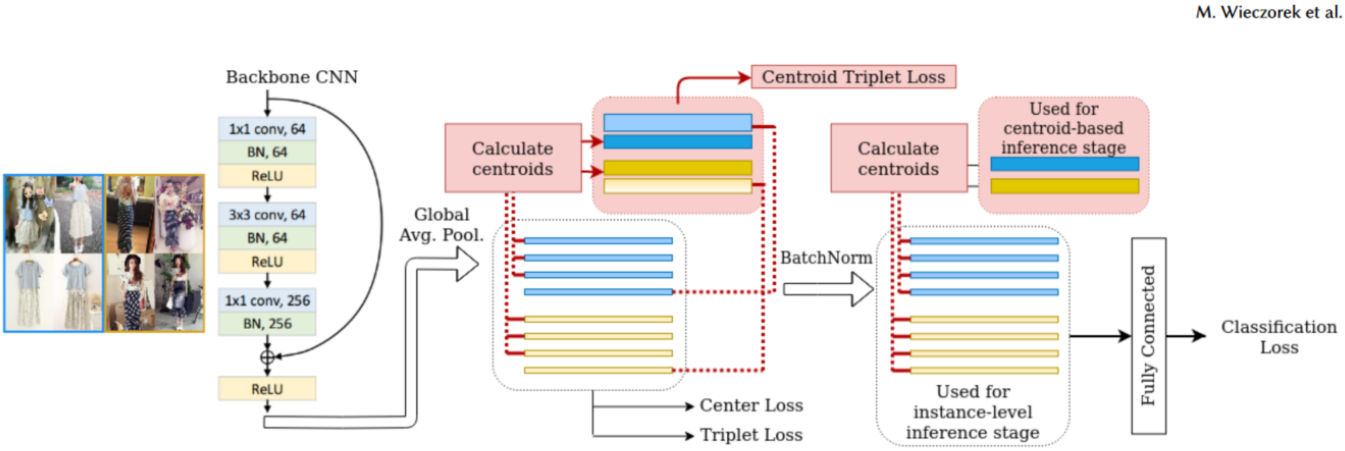


Figure 3.6: Architecture of a proposed CTL model on the original paper[4]

Chapter 4

Method

For the realization of the proposed work I mainly used a workstation running Ubuntu 20.04 LTS with a Intel(R) Xeon(R) Gold 5122 CPU, two Nvidia Quadro RTX 4000 GPUs and 64GB of RAM through a remote connection. As for tools used, most of the code regarding the re-identification pipeline was written in Python using the PyCharm Professional IDE by JetBrains, with some experimentation done using a Jupyter Lab server. For the integration into Geutebrück's software suite, `C#` and Visual Studio 2022 Professional were used.

As stated in Chapter 1, the main objective of the work is that of developing a person re-identification pipeline and it's integration into a CCTV system. The tool should allow authorized users to select a person within the native viewer of the CCTV software, get a view of the image retrieval results, and if in agreement with the results commit them into the system as events.

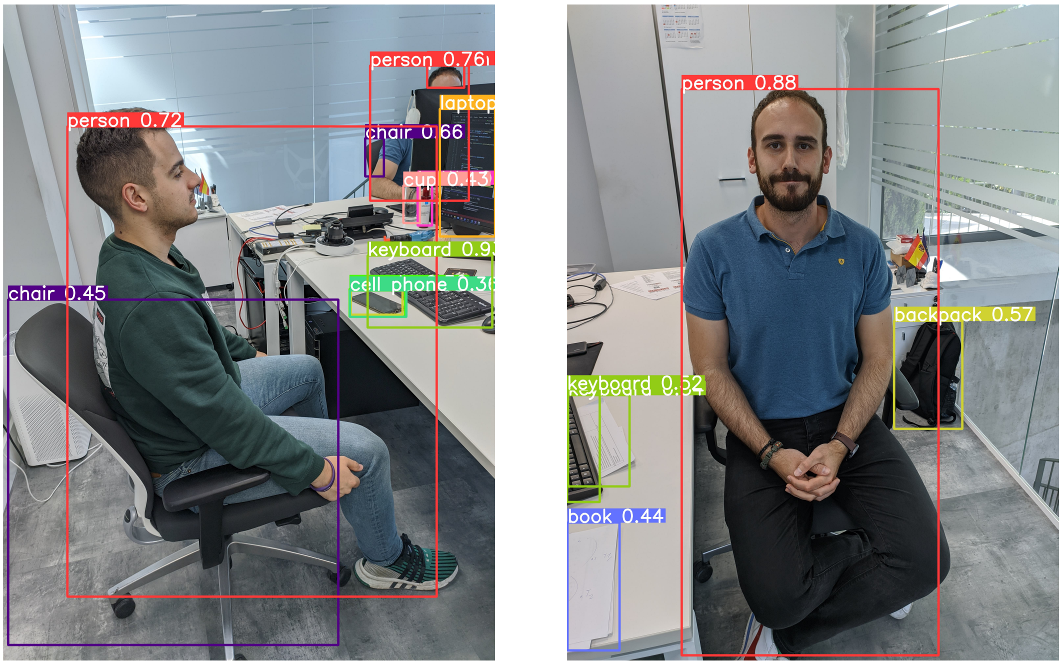
4.1 Gallery set generation

As with any Data Science related project, one of if not the most important parts is the data used. In the case of the proposed ReID pipeline, the data should be a collection of person images within the used CCTV system. The use case of the solution will be that of on-demand re-identification, but even if the image retrieval is not performed on real-time, the performance constrains of the system (the search should not take more than about a minute) demand the gallery set to be kept somewhat up-to-date. While a completely forensic approach to the problem, in which saved video is analysed a posteriori is possible, that would exponentially increase the computing time of the solution. Due to the possible time-critical uses of the technology, the actual approach used is that of near real-time anonymous person detection (no additional personal information is kept other than the image, which is already being recorded in the CCTV system). For person re-identification purposes, images are only relevant within the same day, as opposed to face recognition technologies a simple change of clothes could be enough to trick the system. The scope of the solution finding other instances of a person within the camera system today. Due to this, the gallery set is deleted and renewed on a daily basis.

The gallery set of the problem at hand should be people on the camera today, so the first step is to obtain access to cameras within the system. Most if not all video cameras mounted on CCTV systems nowadays are IP cameras, that is, their main mode of operation is via network streaming to a central recorder system. These cameras work under the Real Time Streaming Protocol (RTSP)[43], an application-level protocol for the real-time transfer of data over IP. Luckily, the Python implementation of the popular OpenCV library[44] provides an easy way to connect to these RTSP streamings and catch frames for their processing.

4.1.1 Person detection

In order to build the gallery set, person detection is performed in a near real-time fashion over the live RTSP streams. The selected model for person detection detection was a YOLOv5[21] model. The reasoning behind the selection of a YOLO model is mainly the detection speed, which needs to be fast enough to be capable of performing the detection task on multiple cameras at the same time without lagging behind the RTSP streaming. The specific model chosen from the YOLO family is a YOLOv5m6 model, a medium-sized iteration of the YOLOv5 architecture (hence the 'm') which takes as input 1280x720 images (most CCTV cameras natively operate around this range). The model is used with PyTorch[45], using the official off-the-shelve Torch Hub implementation[46](under the GNU GPL3 License) for object detection, pre-trained on the COCO dataset. The pre-trained model defaults to the detection of the 1000 classes found on the COCO and ImageNet datasets (Figure 4.1), so it is configured to only detect persons.



((a)) ((b))

Figure 4.1: YOLOv5 sample detections

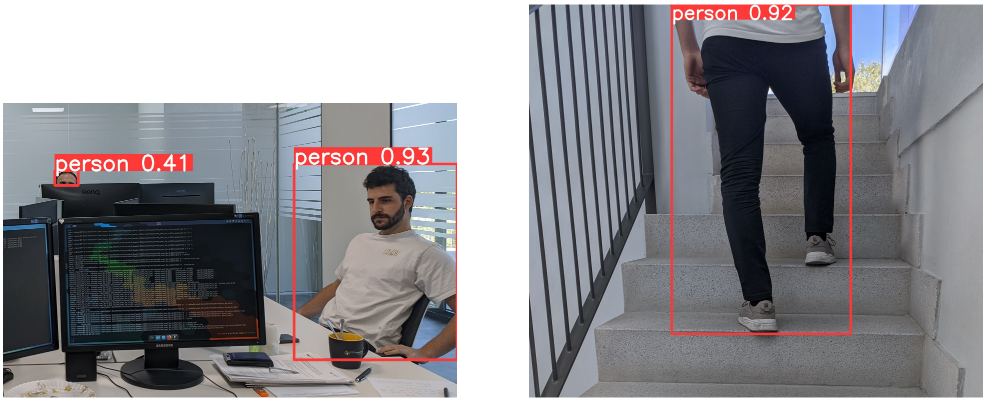
4.1.2 Tracking and detection grouping

The YOLOv5 model only performs object detection on an image by image basis, not on a video. This means that only using the model's detection we cannot ensure the persistence of an identity over several frames. In order to carry over the information of a unique detection over time, a tracking algorithm has to be used. In this case, a Simple Online and Realtime Tracking with a Deep Association Metric[47] (DeepSORT) model was used (GNU GPL3 License). As a brief explanation, the DeepSORT algorithm is a deep learning extension and reimplementation of the popular SORT[48] algorithm, which tracks identities on the fly for given past and present detection of an object based on rudimentary data association and state estimation techniques.

Using DeepSORT, we can provide frame-by-frame detections with a persistent ID in the context of the camera for a continuous shot of an object, so can limit and space the detections of a given person for the gallery set. For the purpose of this project, person detections were saved the first time an object enters the frame (see 4.1.3) and once every minute since then, grouping subsequent detections of the same object. During testing of the method, the Yolov5+DeepSORT combination managed to process around 34 frames per second at a resolution of 768x1280 distributed in 9 cameras, equating to around **3.8 scans per camera per second** on a single Quadro RTX 4000 GPU.

4.1.3 Detection quality and pruning

A sine qua non condition for a successful re-identification is to ensure the quality of representations in the gallery set. The first step to ensure this is to only save person detections above a certain confidence threshold (0.72 was used in the project). While continuous object detections in the project are only saved once per minute, exceptionally an extra detection frame might be saved if the confidence level of the detection goes significantly above the highest historical detection for an identity (an increase of more than 0.05), to try and capture the clearest image possible.



((a)) Left detection only shows top of head

((b)) Only lower body can be seen

Figure 4.2: Example detections leading to bad quality shots

While a good start, the aforementioned method is not enough to ensure the quality of the gallery set for person re-identification. Instances coming from a YOLOv5, while enough to perform object detection with a high confidence, might not hold to the quality necessary for the project. An example of these bad shots can be seen on Figure 4.2, where the cropped image of the detection will not be enough to perform re-identification.

Following the approach taken during the creation of the LUPerson dataset in

"Unsupervised Pre-training for Person Re-identification"[6], the problem of bad quality shots was solved in this project through the use of pose estimation methods. Pose estimation algorithms are generally much slower than detection algorithms, so a lightweight model was chosen for the project in the form of the MMCV[49] implementation (Apache-2.0 license) of a Lite-HRNet[50], a smaller and more efficient version of a High Resolution Network using the efficient shuffle blocks presented on ShuffleNet[51].

Real-time person detections are put into a queue that periodically runs the pose estimation algorithm on the images of the queue every ten seconds (the performance of the Lite-HRNet is much better in batches). As in the LUPerson dataset, detections are then only kept if the following conditions ensure:

- Head is present in the image (no need for the face facing the camera)
- Upper body is mostly present in the image
- Lower body is partially present: hip is present or at least one knee
- Height to width ratio is lower between 1.5 and 5.
- Bounding box is at least 48 pixels wide and long.

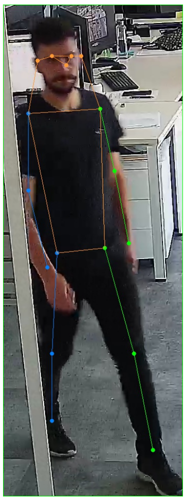


Figure 4.3: Valid detection

An example of a valid detection according to the pose estimation method can be seen in Figure 4.3, head, upper and lower body are present, and the image meets the size and ratio requirements. If and only if all former conditions are met, the image is kept for the purposes of re-identification. Person detection bounding boxes can sometimes be bigger than expected, with a significant part of the image consisting of the background. As a way to reduce background noise in the image, detection crops use the pose estimation to further adjust the bounding box, removing unnecessary background from the image.

