UNIVERSITAT DE BARCELONA

Treball final de grau

GRAU EN ENGINYERIA INFORMÀTICA

Facultat de Matemàtiques i Informàtica
Universitat de Barcelona

Segmentation-guided Privacy Preservation in Visual Surveillance Monitoring

---

Autor: Daniel Geryous Fares

Co-Director:	**Dr. Sergio Escalera Guerrero** Departament de Matemàtiques i Informàtica. Universitat de Barcelona
Co-Director:	**Zenjie Li** Milestone systems.
Co-Director:	**Kamal Nasrollahi** Milestone systems.
Realitzat a:	**Departament de Matemàtiques i Informàtica**
Barcelona,	**June 13, 2022**

Contents

Abstract – Video surveillance has become a necessity to ensure safety and security. Today, with the advancement of technology, video surveillance has become more accessible and widely available. Furthermore, it can be useful in an enormous amount of applications and situations. For instance, it can be useful in ensuring public safety by preventing vandalism, robbery, and shoplifting. The same applies to more intimate situations, like home monitoring to detect unusual behavior of residents or in similar situations like hospitals and assisted living facilities. Thus, cameras are installed in public places like malls, metro stations, and on-roads for traffic control, as well as in sensitive settings like hospitals, embassies, and private homes. Video surveillance has always been associated with the loss of privacy. Therefore, we developed a real-time visualization of privacy-protected video surveillance data by applying a segmentation mask to protect privacy while still being able to identify existing risk behaviors. This replaces existing privacy safeguards such as blanking, masking, pixelation, blurring, and scrambling. As we want to protect human personal data that are visual such as appearance, physical information, clothing, skin, eye and hair color, and facial gestures. Our main aim of this work is to analyze and compare the most successful deep-learning-based state-of-the-art approaches for semantic segmentation. In this study, we perform an efficiency-accuracy comparison to determine which segmentation methods yield accurate segmentation results while performing at the speed and execution required for real-life application scenarios. Furthermore, we also provide a modified dataset made from a combination of three existing datasets, **COCO_stuff164K**, **PASCAL VOC 2012**, and **ADE20K**, to make our comparison fair and generate privacy-protecting human segmentation masks.

Keywords – Real-time Segmentation, Semantic Segmentation, privacy, neural network.

1 Introduction

Video surveillance has become increasingly necessary to ensure safety and security. Video surveillance has a vast volume of applications for monitoring and environmental analysis. Real-time video semantic segmentation is used extensively in security and surveillance companies, by providing a prediction and identification of intruders and study of their behavior, relying in a very significant way on the accuracy of the segmentation process. Real-time human recognition applications require segmenting humans for posterior behavior analysis. However, detecting humans is difficult in uncontrolled surveillance scenarios, and segmenting pixels becomes harder. But, today video surveillance cameras are combined with advanced and intelligent systems that are required to provide speed and high accuracy. Recent approaches propose the application of deep convolutional neural networks to achieve these requirements. Deep-learning semantic segmentation models work on fully understanding the input image content and extracting important features. High-quality spatial information and a sizable Receptive Field are required to classify each pixel in real-time speed.

Image segmentation task has become easier with the availability of the segmentation datasets across the Internet and in machine learning frameworks per default. We present a new dataset that we used to train the semantic segmentation state-of-the-art models. We have combined selected subsets collected from three known datasets which are **COCO_stuff164K** [5], **PASCAL VOC 2012** [17] and **ADE20K** [48, 54, 53]. **COCO_stuff164K** is one of the largest datasets in image segmentation. It has been created from 123k gathered images of complex everyday scenes and contains photos of 91 objects in their natural scenes. Meanwhile, **PASCAL VOC 2012** has 20 classes and contains a total of 6,929 train and validation sets combined. Finally, **ADE20K** contains 25,574 annotated images of objects and parts with over 150 classes. These datasets were selected for containing surveillance-like images and having a human/person as a class. Combining these datasets was easy since the images in these datasets have a similar size originally. The resulting dataset of this fusing was cleansed and preprocessed to preserve only those images having human class and are surveillance-like. (See section 3).

In this study, we aim to evaluate and compare the most successful deep-learning-based state-of-the-art approaches for semantic segmentation and find out which algorithm will fit in with the speed and performance requirements for real applications. We provide an overview and a detailed discussion of their aspects and insights through experiments analyzing their behavior and performance on a surveillance-like dataset while considering their accuracy, speed, and complexity. Furthermore, we provide a real-time visualization of a privacy-protected surveillance video to remove the risk of detection of person-sensitive data.

Using the open source project called **mmsegmentation** [11] that serves as a semantic segmentation toolbox and benchmark, we collected the results of speed and accuracy they got on the **CityScapes** dataset [12]. We run a comparison of the collected data to make it easier for us to choose which models to include in this research. To begin with, **BiSeNet** [52] was added for its novelty and state-of-the-art results in real-time semantic segmentation. **BiSeNetv2** [51] follows it for two reasons in addition to its proposed training strategy. 1) Improving from **BiSeNet** with a better balance between speed and accuracy 2) The proposed design of a Guided Aggregation Layer simplifies the previous model. Due to its simple and efficient structure, **STDC** has achieved amazing results. Present a very high level of improvement over **BiSeNet**. The reason for choosing **STDC** [18] is not only for its impressive results but also for its thoughtful design of a Detail Ground-Truth generation and guidance modules. **SegFormer** [50] innovative architecture built on Mix Transformer (MiT) encoders. **UNet** [34], being a classical standard, serves as the benchmarking model for our comprehensive comparison. This architecture shows elasticity because it can adapt to any test resolution different from the train resolution.

BiSeNet achieved 31.77 FPS and 74.44 mIoU. Similar results were achieved by **BiSeNetv2** with 31.77 FPS and 73.21 mIoU. While **STDC** achieved 23.06 FPS and 71.82 mIoU and 23.71 FPS and 73.15 mIoU for its version with strides of two. **SegFormer** B1 was a bit slower with 4.3 FPS, but still got good results with 78.56 mIoU. Their speed benchmark was conducted on 8 NVIDIA Tesla V100 (32G) GPUs and Intel (R) Xeon (R) Gold 6148 CPU @ 2.40 GHz.

The reminder of this paper is structured as follows: Section 2 provides an overview of related work; Section 3 describes our method for building a subset of combined datasets focused on human segmentation; Section 4 provides a detailed overview of five selected state-of-the-art models that were chosen according to their speed and performance; Section 5 summarizes our results and analysis; Section 6 discusses potential additional applications and the future direction of our work; Section 7 presents our final takes and conclusions.

2 Related work

Image segmentation is an essential topic in Computer Vision. It has three subcategories: semantic segmentation, instance segmentation, and panoptic segmentation. Semantic segmentation is similar to Object-detection. Its main goals are to detect and classify objects by providing labels for all or certain objects in an image. When it comes to localization, semantic segmentation provides boundaries for all objects in an image instead of Object-detection bounding boxes to give them a spatial location. However, Instance segmentation segments each instance of an object in an image as if it were a different class, more specifically, assigning the pixels to pairs (l, z) where l is the class label and z the instance id. Panoptic segmentation is similar to Instance segmentation but, each pixel can only be assigned to one and only one (l, z) pair. [26]

Human Segmentation is a sub-task of semantic segmentation that can be widely applied to the fields of augmented reality devices and video surveillance. On smartphones, one can apply filters like changing the background. Changing hair and eye color as is the case for human part segmentation as **PASCAL-Part** [8] offers something further than object segmentation by providing a segmentation mask for each body part which opens the door for much more applications like altering human faces for example. Human Segmentation can be beneficial for video surveillance for a variety of purposes and applications like analyzing human behavior in monitored environments by providing prediction and identification of intruders by analyzing their behavior, relying in a very significant way on the accuracy of the segmentation process.

Although this task is useful for video surveillance, it is challenging, such as having a moving camera, person appearance variations, and illumination changes. In addition, there is a lack of high-quality annotations as it remains challenging to obtain annotations of entire human body in images that are similar to real-life situations. However, the availability of datasets including a person or human class online made this task easier. But, the use of a deep neural network trained with a specific dataset guarantees a high level of precision. There are many methods and approaches in deep learning that were developed to tackle this specific task by using convolutional network architectures. These networks were designed to achieve a high inference speed and decent segmentation accuracy.

2.1 Semantic Segmentation Task

Given an image of a person standing by his car, we want to classify that image. In image classification, a classifier most likely will classify the image as a human image while ignoring the car, or vice versa. While an image segmentation model will classify each pixel of the given image instead of the whole image. In this case, the semantic segmentation model will assign a person label to each pixel that makes up the human in the image. In addition, it will assign the car label to each pixel representing the car. Semantic segmentation is used and applied as mentioned in the previous section in autonomous cars, video surveillance, and image editing such as changing the background in video calls. This task is also involved in the medical field, for example, in diagnosing tumors in brain cells segmentation task as in [34].