4.2 Re-identification step

For the re-identification task, the chosen approach was to leverage the context information of grouped detections of an identity using the Centroid Triplet Loss method. The model used follows the architecture from Figure [?] from the official implementation of the paper ”On the Unreasonable Effectiveness of Centroids in Image Retrieval”[4, 52] with a ResNet50 CNN backbone. The model takes as input the normalized images of the detections resized to 256x128 and in 3 channels (RGB) in batches and outputs the embeddings in tensors of size 2048, which form the gallery set. As stated before, the chosen method uses same-identity images to form a single representation of an entity with a more robust centroid, reducing at the same time the size of the gallery set. When the time of image retrieval comes, the query image is passed through the net and compared to the centroids on the gallery set through cosine similarity. Finally, the most similar centroids (those with a similarity under a certain threshold) will be returned for the query image.

4.2.1 Dataset

The data used for the fine-tuning of the network is the Market-1501 open dataset. DukeMTMC is the other dataset commonly used for person re-identification and referenced in most research, but in this case it was not used as it has been recently pulled by the Duke University due to privacy and human rights concerns[53].

Market-1501 is an image dataset containing 32,668 manually annotated bounding boxes of 1501 different identities over 6 different cameras. The dataset was collected in a supermarket in Beijing, China. Each identity is present in at least 2 different cameras and there is an average of 20 images per identity. The data also contains 2,793 images labelled as distractors (low quality and invalid detections).

The dataset is already provided with a train/test split, where 19,732 images of 751 identities are used for training and 12,936 images of 750 identities are reserved for testing.

4.2.2 Training

Training a re-identification model is a bit different from more traditional Deep Learning methods. During supervised training, a fully connected network is used as a projection head to turn the training into a classification problem. The projection head is actually only used during training to fine-tune the convolutional layers, after training is over the projection head is removed and the desired output is the flattened output of the CNN backbone (the embedding).

For computer vision projects, most research has shown the unquestionable advantage of usind pre-training weighs from a network trained on generalist tasks as object classification. The chosen backbone for the model is a ResNet50 due to it's prevalent use in research and the availability of pre-trained weighs for the architecture. With this versatile backbone in mind, **2 main training experiments were carried out**. The first one follows the approach of the original Centroid Triplet Loss publication and starts with pre-trained weights of a ResNet50 network **trained on the ImageNet dataset**. The second one maintains the same training parameters but applies uses the weights published by researchers of the University of Science and Technology of China **pre-trained on the LUPerson dataset using the MoCov2 unsupervised training framework for person re-identification**.

4.2.3 Detailed parameters

The projection head consists of a fully connected network with a single layer of 751 neurons (one for each class of the training set) whose input is the flattened output of the ResNet50 backbone. A much more detailed diagram of the ResNet50 backbone used can be found in Anex D: ResNet50 backbone diagram. The loss function is a compound function consisting of: a classification loss function (categorical cross-entropy), standard Triplet Loss function and Centroid Triplet Loss function computed on the batch-normalized embeddings all weighted equally. A batch size of 128 images was used. Although commonly used in image retrieval tasks, resampling is not used here to fill in batches as that would just include noise on identity centroids. The optimizer used is Adam[54] with a learning rate of $3.5e^{-}4$, the training was performed for 120 epochs and the learning rate is decayed by a factor of 10 after epoch number 40 and 70. Input images are first resized to 256x128 and normalized with means $[4.485, 0.456, 0.406]$ and standard deviations $[0.229, 0.224, 0.225]$ per channel ([R,G,B]).

The model and training steps are implemented in PyTorch and the Lighting[55] framework and each training took around 3 hours, distributed across the 2 GPUs of the workstation.

4.2.4 Model evaluation

Image retrieval systems are often evaluated base on their mAP and top-k metrics. While talking about image retrieval, precision is defined as the ratio of retrieved images that are correct or relevant over the total retrieved images (Formula 4.1)

$$Precision = \frac{|\{relevant\ images\} \cap \{retrieved\ images\}|}{|\{retrieved\ images\}|} \tag{4.1}$$

For understanding average precision (AP), the concept of AP@k is useful to understand. When calculating AP@k, we compute the precision of the first k images retrieved for a given query and score them as:

$$AP(q)@k = \frac{1}{GTP} \sum_{i=0}^{k} P@i \cdot correct(i) \tag{4.2}$$

with GPT as the the total number of ground truth positives, *correct(i)* as a function that returns 1 if retrieval i is correct and 0 otherwise and given:

$$P@i = \frac{number\ of\ correct\ retrievals\ up\ to\ i}{total\ number\ of\ retrieved\ images\ up\ to\ i} \tag{4.3}$$

As an example, imagine we tried to compute AP@5 for an image retrieval with a total of 3 ground truth positives (there are 3 correct images in the gallery set). We would perform the query and consider the first 5 results of the retrieval. Now imagine that only the first, fourth and fifth results are correct, with the second and third belonging to an incorrect identity. Then the AP@5 for the query q could be expressed as:

$$AP(q)@k = \frac{1}{\mathbf{3}} \cdot (\frac{\mathbf{1}}{\mathbf{1}} + \frac{0}{2} + \frac{0}{3} + \frac{\mathbf{2}}{\mathbf{4}} + \frac{\mathbf{3}}{\mathbf{5}}) = 0.7 \tag{4.4}$$

Notice how the AP@10 or AP@20 for the query would in this case yield the same results as AP@5, as all ground truth positives have already been retrieved. AP@k is also commonly referred to as **top-*k*** results.

Overall $AP(q)$ is then computed as $AP(q)@n$ given n as the total number of images in the gallery set. Finally, the mean average precision (mAP) is the mean of $AP(q)$ for all q in the test queryset Q.

$$mAP = \frac{1}{|Q|} \sum_{q \in Q} AP(q) \tag{4.5}$$

4.2.5 Test results

Evaluation was performed through the official implementation of the Centroid Triplet Loss code[52] to avoid errors and directly compare the results. The models are evaluated for their performance metrics when querying against the test set, namely mean average precision (mAP) and the top-k results. Table 4.1 showcases the mAP, top-1, top-5 and top-10 results for the current SOTA algorithm as of June 2022 (Centroid Tripled List[4]) taken from the paper, the models metrics published on the LUPerson paper, my replication of the SOTA algorithm and the training of the Centroid Triplet Loss using the proposed unsupervised backbone trained on the LUPerson dataset[6] for the Market-1501 test set.

Model	**mAP**	**top-1**	**top5**	**top-10**
Official CTL published[4]	0.983	0.980	0.989	0.991
Unsupervise backbone[6]	0.9621	-	-	-
Project CTL training	0.9834	0.981	**0.991**	0.991
CTL+Unsupervised backbone	**0.9867**	**0.984**	**0.991**	**0.996**

Table 4.1: Test results on the Market-1501 dataset

From the obtained results, we can observe that the result metrics from the training with the Centroid Triplet Loss approach are the same as in the publication, within margin of error. We can also attest to the potential improvement of the test metrics when using the unsupervised backbone trained on LUPerson dataset. **The model combining both approaches, obtains better results metrics on the Market-1501 dataset than the current SOTA.**

4.2.6 Applied usage experiment

During development, a Jupyter Lab server was used to test the algorithm on a gallery set of collected images of a day in the central office of F.F. Videosistemas with 9 cameras. Obtaining objective and representative metrics of the actual performance of the re-identification step applied on-site and outside of curated datasets is a really hard task, due to the variability of the set, a very limited number of identities and a lack of human-labeled samples, but the empiric results of the experiment were satisfactory enough. As such, a couple of examples of the re-identification task test in Jupyter Lab can be seen in Anex A: Person re-identification experiment samples on Jupyter Lab. It should be noted though, that **good metrics on a specific dataset do not necessarily mean good real-world performance**. As such, the model was empirically observed to struggle with re-identification when the images vary significantly in illumination conditions (for example, a well lit room versus a dark corridor).

4.3 Graphical user interface

One of the main objectives of the present project is to deliver an easy-to-use tool, and for that purpose a graphical user interface (GUI) is needed. While the integration within the Geutebrück software suite is another objective of the project, the development tools of the suite (the official **Geutebrück SDK**[56]) do not provide a way to modify the GUI itself of the viewer software *G-View*. Instead, the re-identification tool developed on the project relies on a support platform complimentary to Geutebrück's software for its graphical user interface.

The platform used is a webapp previously developed for general usage and configuration purposes of the Geutebrück suite by F.F Videosistemas. The webapp is based on the Django[57] framework for web development, which uses the Python language for the backend, which allowed for the porting of the inference code to the webapp with minimal effort. The Django framework follows a model-view-controller (MVC) pattern and the usage of the tool is behind a required authentication step as to only allow authorized users.

Web development is not the focus of the project, so the detailed nuances of the web development will go largely unexplained, but I will provide an overview of the implementation.

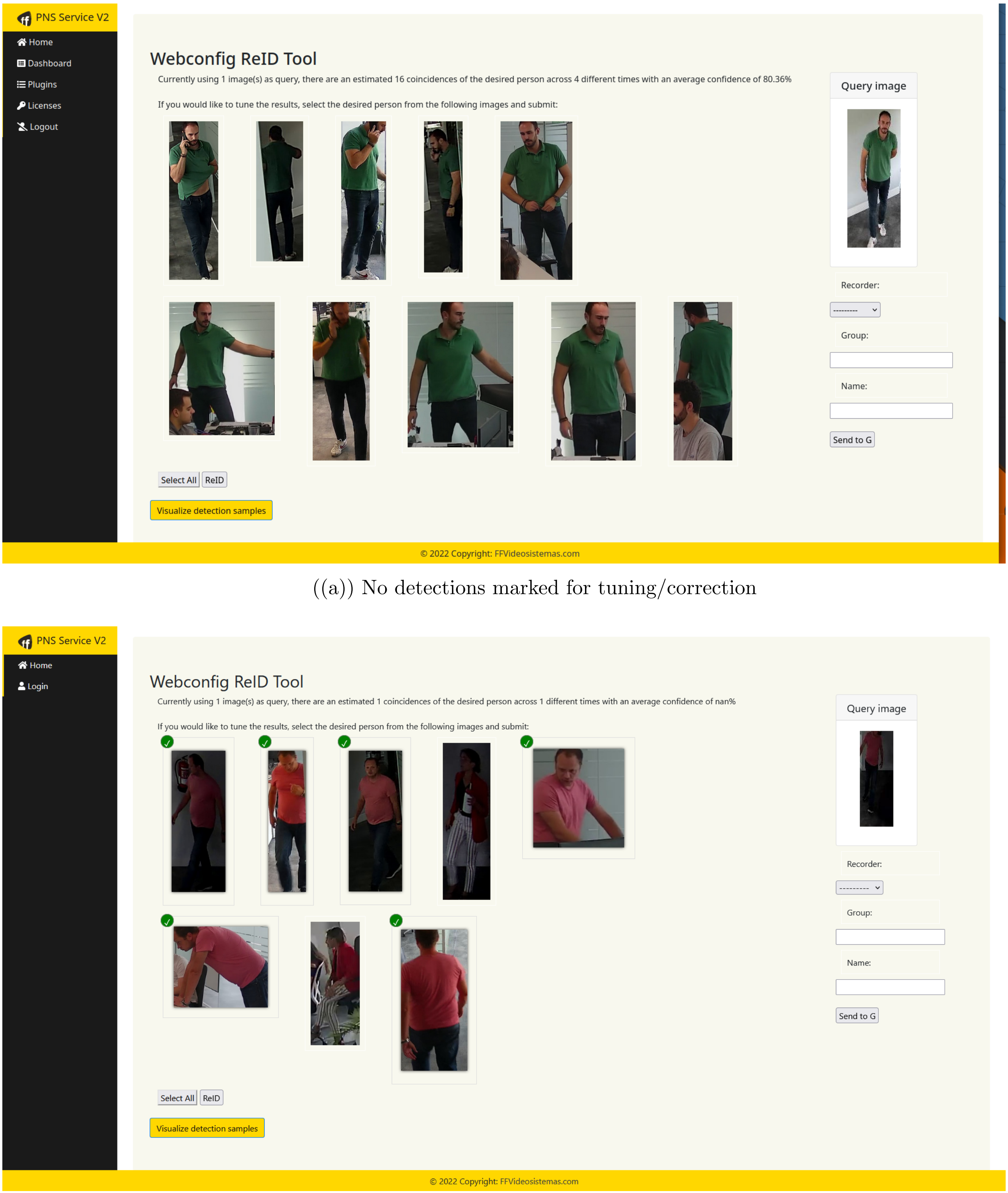
The re-identification tool is implemented as a standalone Django app within the web project, meaning it is largely independent from the rest of the site and can be ported to other Django projects. The main endpoint for the tool is found under `{server_ip}/reid` and contains the general interface for the user. Two other endpoints are implemented, one is `{server_ip}/api/make_embeddings` which is used as a REST API endpoint to manually update the embeddings (when `{server_ip}/reid` is first accessed during a session this is also called upon). The other implemented endpoint is `{server_ip}/uploadImage`, which is used to get the query image into the server (either manually or as an API call). The Django server is configured to correctly serve the detection images and makes use of the GPU for inference and acts as a wrapper around the re-identification model for image retrieval.