In short, the task of semantic segmentation is not new and has a lot of history. There are quite a few algorithms that were designed to solve this task, such as Edge detection [37, 4, 6], Watershed-based algorithms [43], image thresholding [28], and K-means clustering as in [45]. Watershed basically, identifies each pixel in an image as being occupied generally associated with the region minimum to the location where the drop of water flows toward single or multiple region minimums. Image histogram thresholding is a spatially blind technique. Its concept is based on image segmenting by identifying peaks, valleys, and shapes in the image intensity histogram. And clustering techniques such as K-means is also spatially blind. K-means analyze and partition images by minimizing a predefined metrics function, usually the Euclidean distance function, to identify meaningful groupings of pixels that are known as clusters. Also, it's worth mentioning the most commonly used based edge detection algorithms such as Roberts edge detection, Canny edge detection, Prewitt edge detection, and Sobel edge

detection [4, 37]. Recently Convolutional neural networks outperformed the existing techniques based on speed and accuracy by providing promising results that can be used in real-world applications [41].

The most commonly used loss function in the Image segmentation task is pixel-wise Cross entropy loss which averages the cross entropy of the prediction of each pixel over all pixels. Another loss function that is mostly used in this task is Dice coefficient loss which measures the overlap between the predicted segmentation map and ground truth. It ranges between 0 and 1, where a Dice coefficient of 1 denotes perfect and complete overlap.

$$Dice = \frac{2\,|y \cap \hat{y}|}{|y| + |\hat{y}|}$$

There are quite a few metrics used in the Image segmentation task we count: Pixel accuracy as the sum of predicted positively pixels of class *i* over the total sum of pixels of class *i*; Intersection over Union as the ratio of the intersection of the predicted segmentation map and ground truth over their union; the mean IoU for multi-class problems which can be weighted with the frequency of each class compared to each other; finally, Dice coefficient and F1 score.

$$IoU = \frac{|y \cap \hat{y}|}{|y \cup \hat{y}|}$$

2.2 Deep learning basics

Machine learning systems are widely used nowadays. In identifying objects in images recognizing human voices and even detecting fraud in email messages detecting anomalies in a computer file system. Deep learning resides as part of machine learning methods. Each ML solution always has an ML model and data. And data is a key component of ML solutions. ML types are based on the motivation and the type of data used to solve the problem. **Supervised Learning** is a type of machine learning in which the function is learned from labeled data. Similarly, **Unsupervised Learning** is when learning a function from input data patterns without knowing its corresponding output (unlabeled data). It is often used in probabilistic reasoning. On the other hand, **Semi-supervised Learning** is a combination of the two, both labeled and unlabeled data can be used to learn the function. Meanwhile, **Reinforcement Learning** enables learning based on the state of the environment. Let's assume we have an agent in an accessible environment. As the agent interacts with the environment, its state changes accordingly. Then, the agent reads the environment's new state and learns from it. Receives a reward or gets penalized depending on whether the action taken by the agent was accurate or not.

Biological human brain. Because the human brain is the most advanced system that we know. Each brain cell is a single neuron (also known as a nerve cell). The neuron dendrites receive electrical signals from the axons of other neurons. The **Perceptron** [32] is the mathematical model of a biological neuron. And these electrical signals in the perceptron are represented with a numerical value. The perceptron is like logistic regression. It has weights ω and outputs a function of the dot product of the weights and the input values. Finally, the step function will return 1 if the product is positive and 0 otherwise. So in short, it is a mathematical function where the input data represents the function arguments and the network weights, ω, are its parameters so it takes one or more input values and outputs one single numerical value. And we can consider this as an example of a simple feed-forward network.

$$f(x) = \sigma(\sum x_i \omega_i + b)$$

Where **b** is a bias that is always the value 1 and its associated weight is equivalent to a threshold of the opposite sign and is treated like any other weight. σ is the activation function. ω represent the weights. **x** represent the input value.

After doing the sum of the product of the input values with the weights. The result of this sum is the input to the activation function σ. They all have satisfied the requirements to be non-linear [23]. Activation functions are Identity functions where the output is the same value as the input, the Binary threshold function returns 1 if the input value is positive and 0 otherwise, and the Logistic sigmoid function is a non-linear function that returns a value between 0 and 1 defined as follows $f(x) = \frac{1}{1+e^{-}x}$, Hyperbolic tangent it is a continuous non-linear function return values between -1 and 1 and like sigmoid, it is computationally expansive because they are exponential functions. Hyperbolic tangent function is defined as follows: $f(x) = \frac{1-e^{-}x}{1+e^{-}x}$, Rectified Linear Unit (ReLU) and its variations like the noisy ReLU, Leaky ReLU, Exponential Linear Unit (ELU). ReLU returns the same input value if it is positive and 0 otherwise. Finally, the SoftMax activation function squashes the input values in 0,1 intervals like the logistic function. The sum of all the squashed outputs adds up to 1. SoftMax normalizes the output properties to the probability distribution of classes [21].

Combining the perceptrons in layers can yield the possibility to learn a lot more information, such as learning the **XOR** logical function [21]. The perceptron can output a value but a set of perceptrons can output a vector of values. This is not only because the vector has multiple values, but also because the relative ratios between them have additional information. A network is a deep feed-forward neural network when at least has one hidden layer.

While the loss function computes the difference between the generated output of the deep network with the given ground truth, optimizers are used to optimize the parameters to minimize the loss function. Gradient descent is the most common algorithm. Stochastic Gradient Descent (SGD), computes the cost for each input data before each parameter update, instead of computing the gradient for the whole dataset [38]. This is what makes it less redundant than batch gradient descent. Mini-batch gradient descent is very efficient and computes the gradients on small batches of data. Adagrad performs small updates to the parameters associated with frequent features and big updates for the parameters associated with the rarest ones. Adagrad, unlike SGD, has a different learning rate for each parameter at every step. It is highly possible to arrive at a very low learning rate after a long training time and prevent the network from learning [25]. Meanwhile, the Adam optimizer (also called adaptive moment estimation) maintains a separate adaptive learning rate (or learning rates schedules) for each parameter and weighted average of momentum. The algorithm updates exponential decaying averages by combining momentum [36] and the Adagrad algorithm [25].

Let's say we have a simple feed forward network where the input layer is directly connected to the output. It is easy to adjust the weights (or parameters) to reduce the difference between y and $\hat{y}$ (ground truth and the network predicted output). When it comes to an MLP architecture, we can compute the gradient (or the derivative) of the loss function to the weights of the network for a single input-output to find the global minimum of that loss function. Computing each layer in the chain rule at a time applying the back-propagation function on the model and iterating backward, propagating the derivatives from the top layer (or output layer) and back to the bottom layer (input layer). To find the combination of weights that reduces the difference between$y$ and $\hat{y}$.

Feed forward neural networks have several types of special layers. For example, CNNs are designed the same way biological cells are organized in the visual cortex of the brain. The convolutional layer slides a filter over the incoming signal. A convolutional layer neuron acts the same way as the perceptron but, it only takes input from a small number of neighboring neurons from the previous layer. These neurons are spatially close to each other. Meanwhile, Pooling layers split the input feature map into a grid, where each grid represents the Receptive Field of some neurons. Then, apply the pooling operation over each grid. In discovery pooling layers have some importance to capture global context and compute an attention vector to guide the feature learning [15]. Max pooling, preserves the features detected in its output. While, Average pooling, instead of taking the maximum within each grid, takes the average. Global average pooling generates one feature map for each corresponding category of the classification task in the last Convolutional layer. One advantage of global average pooling over the fully

connected layers is that it is more native to the convolution structure by enforcing correspondences between feature maps and categories. Global average pooling sums out the spatial information, thus it is more robust to spatial translations of the input. The role of the convolutional layer is to detect local conjunctions of features from the previous layer. The role of the pooling layer is to merge semantically similar features into one [15].

2.3 Deep learning segmentation review

Semantic segmentation requires both spatial information and sizeable Receptive Fields to extract important features from the input image [52]. These requirements provide some freedom and trade-off between accuracy and speed for semantic segmentation models. Most modern approaches boost their speed by reducing the input image size, which might result in a huge loss of spatial information; reducing the number of channels in the standard model might reduce the model complexity and boost the speed, but might damage spatial information and lower the segmentation quality (as mentioned in [52]). Similarly, adding more channels or using a U-nified structure to increase the spatial resolution of each layer feature map decreases the inference speed due to the additional computation.

Moreover, spatial resolution refers to the ability to distinguish two objects that are close to each other, not only by the number of pixels in an image but also by the number of independent pixel values per unit length. While Receptive field in CNNs is the size of the area of an output feature of any layer to the input path for the next layer. Meaning that each neuron in a layer receives connections only from a subset of neurons in the previous layer. Thus, the Receptive field of a neuron in the higher layers is formed from a combination of the Receptive fields of a subset of the neurons from the previous layer. In this way, higher layers are capable of learning all the crucial and abstract features that were not taken into account in the lower layers [15]. Some of the tricks to increase the Receptive field in convolutional neural networks. 1) Adding more convolutional layers which will increase the RF linearly. 2) Adding pooling layers or convolutional layers with higher stride to increase the RF multiplicatively or combined with Dilated convolutions to increase it exponentially. 3) Adding skip connections to learn features increases the range of different Receptive fields, but it tends to make the Effective Receptive Field smaller [52].