Main page

Figure 4.4 shows screenshot of the main page of the re-identification tool. When accessed trough a GET request, the server computes the embedding for the query image and refreshes the gallery set (see 4.4 for how the query image is obtained), then image similarities are computed and the gallery set is ranked. Images under a cosine similarity threshold of 0.4 (60% similar) are first considered as relevant matches. The web page shows the query image being used on the top right of the site. The text also informs the user about the number of current matches of the re-identification step as well as the average similarity of those matches. The visualize button shows a preview of the currently retrieved images in a modal window. From here, the user can correct and/or expand the detections (see Section 4.3.1) or send them to the CCTV system (see Section 4.4) by selecting a recorder system and optionally introducing some text fields to help classify the detections.

4.3.1 Result correction and optimizations

As opposed to the Market-1501, real-world situations, specially CCTV systems have wildly variable conditions for each camera. Each camera could have a totally different angle (some are on the ceiling with an aerial view), different contrast and color grading, different illumination conditions, video stream artifacts, et cetera. Due to this, real-world performance of the algorithm will more inconsistent than on an ho-



PNS Service V2

- Home
- Dashboard
- Plugins
- Licenses
- Logout

Webconfig ReID Tool

Currently using 1 image(s) as query, there are an estimated 16 coincidences of the desired person across 4 different times with an average confidence of 80.36%

If you would like to tune the results, select the desired person from the following images and submit:

Select All | ReID

Visualize detection samples

Query image

Recorder: ---------

Group:

Name:

Send to G

© 2022 Copyright: FFVideosistemas.com

((a)) No detections marked for tuning/correction

PNS Service V2

- Home
- Login

Webconfig ReID Tool

Currently using 1 image(s) as query, there are an estimated 1 coincidences of the desired person across 1 different times with an average confidence of nan%

If you would like to tune the results, select the desired person from the following images and submit:

Select All | ReID

Visualize detection samples

Query image

Recorder: ---------

Group:

Name:

Send to G

© 2022 Copyright: FFVideosistemas.com

((b)) Six detections marked for tuning/correction

Figure 4.4: Web app main page

mogeneous dataset. In order to mitigate the problem, result correction techniques and several query optimizations are used.

Correction

On the first run of the algorithm, only images under a threshold of 0.4 are retrieved (60% similarity), then the user is prompted with the result of the iteration. But in case users are not content with the results of the iteration, they are prompted with 10 more samples of identity centroids that are on the verge of confirmation, i.e. images that are not necessarily under the threshold for detection but close enough to believe that the identity could be the same, in this case 50% similarity is used.

Some detections that have already been retrieved by the system but not hand-chosen by the user are also shown, as to validate the current inference results. Users can select the correct retrievals and press the "ReID" button to tune the re-identification results and reduce false positives. From that moment, the representations picked by the user are immutable from the results (they are confirmed true) and are used as part of the queryset for subsequent inference steps. For every re-identification step performed on the same identity the threshold is lowered (similarity needs to be higher every time) to help reduce the result space from using several query images and increase the average confidence level of all retrievals, when a threshold of 0.2 is hit (80% similarity), it remains constant.

For all tuning steps, the queryset consists of: the original query image, the centroids of every confirmed identity, the centroid of all confirmed centroids and the centroid of all images of confirmed identities. The last two help aggregate different representations of the identity, the centroid of centroids aggregates equally all **unique** detections, while the centroid of all images is biased towards the most common representation.

At all moments, the re-identification results can be visualized within the web app as in Figure 4.5(b)

Query optimizations

The gallery set can be quite big depending on the installation, but since we can leverage information of the camera installation we can reduce the gallery set on two main fronts: time and space. Knowing the position of all cameras and the time of detection we can reduce the gallery set to the real of the physically possible. For example, given the query image as taken at 15:00 and a confirmed identity with detections at 14:58 and 14:59, in a camera 40 meters away, we know that is physically impossible (or at least highly unlikely) for the person to be found in cameras 120 meters away from the query at the same minute. For all confirmed identifications, we can keep on pruning the gallery set even further.

Furthermore, **when users select photos for detection tuning, they are also discarding** incorrect identities in the ones not being selected. Discarded identities are blacklisted for the query and not shown again, and for every discarded identities a re-identification step is computed and all identities that are very similar to discarded ones are also discarded (those above a 95% similarity with a discarded identity, to help reduce false positives). During manual testing of the tool, these optimizations proved to help to obtain a good set of results in less time and less tuning steps.

4.4 Integration and deployment

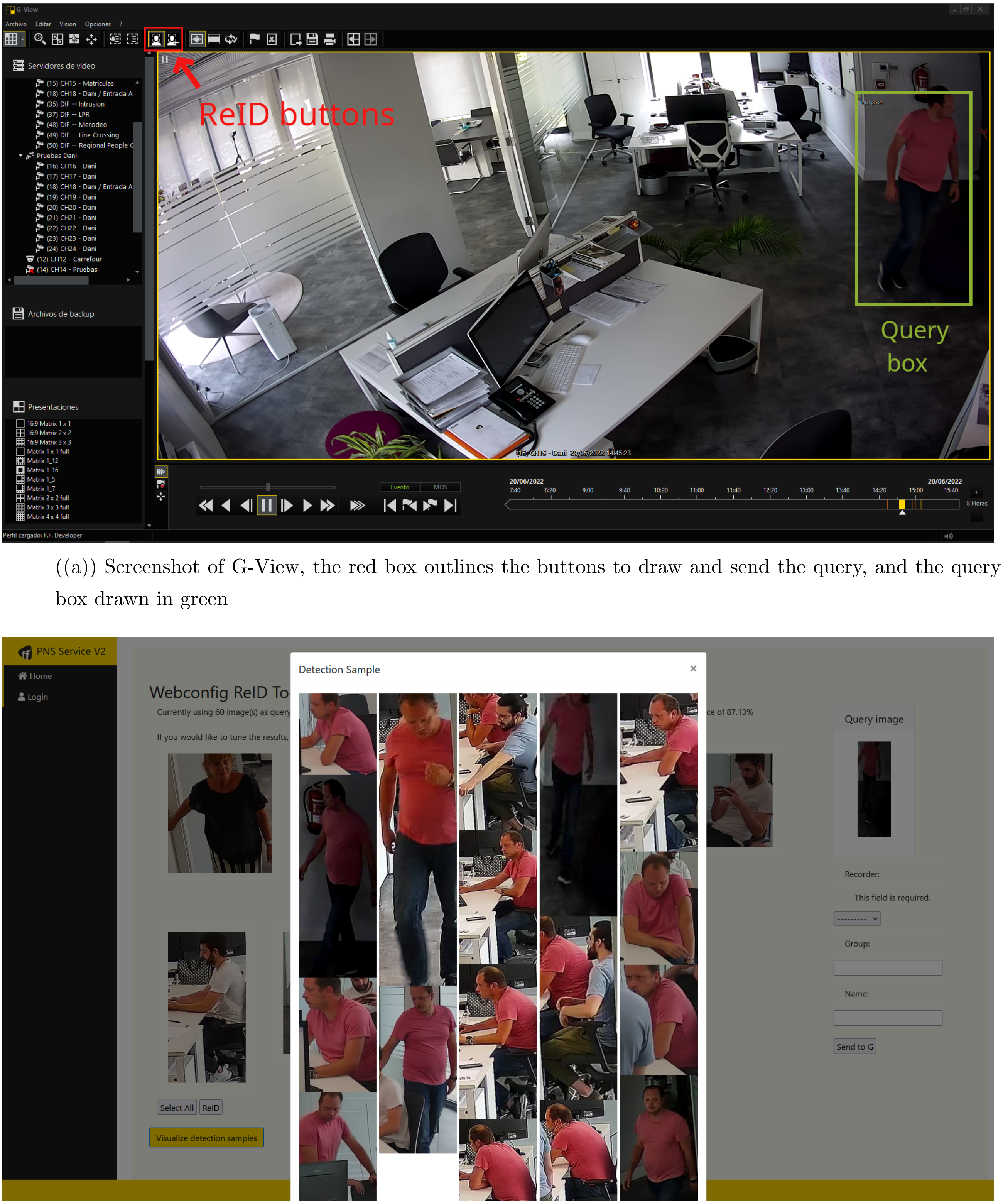
As discussed before, a fully complete integration with the CCTV software was not possible as the need of a supplementary GUI arose, but even the existing integration is still pretty seamless for the user.

For integration purposes, Geutebrück provides a software development kit (SDK) for partners. The SDK is a set of `C#` and `C++` libraries that allow access and interaction with the G-Core, which is the main part of their software and serves the recording, events and alarms databases. During the development of the present project, a simple `C# .NET Framework`[58] project that interfaces with G-Core. For readability purposes, this piece of software will be referred to as the `(`SDK connection service).

4.4.1 Query image

G-View is the software used for video surveillance in the Geutebrück suite. In G-View, users can watch the live feeds of cameras in the system, reproduce recorded video, receive and browse events and alarms and send actions.

G-View provides a standard tools for sending actions to G-Core by selecting the



((a)) Screenshot of G-View, the red box outlines the buttons to draw and send the query, and the query box drawn in green

Currently using 60 image(s) as query

If you would like to tune the results,

ce of 87.13%

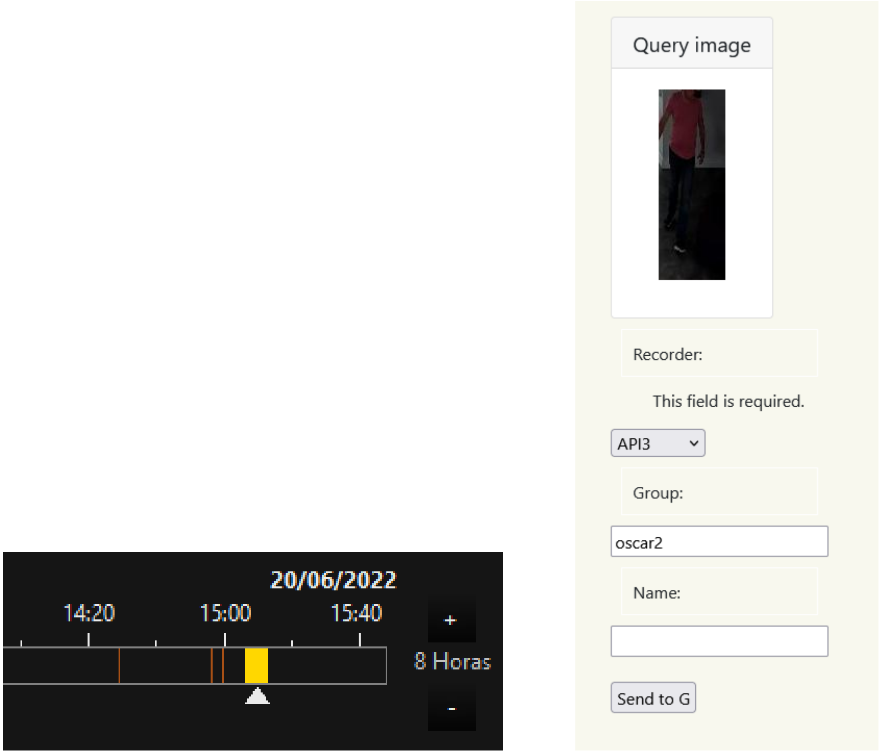
This field is required.

((b)) Web App visualization of ReID results before committing

Figure 4.5: Query and result visualization

tool, and drawing a rectangle in a video feed (both live and recording). Once the rectangle is drawn, the user can send the action containing the time, camera and bounding box for that rectangle to G-Core by simply clicking again.

From the `SDK connection service` (the developed software), a connection to G-Core for the recorder is opened and a callback function is declared so that, when G-Core receives a box selection action from G-View, the image for that box is retrieved from the recordings and sent via HTML POST request to the `{server_ip}/uploadImage` endpoint in the Django sever accompanied by the time and camera information. Figure 4.5(a) shows an annotated screenshot of the process in G-View.



Recorder: API3

This field is required.

Group: oscar2

Name:

((a)) G-View timeline zoom with ReID events

((b)) Send events to G-Core from the web app

Figure 4.6: Timeline and event commit

4.4.2 Event integration

After the re-identification step and once the user is content with the results, they can be sent back to G-Core so that they show up in the CCTV as native events.