UNet: Built upon an architecture called "Fully convolutional network". An **FCN** model consists of only convolutional layers [41]. The network output will be a segmentation map of the input image instead of a standard classification/regression score. **UNet** extends this architecture to work with less training images and gives more precise segmentation maps. It was considered efficient at its time (8 March 2015). **UNet** inference takes less than a fifth of a second for a typical image.

BiSeNet: Also known as Bilateral Segmentation Network. This state-of-the-art model presented quite interesting results in both speed and accuracy. Implemented in 2018 to tackle the loss of spatial resolution and shrinkage of the Receptive Field to accelerate inference speed. **BiSeNet** proposes using two paths to capture spatial information and generate high resolution features preserving the spatial size of the original image while the other path is addressed to provide sufficient Receptive Field utilizing a lightweight model combined with fast downsampling strategy to encode high-level semantic context information.

BiSeNetv2: The second version of the Bilateral Segmentation network, implemented in 2020. Has two branches; the Detail branch to capture spatial information with wide channels and shallow layers; the Semantic branch to extract semantic details with narrow channels and deep layers. **BiSeNetv2** removes **BiSeNet** cross connections to make the architecture clearer, simpler, and more effective. It redesigns the lightweight model based on depthwise convolutions. Meanwhile, it designs a Guided Aggregation Layer to enhance connections and fuse both paths with different feature representations, in addition to adding a booster training strategy to improve segmentation accuracy without diminishing the inference speed. The booster head can be inserted at any position on the semantic branch.

STDC: In April 2021, **STDC** addressed the problem that adding an extra path like **BiSeNet** does, is time-consuming, and using backbones borrowed from different tasks e.g. the Image classification task, provides inferior segmentations as it is not designed for this task. Short-Term Dense Concatenate network reduces **BiSeNet** feature map dimensions and removes the extra path for spatial information as it is considered a lack of low-level information guidance. The **STDC** model is integrated into a UNet structure due to the additional skip connections in the encoder part to encode input features using different respective field sizes. The decoder part integrates a Detail aggregation module which is not required in the inference phase to generate the detail map ground truth from the segmentation ground truth by Laplacian operator to integrate the learning of low-level information into low layers. Finally, the low-level information and high-level details are fused to output the final segmentation predictions.

SegFormer: SegFormer is a simple encoder-decoder model. In October 2021, this segmentation model feeds on each image as a sequence of tokens to its hierarchical Transformer encoder model and extracts coarse and fine features, and passes them to its lightweight MLP decoder model to fuse these multi-level features to produce the final semantic segmentation mask. However, using the original multi-head self-attention process comes at a quadratic cost. **SegFormer** instead uses the sequence reduction process to reduce the length of the sequence. This state-of-the-art model can be scaled from **SegFormer-B0** to **SegFormer-B5** to enhance performance and efficiency.

3 Dataset

We want to make a fair comparison between the selected models. We are interested in conducting a complete evaluation and analysis of a surveillance-like database. At the same time, we want to provide real-time visualization of a privacy-protected surveillance video and remove the risk of detection of person-sensitive data. We generated a new dataset from several datasets. Those datasets were chosen based on their content, and the number of images that are segmented and contain human-labeled pixels. From a long list of datasets, **VOC PASCAL 2012** [17] had 11,355 segmented images, 4,087 of them included a human label. While, **COCO_stuff164K** [5] - had 118,287 images for training and 5000 for testing. After digesting this huge dataset, we found that there were 66,646 images in total that contained human-labeled pixels. **ADE20K** [48, 54, 53] - only had 20210 images for training and 2000 for validation and 3352 for testing. We found that 5,579 were segmented and contained human labels. Other interesting datasets were excluded because they did not include human/person as a separate class or did not have at least a minimum amount of images. These datasets are **BSDS500** [2], **ICCV09DATA** [22], **CCMData2008** [24], **cvp10Data** [29, 39, 40] and **MSRC** [31].

Combining these datasets was easy since the images in these datasets had a similar size originally. We managed to group 76,285 unique images containing human-labeled pixels. Later, it was cleansed and preprocessed to preserve only those images we considered surveillance-like.