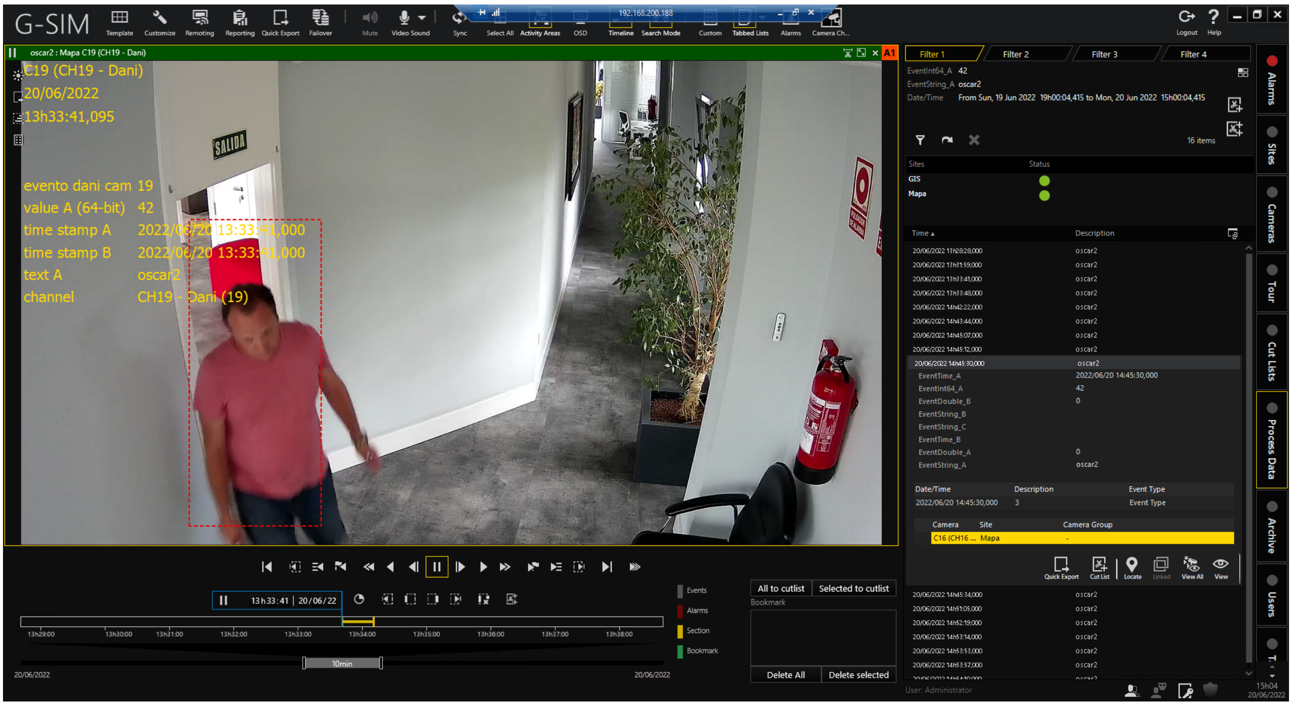
Users can select the recorder to send them to (recorders are already registered in the web configuration app), optionally assign a group and name to the results and send them to G-Core by pressing the button (Figure 4.6).

Once the button is pressed, all images that form the result set are sent to G-Core via Telnet Action Command Interface (TACI), an API like connection. Images are not sent directly, but rather the time, camera id and bounding box location of the detection is sent (along with the optional group and name strings if the user filled them). From that moment on, all users connected to the recorder can observe the re-identification results in G-View's native timeline with bounding boxes (4.6), or actively search for them with filters in another piece of software called G-Sim (Figure 4.7).

4.5 Complete solution

In the end, the final solution is a two-part system which makes use of four different neural networks for its purpose. On one hand, there is the realtime gallery set generation pipeline which involves person detection, tracking and pose estimation working in Python. On the other hand, there is the on demand re-identification tool which involves a state of the art image retrieval model, a `C#` SDK implementation into Geutebrueck's software suite and a complimentary Django web application and API for result tuning and correction. A complete diagram of the solution can be found in Anex B: Complete pipeline diagram.

Additionally, Anex C: Sequence diagram shows a sequence diagram for the common usage of the tool by the security personnel of an installation.



C19 (CH19 - Dani)
20/06/2022
13h33:41,095

evento dani cam 19
value A (64-bit) 42
time stamp A 2022/06/20 13:33:41,000
time stamp B 2022/06/20 13:33:41,000
text A oscar2
channel CH19 - Dani (19)

Filter 1 | Filter 2 | Filter 3 | Filter 4

EventInt64_A 42
EventString_A oscar2
Date/Time From Sun, 19 Jun 2022 19h00:04,415 to Mon, 20 Jun 2022 15h00:04,415

16 items

Sites	Status
GIS	●
Mapa	●

Time ▴	Description
20/06/2022 13h28:28,000	oscar2
20/06/2022 13h31:59,000	oscar2
20/06/2022 13h33:41,000	oscar2
20/06/2022 13h33:48,000	oscar2
20/06/2022 14h42:22,000	oscar2
20/06/2022 14h43:44,000	oscar2
20/06/2022 14h45:07,000	oscar2
20/06/2022 14h45:12,000	oscar2
20/06/2022 14h45:30,000	oscar2
EventTime_A	2022/06/20 14:45:30,000
EventInt64_A	42
EventDouble_B	0
EventString_B	
EventString_C	
EventTime_B	
EventDouble_A	0
EventString_A	oscar2

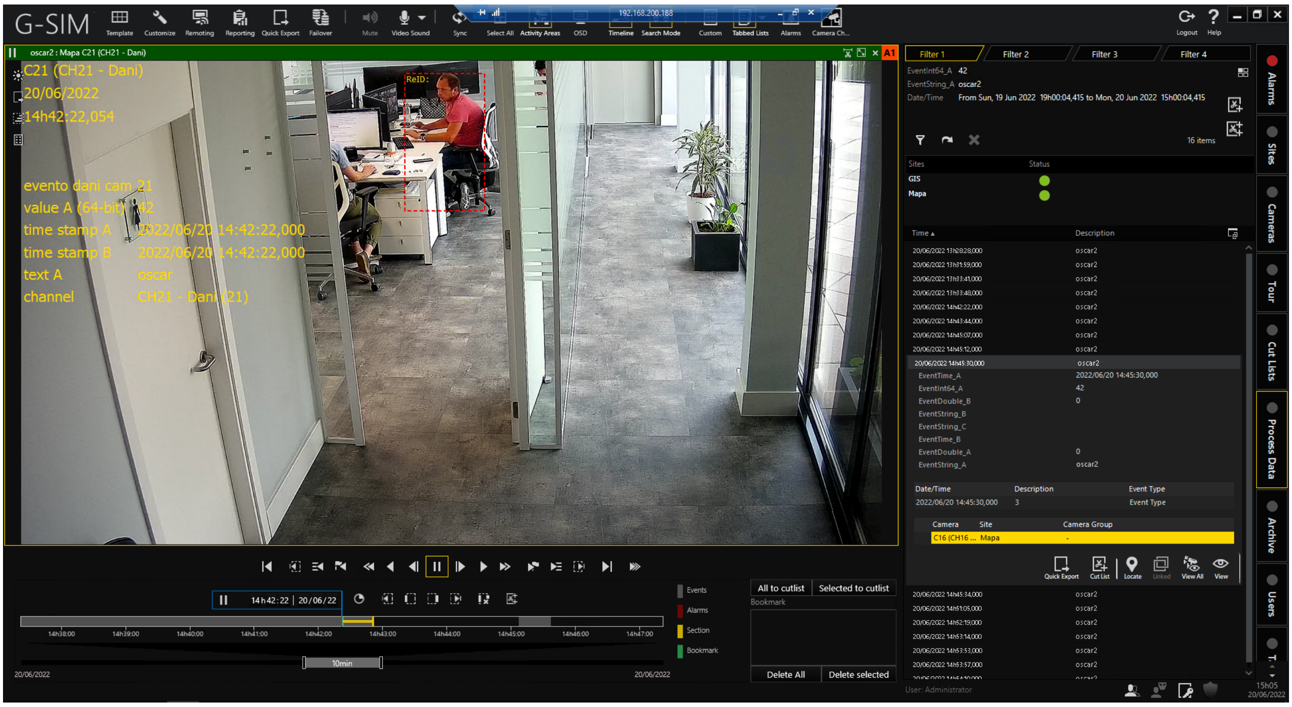
Date/Time	Description	Event Type
2022/06/20 14:45:30,000	3	Event Type

Camera	Site	Camera Group
C16 (CH16 ...	Mapa	-

Time ▴	Description
20/06/2022 14h45:34,000	oscar2
20/06/2022 14h51:05,000	oscar2
20/06/2022 14h52:19,000	oscar2
20/06/2022 14h53:14,000	oscar2
20/06/2022 14h53:53,000	oscar2
20/06/2022 14h53:57,000	oscar2

13h33:41 | 20/06/22
20/06/2022
User: Administrator
15h04
20/06/2022

((a))



C21 (CH21 - Dani)
20/06/2022
14h42:22,054

evento dani cam 21
value A (64-bit) 42
time stamp A 2022/06/20 14:42:22,000
time stamp B 2022/06/20 14:42:22,000
text A oscar
channel CH21 - Dani (21)

Filter 1 | Filter 2 | Filter 3 | Filter 4

EventInt64_A 42
EventString_A oscar2
Date/Time From Sun, 19 Jun 2022 19h00:04,415 to Mon, 20 Jun 2022 15h00:04,415

16 items

14h42:22 | 20/06/22
20/06/2022
User: Administrator
15h05
20/06/2022

((b))

Figure 4.7: Sample detection events in G-Sim, notice the searchbar at the right

Chapter 5

Results, Conclusions and Future Work

5.1 Discussion of results

The person re-identification pipeline that the project set out to implement is complete, functional and integrated with the Geutebrueck CCTV software suite. And re-identification model results have reached state of the art performance in the commonly used Market-1501 dataset, but that does not paint the full picture.

Although almost all on-site empirical experiments and tests with the tool have been sufficiently successful, something to keep in mind is that fine-tuned performance on an open dataset does not necessarily equate to better real world performance. While the model seemed to be quite performant on the tests by being able to retrieve and correctly identify most identities, it definitively did not achieve the same level of consistency as with the Market-1501 dataset in which it scored 99.1% top-5 accuracy. The difference in accuracy can be attributed to the discordance of real world CCTV images with those of readily available datasets. CCTV images tend to be much more heterogeneous in their general image conditions, be it wildly different lightning across cameras, different color gradings, resolutions or weird visualization angles. This problem is further exacerbated by the difficulty of measuring such real world performance, as meaningful metrics should be computed on sufficiently big amounts of diverse, hand-labeled data which prove to be a huge task to obtain.

The effects of these real world problems have been mitigated in the project with the result correction and tuning step, in which authorized security personnel can monitor, verify and improve the re-identification step, reducing the number of missidentifications.

Finally, the project has been successful in accomplish the set of objectives:

- Several video feeds are processed at once in real time (at least up to 9 during testing with the provided hardware).
- Identity consistency is achieved with the DeepSORT tracking algorithm.
- Person detection crop quality is maintained with pose estimation techniques.
- Meaningful identity embeddings are computed and can be used to retrieve instances of a given query.
- An easy to use graphical user interface is available, which also allows the monitoring and verification of the re-identification results before committing to the system.
- The solution is integrated within the Geutebrueck software suite.

5.2 Conclusions

At the end of the project we can conclude that state of the art person re-identification models based around the usage of the new Centroid Triplet Loss function can be improved even further, by taking advantage of convolutional backbones trained using unsupervised, self-supervised contrastive learning techniques. The resultant model is one capable of leveraging the versatility of such pre-trained backbones when learning feature mapping generalization resulting in state of the art results on the Market-1501 person re-identification dataset.

CCTV systems are also at a privileged position to use centroid-based querying methods, having access to large repositories of recorded images across different cameras, allowing the system to capture more diverse shots of an identity which will lead to a more robust representation. In these systems, on-demand person re-identification tools like the one in the project could provide a much less intrusive solution than face recognition technologies.

5.3 Social Impact, Regulations and Concerns

The subject of the work is a very sensitive one and there are very plausible concerns about the abuse of it for ethically controversial use cases. While the intent of the developed tool is that of purely forensic analysis, carried out by authorized and responsible security personnel and in environments where the usage of it is justified, it could be used in an abusive manner for intrusive video surveillance intents. As stated in 4.2.1 Dataset, the Duke MTMC dataset was not used as it had been pulled out of the public domain due to research linking the usage of the dataset with the development of technologies used for perpetrating Human Rights violations against Uighur muslims in China[53]. Such an event amidst the controversial ethics debate surrounding similar technologies sets a horrifying example of how they can be used and the crucial importance of regulating the responsible usage of them.