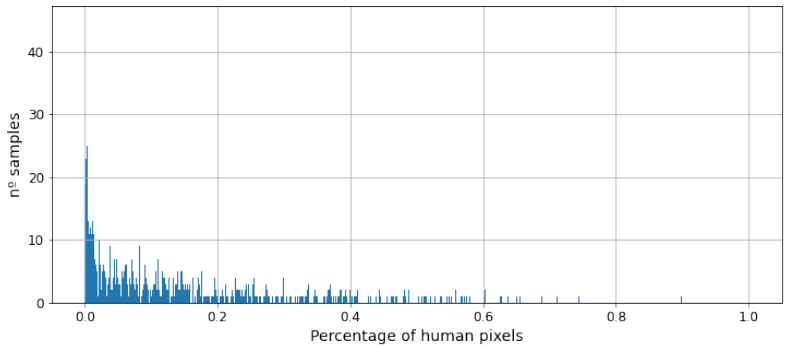


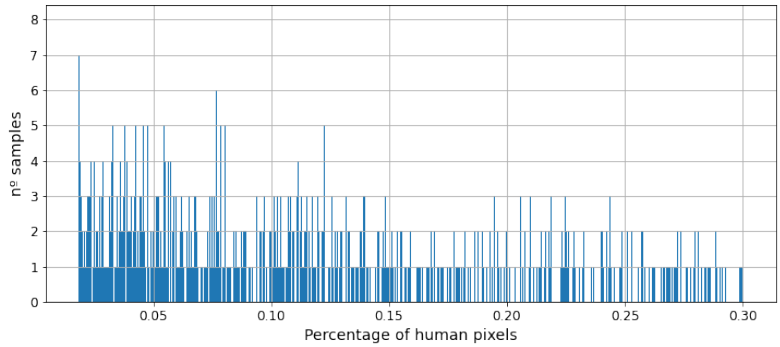
Figure 1: Visualization of the distribution of the human pixels over the full dataset before applying data cleansing.

First, it is important to define what types of images are surveillance-like and, at the same time, any privacy safeguard technique can be applied to them. According to our observation, we found that surveillance-like images normally cover a wide area trying to cover all possible angles from the camera position. Hence, the percentage of human pixels in such images is typically low compared to the camera's field of view. A privacy safeguard technique used in video surveillance will not be necessary if the human is very far away and almost visible in the image. Hence, images with a small percentage of human pixels can be discarded. To be fair in our analysis, we removed all the images that did not comply with these defined conditions.

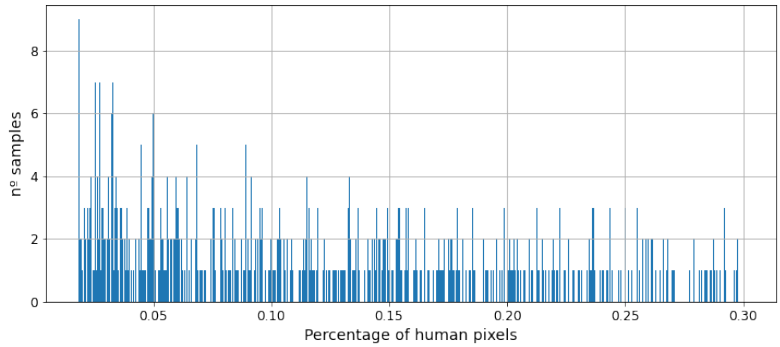
Two thresholds and milestone's human detector software helped in cleansing the dataset. The accepted thresholds that are needed to define a surveillance-like dataset are not specific in our case, at least at the moment. The images that exceeded the upper threshold or fell below the lower threshold were removed. Figure 1. Illustrates how the number of images with a high percentage of human pixels is relatively low, particularly after 30% of human pixels. At the same time, the number of images with a very small percentage of human pixels e.g. 2% is

significantly high. Applying these thresholds removed 29,726 not needed images. Then, we used milestone's human detector software to remove 2,420 images that did not contain an identifiable human in them. In addition to that, we have added back the equivalent of 10% of the new dataset's total size with images that do not contain a human/person class.

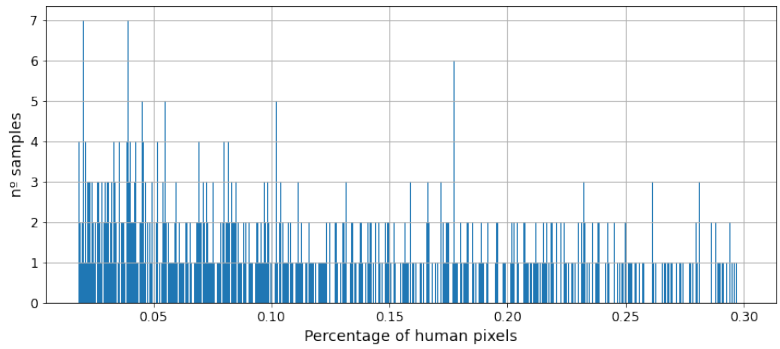
We are interested in segmenting humans only. As a result, there were over 20 classes in **VOC PASCAL 2012** and 150, 183 classes in **ADE20K** and **COCO_stuff 164K**, respectively. To remove the other classes, we iterated over the different datasets and applied the following: 1) Identify the human class id. 2) Iterate over all the images in that dataset and modify its correspondent annotation map to have 1 for human pixels and 0 for any other class (the added images with different objects got assigned null-matrices instead of modifying their annotation maps). Ultimately, the final dataset was divided into three separate datasets for training, validation, and testing. The training set comprises 85% of the total size, the validation set 5% and the testing set 10%. Figure 2. Show the distribution of human pixels within the dataset.



(a)



(b)



(c)

Figure 2: Visualization of the distribution of the human pixels over the full dataset after applying data cleansing and splitting it into three separate datasets for training, validation, and testing. (a) Training set. (b) Validation set. (c) Test set. The overall distribution of human pixels percentage per sample is the same for all subsets. The three resulting datasets have an **11%** mean percentage of human pixels. This way we can conduct a comparison of our models and evaluate them fairly.

4 Methods

This section provides an overview of recent deep learning approaches for real-time semantic segmentation, chosen based on their speed and precision. Most of these models focus mainly on providing a state-of-the-art approach to real-time semantic segmentation. We focused in this paper on studying five of these models which are **UNet**, **BiSeNet**, **BiSeNetv2**, **STDC**, and **SegFromer**.

4.1 UNet

UNet is built on another structure called **FCN**, which convolutionalizes proven classification approaches such as **AlexNet**, **VGGNets**, and **GoogLeNet**. To construct this structure, the last classifier layer of each net was removed. All connected layers were converted to convolution layers. Finally, adding a final 1x1 convolutional layer with a channel dimension equal to the number of classes in the dataset. **UNet** modifies and extends it to work with far fewer training images according to their data augmentation technique yet, yielding more precise segmentation by replacing pooling layers with up-sampling operators to increase the resolution of the output. Thus, this converted these layers into what is called an expansive path that is, more or less, symmetric with the contracting path. Therefore, it also has a large number of feature channels that allow the context information to propagate to higher resolution layers. The output of the layers from the contracting path is combined with their equivalent at the expansive path to localize more high-resolution features. Hence, the network yields a U-shape architecture.

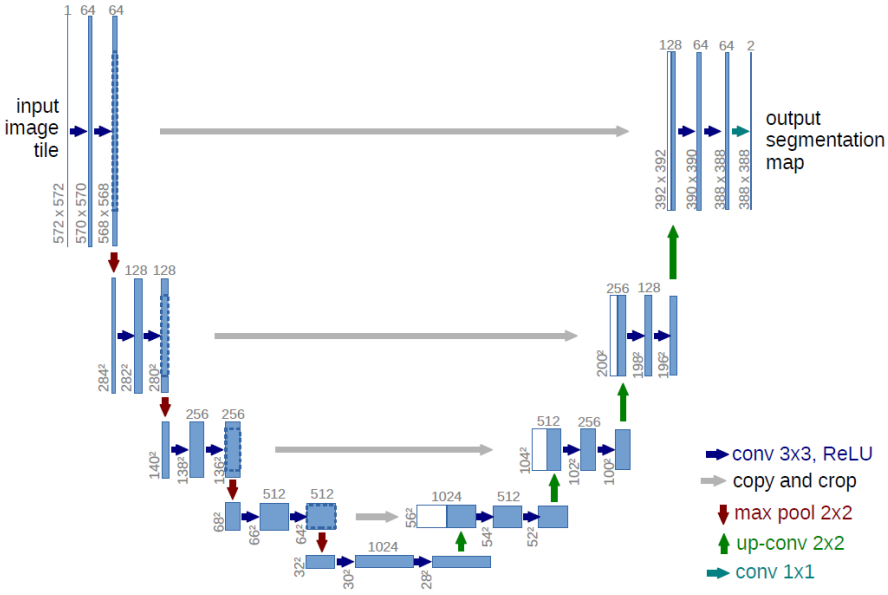


Figure 3: Obtained from [34]. UNet architecture. Blue boxes are 3x3 convolutional layers, with Red arrows for down-sampling and green arrows for up-sampling. Gray arrow for cross connections between the contracting path and the expansive path.