Regulations

As an introductory disclaimer, I will commence by stating I do not have any formal studies in the field of Law and the following legal analysis has been performed on the best of my knowledge. On a positive note, I will also state that the present work will be assessed and reviewed by a proper legal expert before any possible applications to comply with the regulations.

Within the framework of the European Union, similar technologies such as face recognition are rightfully regulated under the General Data Protection Regulation[59] (GDPR) and the "A.I. Act"[60] which classify these technologies into 3 tiers ('no risk', 'low risk' and 'high risk') regarding the processing and storage of personal data. The images of people captured by CCTV systems and the biometric data that the representation embeddings contain already set the tool on the tier of 'low risk' applications at the bare minimum, and the usage of them ought to be justified and handled with responsibility and accountability.

The usage of facial recognition systems is almost unanimously classified as 'high risk' on the premises of its intrusiveness and interference with the rights and freedoms of the persons concerned. Face recognition systems may lead to constant surveillance and indirectly dissuade the exercise of the freedom of assembly or other fundamental rights. Furthermore, the immediacy, persistence and inescapability of face recognition (you cannot change your facial structure) increase the concerns of usage of the technology, which can also lead to wrong or biased results and entail discriminatory effects. As such, face recognition usage on public spaces is generally banned. In practice, the 'A.I. Act' aims to **prohibit the use of real-time face recognition** (with an emphasis on real-time) for the purpose of law enforcement. Police forces should not be able to deploy facial recognition systems to target persons participating in a public protest or abuse them to locate persons who have only committed minor offences.

There are, however, three main exceptions for the usage of facial recognition systems in which the public interests outweigh the risks for fundamental rights. Those are: the **targeted search** for potential victims of crime (including missing children), the **prevention** of substantial and imminent threats to physical safety of people (such as a **terrorist attack**) and the **identification and localisation** of a perpetrator or individual suspected of a **serious criminal offence** (as dictated by the European Arrest Warrant Framework[61]). The draft, however, leaves the ultimate regulation to the Member States to decide whether they want to implement the exceptions or not as national security matters generally remain an exclusive competence of the Member States.

On the subject of the developed pipeline, the person re-identification method used could be deemed less intrusive in that the scope of the representation is nowhere near that of facial recognition. The learned representation within the system does not depend on the immutable facial structure of the persons, but rather in the general resemblance of images, and as such **can be bypassed by a simple change of clothes**. This crucial distinction lowers the risk of its usage in unrightful, permanent surveillance as the detections of one day cannot be matched by the system with those of another day, or even within the same day (given a change of clothes or any other major aesthetic change). The saved detections and gallery sets are also anonymised, **no other personal data is saved** tied to it that could help link the image/embedding with a real world identity, such as a name or identification document number. Embeddings are not saved for more that 24 hours (as they become irrelevant anyways) and are not compared against any permanent database to match people with external sources or blacklists such as a criminal record database. The re-identification step is also performed on demand (not in real-time) under a justifiable cause, which could very well fall into one of the 3 aforementioned exceptions. It is also not fully automatised, an authorized user is also able to monitor, verify and correct the re-identification results as to weed out incorrect results from the system and help prevail the fundamental rights of innocent persons.

Concerns

As a final note on the developed model, and apart from the mentioned concerns around its unethical usage in the violation of Human Rights there exists a concern for a racial bias within the model.

The image retrieval model, although pre-trained on the more diverse LUPerson dataset which contains images of people of all ethnicities, is ultimately fine-tuned on the Market-1501 dataset which primarily contains images of people of Asian (more specifically, Han Chinese) descent. For this, the model could have an ingrained racial bias which could affect the re-identification accuracy of different ethnicities. Unfortunately, the problem is very hard to solve at the moment due to a general lack of availability of labelled data of other type, and more work should be made in this front in the future.

5.4 Future Work

There is a lot to be done to continue and improve the existing project. The general performance of the solution can be vastly improved by migrating the bulk of the code from Python to `C++` and using a general model compilation method such as the ONNX format[62]. The current state of the solution also requires several decoupled parts, so there is a plan to package the complete solution under a single executable file or Docker container to ease the installation and start up of it. For large scale installations, embedding retrieval and comparison could also be improved through the use of a scalable vector database such as Pinecone[63] or Milvus[64].

On the user experience front, more filtering options such a time constraint or the usage of only some cameras could be built in. The functionality of the web app could also be ported to a .NET project when the code is migrated or integrated more deeply into Geutebrück's software suite if they ever allow for that. Similarly, the tool could also be integrated into other CCTV systems in the future.

As the field advances and new research is published, the models used could be exchanged for new ones and/or retrained with new available data. On anoter note, the existing pipeline could also be tweaked to change the focus of the re-identification subject from persons to vehicles or other objects such as animals or parcels in logistic hubs for new projects.

Bibliography

[1] Chrislb. (2005) Diagram of an artificial neuron. [Online]. Available: https://commons.wikimedia.org/wiki/File:ArtificialNeuronModel_english.png

[2] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," 2016. [Online]. Available: https://arxiv.org/abs/1612.08242

[3] X. Chen, H. Fan, R. Girshick, and K. He, "Improved baselines with momentum contrastive learning," 2020. [Online]. Available: https://arxiv.org/abs/2003.04297

[4] M. Wieczorek, B. Rychalska, and J. Dabrowski, "On the unreasonable effectiveness of centroids in image retrieval," 2021. [Online]. Available: https://arxiv.org/abs/2104.13643

[5] Y. LeCun, P. Haffner, L. Bottou, and Y. Bengio, "Object recognition with gradient-based learning," in *Shape, Contour and Grouping in Computer Vision*, ser. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), J. Mundy, R. Cipolla, D. Forsyth, and V. di Gesu, Eds. Springer Verlag, 1999, pp. 319–345, international Workshop on Shape, Contour and Grouping in Computer Vision ; Conference date: 26-05-1998 Through 29-05-1998. [Online]. Available: http://yann.lecun.com/exdb/publis/pdf/lecun-99.pdf

[6] D. Fu, D. Chen, J. Bao, H. Yang, L. Yuan, L. Zhang, H. Li, and D. Chen, "Unsupervised pre-training for person re-identification," 2021.

[7] A. Kratsios, "Universal approximation theorems," 10 2019.

[8] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014. [Online]. Available: https://arxiv.org/abs/1409.1556

[9] 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.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015. [Online]. Available: https://arxiv.org/abs/1512.03385

[11] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," 2016. [Online]. Available: https://arxiv.org/abs/1611.05431

[12] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," 2016. [Online]. Available: https://arxiv.org/abs/1608.06993

[13] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," 2019. [Online]. Available: https://arxiv.org/abs/1905.11946

[14] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei, "Language models are few-shot learners," 2020. [Online]. Available: https://arxiv.org/abs/2005.14165

[15] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. Garcia, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan,

S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Diaz, O. Firat, M. Catasta, J. Wei, K. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel, "Palm: Scaling language modeling with pathways," 2022. [Online]. Available: https://arxiv.org/abs/2204.02311

[16] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Microsoft coco: Common objects in context," 2014. [Online]. Available: https://arxiv.org/abs/1405.0312

[17] E. Ristani, F. Solera, R. S. Zou, R. Cucchiara, and C. Tomasi, "Performance measures and a data set for multi-target, multi-camera tracking," in *ECCV Workshops*, 2016.

[18] L. Zheng, L. Shen, L. Tian, S. Wang, J. Wang, and Q. Tian, "Scalable person re-identification: A benchmark," in *Computer Vision, IEEE International Conference on*, 2015.

[19] W. Li, R. Zhao, T. Xiao, and X. Wang, "Deepreid: Deep filter pairing neural network for person re-identification," in *CVPR*, 2014.

[20] L. Wei, S. Zhang, W. Gao, and Q. Tian, "Person transfer gan to bridge domain gap for person re-identification," 2017. [Online]. Available: https://arxiv.org/abs/1711.08565

[21] G. J. et. al., "ultralytics/yolov5: v6.0 - YOLOv5n 'Nano' models, Roboflow integration, TensorFlow export, OpenCV DNN support," Oct. 2021. [Online]. Available: https://doi.org/10.5281/zenodo.5563715

[22] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," 2013. [Online]. Available: https://arxiv.org/abs/1311.2524

[23] R. Girshick, "Fast r-cnn," 2015. [Online]. Available: https://arxiv.org/abs/1504.08083

[24] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2017. [Online]. Available: https://arxiv.org/abs/1708.02002

[25] J. Hosang, R. Benenson, and B. Schiele, "Learning non-maximum suppression," 2017. [Online]. Available: https://arxiv.org/abs/1705.02950

[26] Human Pose Estimation Model HRNet Breaks Three COCO Records; CVPR Accepts Paper. [Online]. Available: https://medium.com/syncedreview/human-pose-estimation-model-hrnet-breaks-three-coco-records-cvpr-accepts\-paper-74e57fabdeb6

[27] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, and B. Xiao, "Deep high-resolution representation learning for visual recognition," 2019. [Online]. Available: https://arxiv.org/abs/1908.07919

[28] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan, "Supervised contrastive learning," 2020. [Online]. Available: https://arxiv.org/abs/2004.11362

[29] Y. Sun, C. Cheng, Y. Zhang, C. Zhang, L. Zheng, Z. Wang, and Y. Wei, "Circle loss: A unified perspective of pair similarity optimization," 2020. [Online]. Available: https://arxiv.org/abs/2002.10857

[30] X. Dong and J. Shen, "Triplet loss in siamese network for object tracking," pp. 472–488, 2018.

[31] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2018. [Online]. Available: https://arxiv.org/abs/1810.04805

[32] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," 2019. [Online]. Available: https://arxiv.org/abs/1908.10084

[33] R. Hadsell, S. Chopra, and Y. LeCun, "Dimensionality reduction by learning an invariant mapping," in *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, vol. 2, 2006, pp. 1735–1742.

[34] T. Chen, S. Kornblith, M. Norouzi, and G. Hinton, "A simple framework for contrastive learning of visual representations," 2020. [Online]. Available: https://arxiv.org/abs/2002.05709

[35] T. Chen, S. Kornblith, K. Swersky, M. Norouzi, and G. Hinton, "Big self-supervised models are strong semi-supervised learners," 2020. [Online]. Available: https://arxiv.org/abs/2006.10029

[36] K. He, H. Fan, Y. Wu, S. Xie, and R. Girshick, "Momentum contrast for unsupervised visual representation learning," 2019. [Online]. Available: https://arxiv.org/abs/1911.05722

[37] X. Chen, S. Xie, and K. He, "An empirical study of training self-supervised vision transformers," 2021. [Online]. Available: https://arxiv.org/abs/2104.02057

[38] W. Ågren, "The nt-xent loss upper bound," 2022. [Online]. Available: https://arxiv.org/abs/2205.03169

[39] Z. Zhang, C. Lan, W. Zeng, Z. Chen, and S.-F. Chang, "Beyond triplet loss: Meta prototypical n-tuple loss for person re-identification," 2020. [Online]. Available: https://arxiv.org/abs/2006.04991

[40] T.-T. Do, T. Tran, I. Reid, V. Kumar, T. Hoang, and G. Carneiro, "A theoretically sound upper bound on the triplet loss for improving the efficiency of deep distance metric learning," 2019. [Online]. Available: https://arxiv.org/abs/1904.08720

[41] H. Xuan, A. Stylianou, X. Liu, and R. Pless, "Hard negative examples are hard, but useful," 2020. [Online]. Available: https://arxiv.org/abs/2007.12749

[42] Y. Yuan, W. Chen, Y. Yang, and Z. Wang, "In defense of the triplet loss again: Learning robust person re-identification with fast approximated triplet loss and label distillation," 2019. [Online]. Available: https://arxiv.org/abs/1912.07863

[43] H. Schulzrinne, A. Rao, and R. Lanphier, "Rfc2326: Real time streaming protocol (rtsp)," USA, 1998.

[44] G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.

[45] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito,

M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. [Online]. Available: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

[46] P. Ultralytics. Official yolov5 pytorch implementation. [Online]. Available: https://pytorch.org/hub/ultralytics_yolov5/

[47] N. Wojke, A. Bewley, and D. Paulus, "Simple online and realtime tracking with a deep association metric," 2017. [Online]. Available: https://arxiv.org/abs/1703.07402

[48] A. Bewley, Z. Ge, L. Ott, F. Ramos, and B. Upcroft, "Simple online and realtime tracking," in *2016 IEEE International Conference on Image Processing (ICIP)*. IEEE, sep 2016. [Online]. Available: https://doi.org/10.1109%2Ficip.2016.7533003

[49] M. Contributors, "MMCV: OpenMMLab computer vision foundation," https://github.com/open-mmlab/mmcv, 2018.

[50] C. Yu, B. Xiao, C. Gao, L. Yuan, L. Zhang, N. Sang, and J. Wang, "Lite-hrnet: A lightweight high-resolution network," 2021. [Online]. Available: https://arxiv.org/abs/2104.06403

[51] X. Zhang, X. Zhou, M. Lin, and J. Sun, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," 2017. [Online]. Available: https://arxiv.org/abs/1707.01083

[52] mikwieczorek, "On the unreasonable effectiveness of centroids in image retrieval, official implementation," https://github.com/mikwieczorek/centroids-reid, 2022.

[53] Exposing.ai, "Duke mtmc concerns — exposing.ai," https://exposing.ai/duke_mtmc/, 2020.

[54] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014.

[55] The PyTorch Lightning team, “Pytorch lightning.” [Online]. Available: https://www.pytorchlightning.ai

[56] Geutebrück SDK. [Online]. Available: https://www.geutebrueck.com/solutions-technology/technology/interfaces/software-development-kits-sdk.html

[57] Django Software Foundation, “Django.” [Online]. Available: https://djangoproject.com

[58] Microsoft, “.net framework.” [Online]. Available: https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet

[59] European Commission, “Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance),” 2016. [Online]. Available: https://eur-lex.europa.eu/eli/reg/2016/679/oj

[60] European Council and Parliament, “Regulation (EU) 2021/206 of the European Parliament and of the Council laying down the harmonised rules on Artificial Intelligence (Artificial Intelligence Act) and amending certain Union Legislative Acts,” 2021. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX:52021PC0206

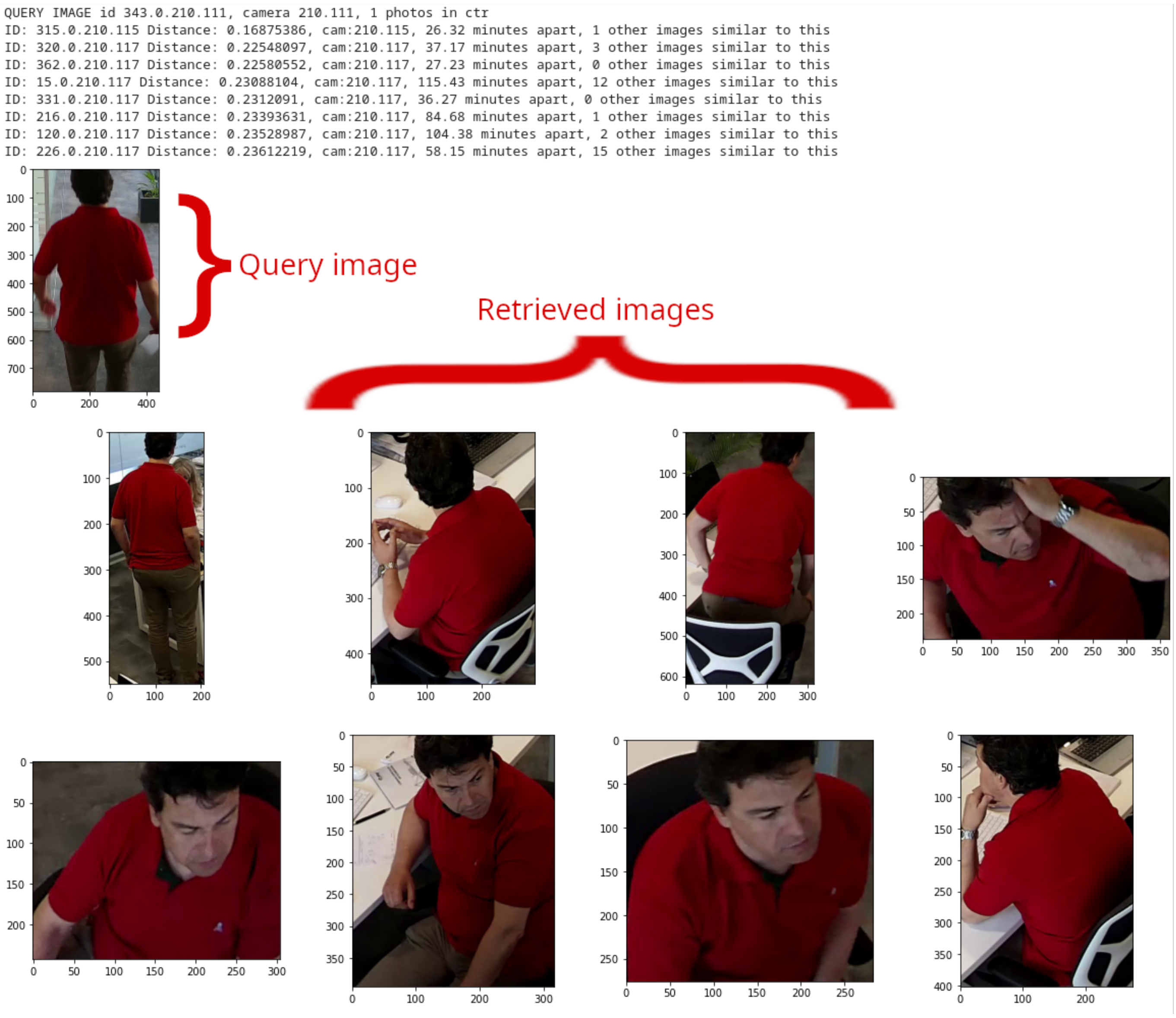
[61] European Council, “Council Framework Decision of 13 June 2002 on the European arrest warrant and the surrender procedures between Member States - Statements made by certain Member States on the adoption of the Framework Decision,” 2002. [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A32002F0584

[62] T. L. Foundation, “Open neural network exchange format.” [Online]. Available: https://onnx.ai/

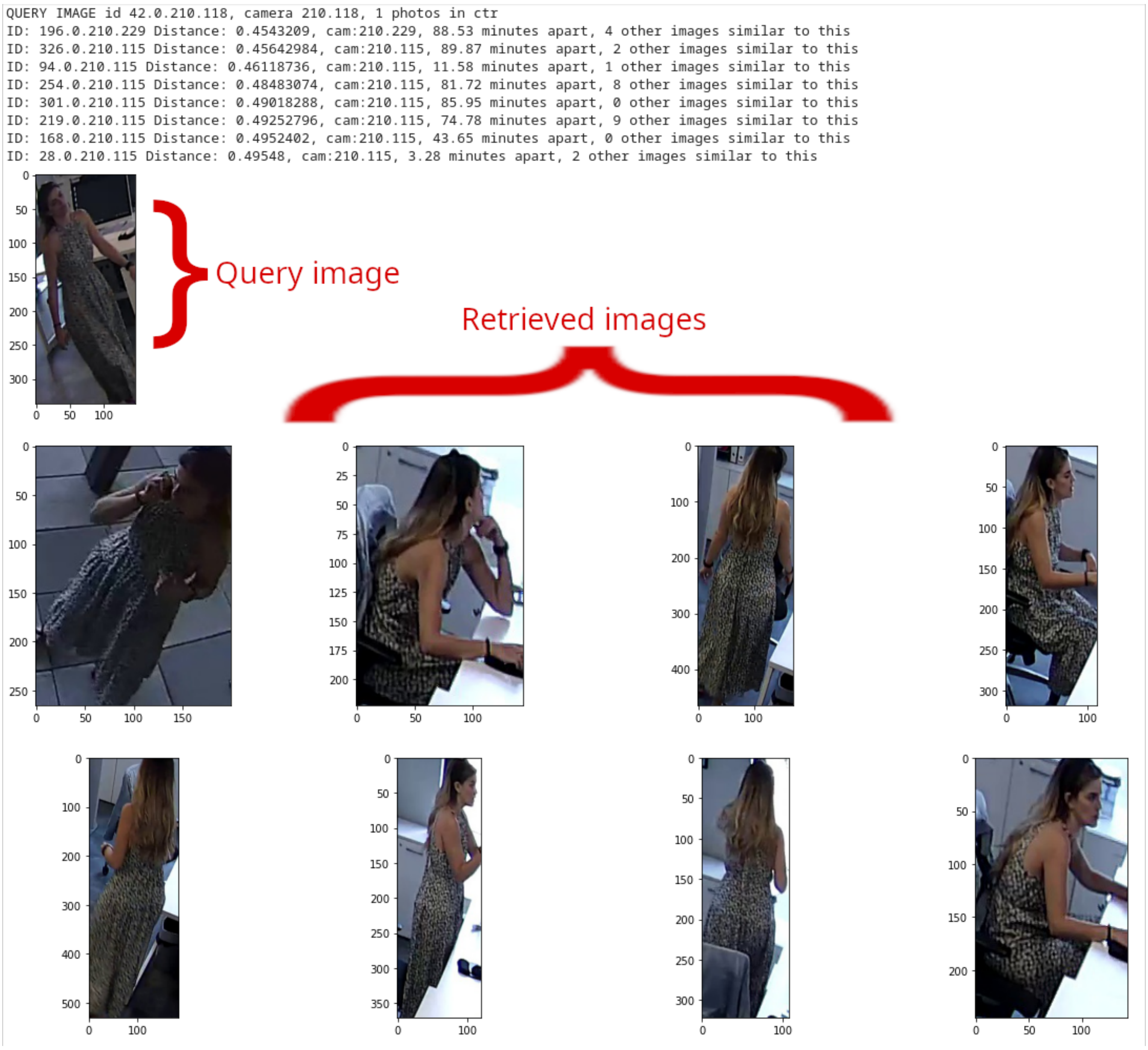
[63] P. S. Inc., “Pinecone vector database.” [Online]. Available: https://www.pinecone.io/

[64] T. L. Foundation, "Milvus vector database." [Online]. Available: https://milvus.io/

Anex A: Person re-identification experiment samples on Jupyter Lab

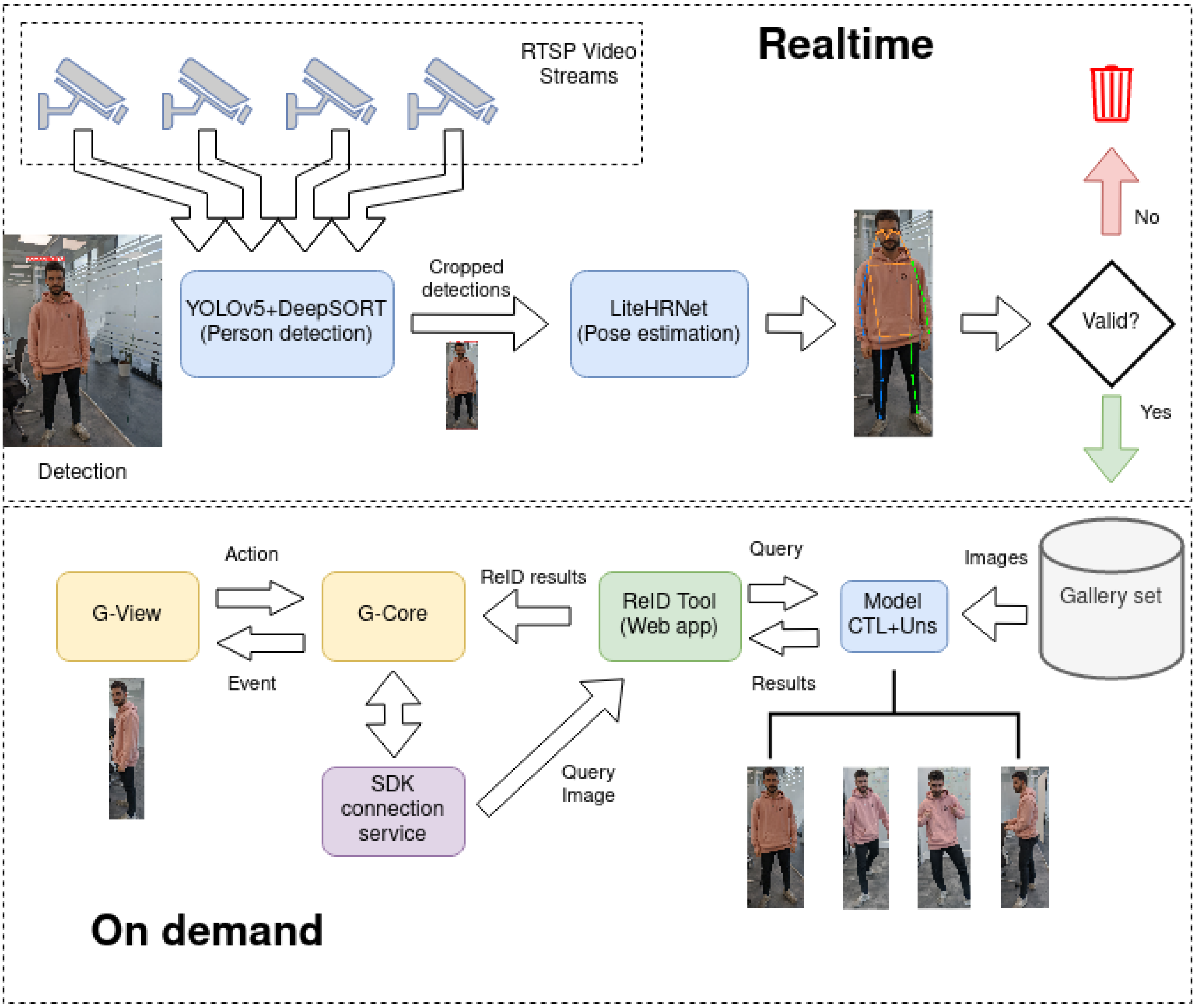


```
QUERY IMAGE id 343.0.210.111, camera 210.111, 1 photos in ctr
ID: 315.0.210.115 Distance: 0.16875386, cam:210.115, 26.32 minutes apart, 1 other images similar to this
ID: 320.0.210.117 Distance: 0.22548097, cam:210.117, 37.17 minutes apart, 3 other images similar to this
ID: 362.0.210.117 Distance: 0.22580552, cam:210.117, 27.23 minutes apart, 0 other images similar to this
ID: 15.0.210.117 Distance: 0.23088104, cam:210.117, 115.43 minutes apart, 12 other images similar to this
ID: 331.0.210.117 Distance: 0.2312091, cam:210.117, 36.27 minutes apart, 0 other images similar to this
ID: 216.0.210.117 Distance: 0.23393631, cam:210.117, 84.68 minutes apart, 1 other images similar to this
ID: 120.0.210.117 Distance: 0.23528987, cam:210.117, 104.38 minutes apart, 2 other images similar to this
ID: 226.0.210.117 Distance: 0.23612219, cam:210.117, 58.15 minutes apart, 15 other images similar to this
```

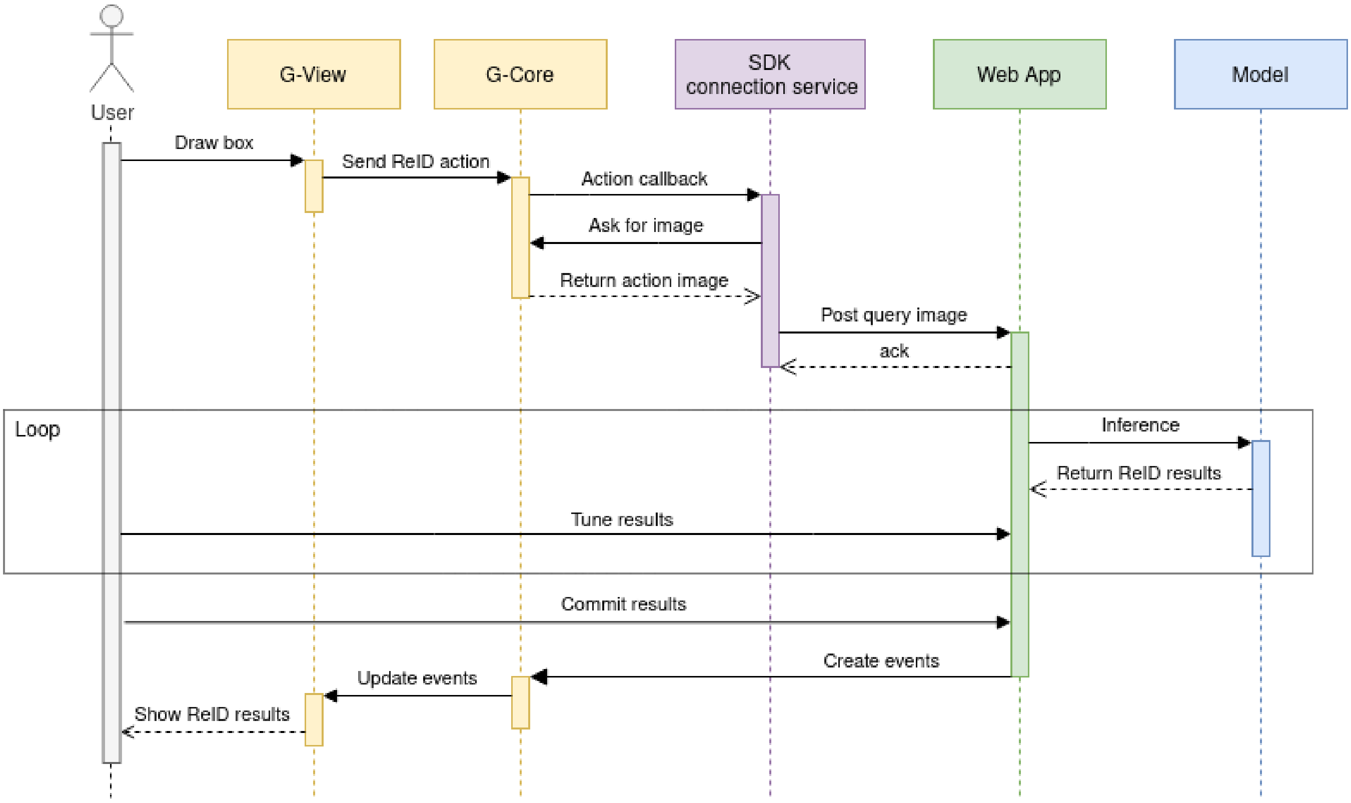


```
QUERY IMAGE id 42.0.210.118, camera 210.118, 1 photos in ctr
ID: 196.0.210.229 Distance: 0.4543209, cam:210.229, 88.53 minutes apart, 4 other images similar to this
ID: 326.0.210.115 Distance: 0.45642984, cam:210.115, 89.87 minutes apart, 2 other images similar to this
ID: 94.0.210.115 Distance: 0.46118736, cam:210.115, 11.58 minutes apart, 1 other images similar to this
ID: 254.0.210.115 Distance: 0.48483074, cam:210.115, 81.72 minutes apart, 8 other images similar to this
ID: 301.0.210.115 Distance: 0.49018288, cam:210.115, 85.95 minutes apart, 0 other images similar to this
ID: 219.0.210.115 Distance: 0.49252796, cam:210.115, 74.78 minutes apart, 9 other images similar to this
ID: 168.0.210.115 Distance: 0.4952402, cam:210.115, 43.65 minutes apart, 0 other images similar to this
ID: 28.0.210.115 Distance: 0.49548, cam:210.115, 3.28 minutes apart, 2 other images similar to this
```

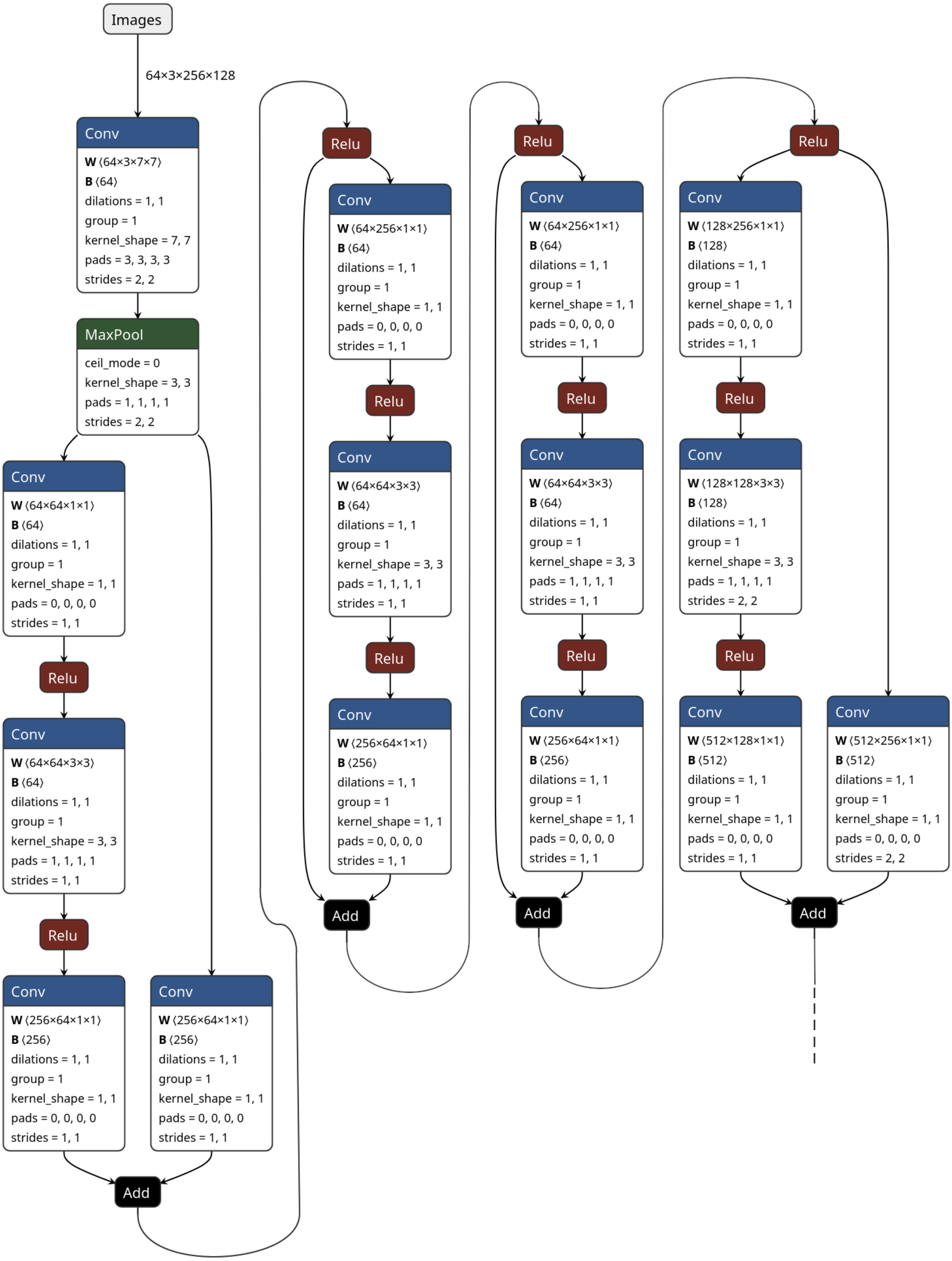
Anex B: Complete pipeline diagram

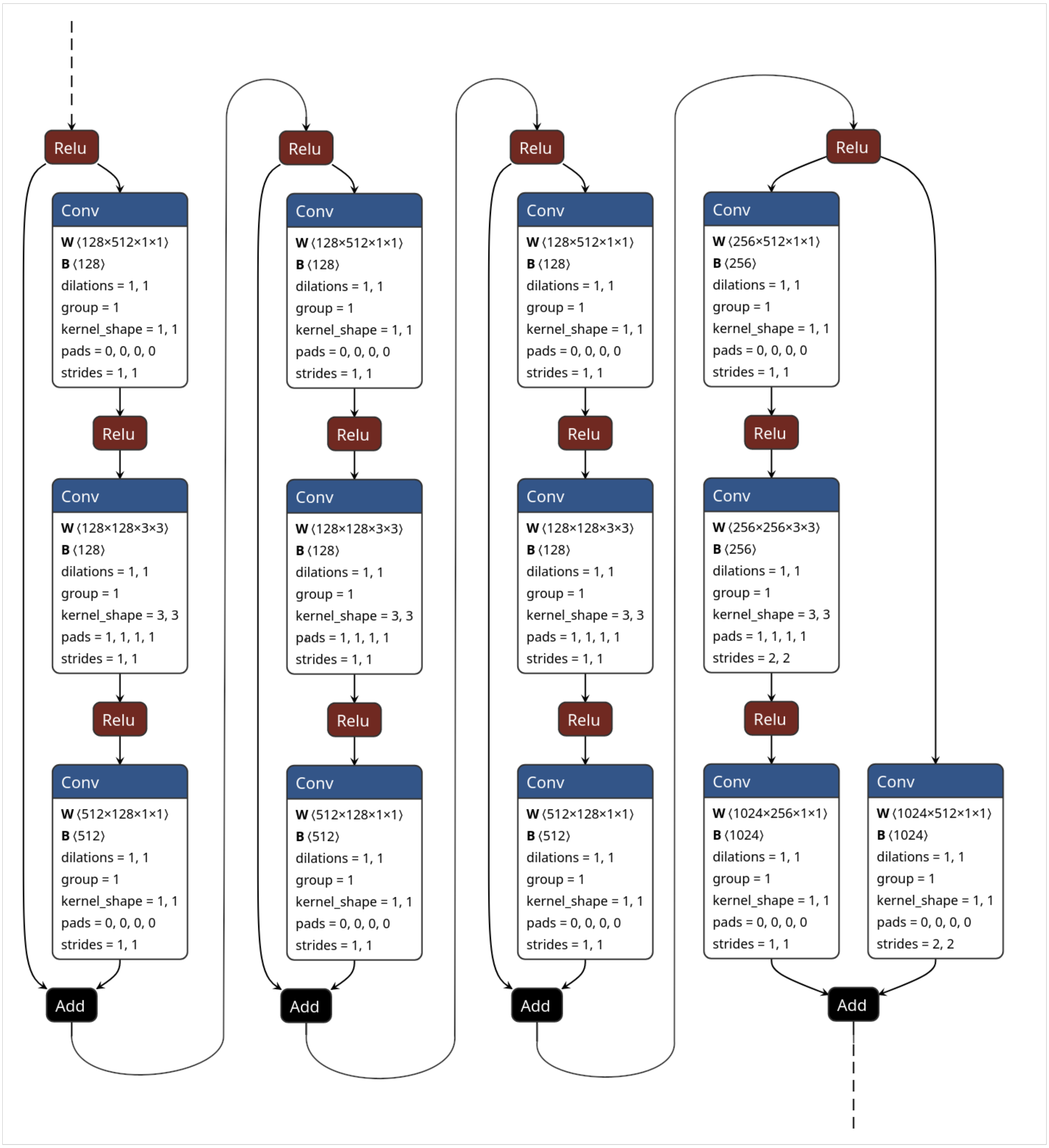


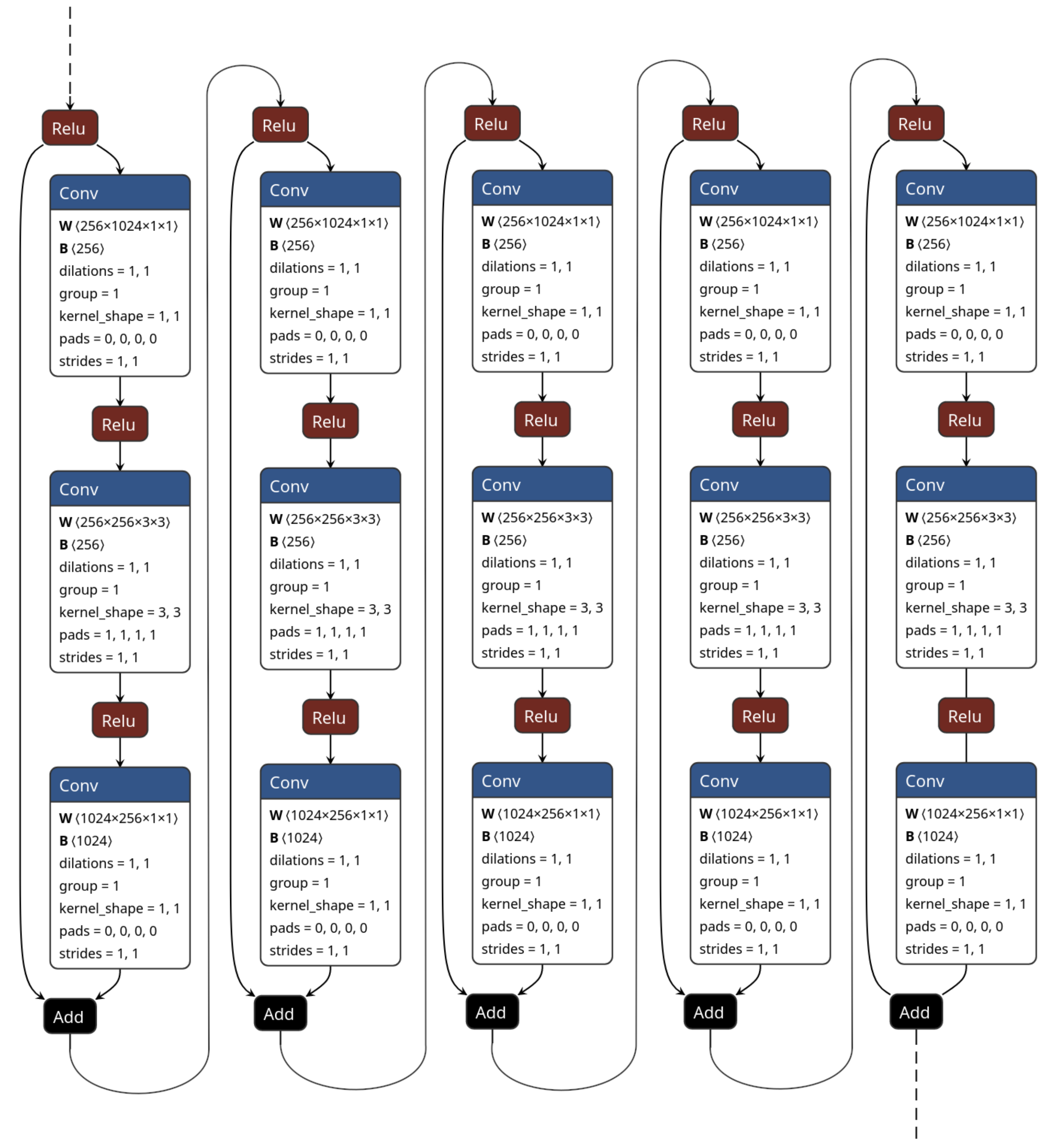
Anex C: Sequence diagram

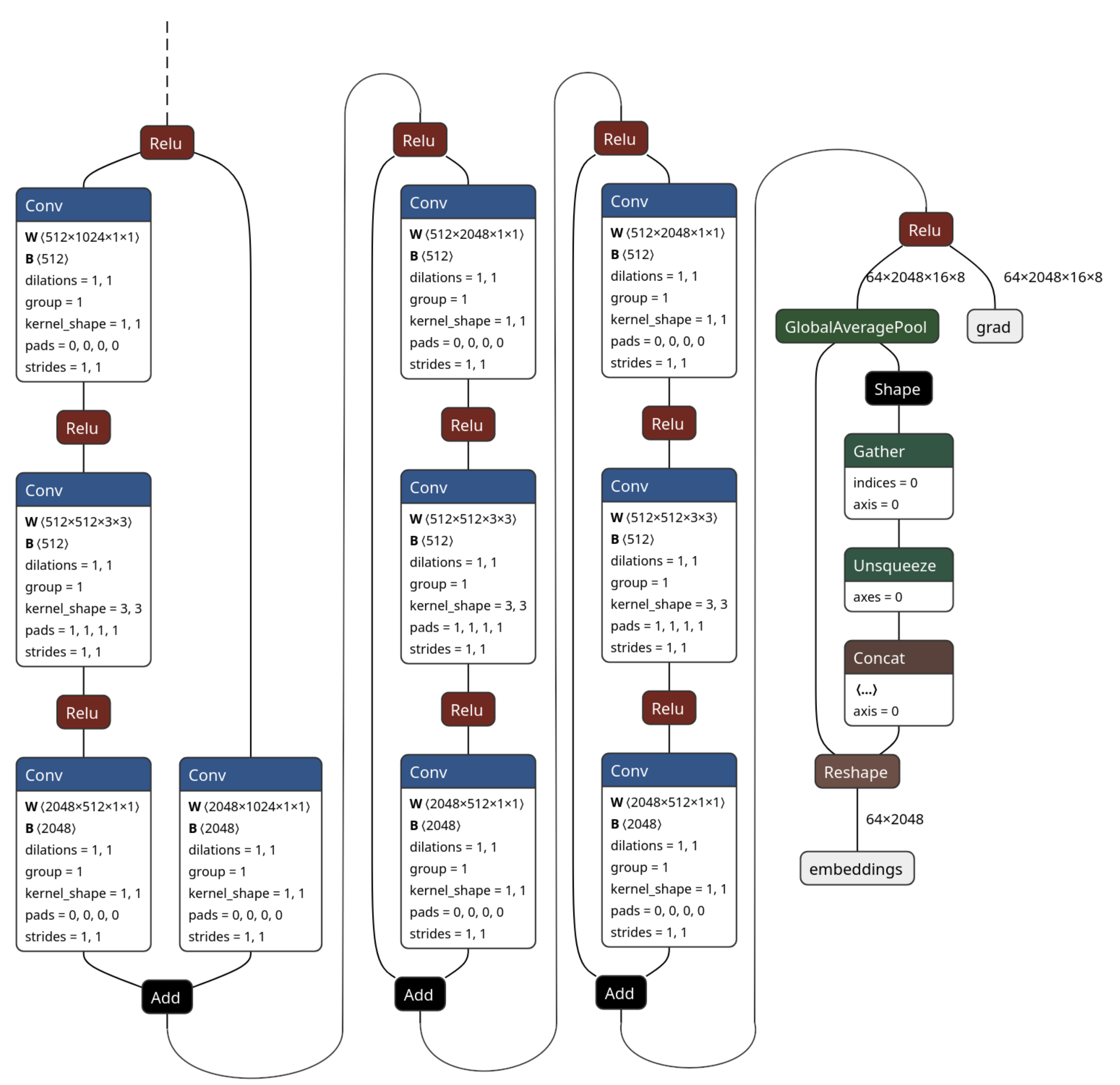


Anex D: ResNet50 backbone diagram


Images
64×3×256×128
Conv
W ⟨64×3×7×7⟩
B ⟨64⟩
dilations = 1, 1
group = 1
kernel_shape = 7, 7
pads = 3, 3, 3, 3
strides = 2, 2
MaxPool
ceil_mode = 0
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 2, 2
Conv
W ⟨64×64×1×1⟩
B ⟨64⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨64×64×3×3⟩
B ⟨64⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨256×64×1×1⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Conv
W ⟨256×64×1×1⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨64×256×1×1⟩
B ⟨64⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨64×64×3×3⟩
B ⟨64⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨256×64×1×1⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨64×256×1×1⟩
B ⟨64⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨64×64×3×3⟩
B ⟨64⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨256×64×1×1⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨128×256×1×1⟩
B ⟨128⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨128×128×3×3⟩
B ⟨128⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 2, 2
Relu
Conv
W ⟨512×128×1×1⟩
B ⟨512⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Conv
W ⟨512×256×1×1⟩
B ⟨512⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 2, 2
Add


Relu
Conv
W ⟨128×512×1×1⟩
B ⟨128⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨128×128×3×3⟩
B ⟨128⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨512×128×1×1⟩
B ⟨512⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨128×512×1×1⟩
B ⟨128⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨128×128×3×3⟩
B ⟨128⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨512×128×1×1⟩
B ⟨512⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨128×512×1×1⟩
B ⟨128⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨128×128×3×3⟩
B ⟨128⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨512×128×1×1⟩
B ⟨512⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨256×512×1×1⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨256×256×3×3⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 2, 2
Relu
Conv
W ⟨1024×256×1×1⟩
B ⟨1024⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Conv
W ⟨1024×512×1×1⟩
B ⟨1024⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 2, 2
Add


Relu
Conv
W ⟨256×1024×1×1⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨256×256×3×3⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨1024×256×1×1⟩
B ⟨1024⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨256×1024×1×1⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨256×256×3×3⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨1024×256×1×1⟩
B ⟨1024⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨256×1024×1×1⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨256×256×3×3⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨1024×256×1×1⟩
B ⟨1024⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨256×1024×1×1⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨256×256×3×3⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨1024×256×1×1⟩
B ⟨1024⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨256×1024×1×1⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨256×256×3×3⟩
B ⟨256⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨1024×256×1×1⟩
B ⟨1024⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add


Relu
Conv
W ⟨512×1024×1×1⟩
B ⟨512⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨512×512×3×3⟩
B ⟨512⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨2048×512×1×1⟩
B ⟨2048⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Conv
W ⟨2048×1024×1×1⟩
B ⟨2048⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨512×2048×1×1⟩
B ⟨512⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨512×512×3×3⟩
B ⟨512⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨2048×512×1×1⟩
B ⟨2048⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
Conv
W ⟨512×2048×1×1⟩
B ⟨512⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Relu
Conv
W ⟨512×512×3×3⟩
B ⟨512⟩
dilations = 1, 1
group = 1
kernel_shape = 3, 3
pads = 1, 1, 1, 1
strides = 1, 1
Relu
Conv
W ⟨2048×512×1×1⟩
B ⟨2048⟩
dilations = 1, 1
group = 1
kernel_shape = 1, 1
pads = 0, 0, 0, 0
strides = 1, 1
Add
Relu
64×2048×16×8
64×2048×16×8
GlobalAveragePool
grad
Shape
Gather
indices = 0
axis = 0
Unsqueeze
axes = 0
Concat
⟨...⟩
axis = 0
Reshape
64×2048
embeddings