With more details, **UNet** is an encoder-decoder FCN model that is composed of two paths, a contracting path, and an expansive path. The contracting path mainly consists of repeated two 3x3 convolutions, each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with stride = 2 for down-sampling. At each

step, it doubles the number of feature channels. Every step in the expansive path consists of an up-sampling of the feature map followed by a 2x2 convolution ("up-convolution") that halves the number of feature channels. Good localization (spatial information) and use of context details and spatial information are possible by adding concatenation layers with the correspondingly cropped feature map from the contracting path to the expansive path, followed by two 3x3 convolutions, each followed by a ReLU. The cropping is necessary due to the loss of border pixels in every convolution. At the final layer, a 1x1 convolution is used to map each feature vector to the desired number of classes. In total, the network has 23 convolutional layers. It is important to select the input tile size so all 2x2 max-pooling operations are applied to a layer with an even x- and y-size.

4.2 BiSeNet

Two concurrent paths model. Works on solving the loss of spatial information problem and shrinkage of the Receptive Field while achieving real-time speed. Its two paths are the Spatial path to retain affluent spatial information and the Context path which is designed to provide a sufficient Receptive Field.

The spatial path is formed from three layers. Each layer is a convolution layer with a stride of 2, followed by Batch normalization and ReLU activation function. Therefore, this path extracts a feature map that is 1/8 of the original image. Due to the large spatial size of its feature maps, this path encodes rich spatial information. Meanwhile, the Context path utilizes a lightweight model combined with fast down-sampling strategy, which can encode high-level semantic information (context details). The added global average pooling the tail of the lightweight model to provide the maximum Receptive Field with global context information. The lightweight model fused the features of the last two stages with the up-sampled output features of the global average pooling which yields an incomplete U-shape structure.

The **Attention Refinement Module** is part of the context path to refine the features at each stage. First employs global average pooling to capture global context and computes an attention vector to guide the feature learning. It demands low computation cost as it integrates the global context information without any up-sampling operations.

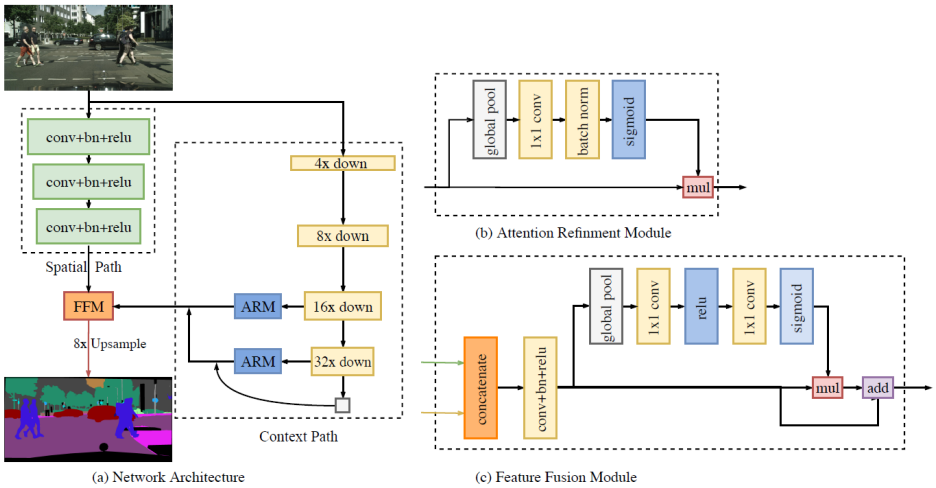


Figure 4: Obtained from [52]. (a) BiSeNet architecture. The length of blocks indicates the resolution of its output features map while the thickness of each block represents the channel capacity. (b) Components of the Attention Refinement Module. (c) Components of the Feature Fusion Module.

Feature Fusion Module to fuse the features from the spatial path with the features from the context path. This is due to the difference in the level of feature representation between both paths. It first concatenates the output features of spatial path and context path. Then, utilize a Batch normalization to balance the scales of the features. And globally pools the concatenated features to the feature vector and computes a weight vector to re-weight the features realizing feature combination and selection.

4.3 BiSeNetv2

In an updated version of **BiSeNet**, the architecture has two paths: one with wide channels and shallow layers to capture low-level spatial information and another with narrow channels and deep layers to extract high-level semantic information (context details).

This model has a simpler structure than **BiSeNet** to achieve an effective and more efficient architecture for real-time semantic segmentation. **BiSeNetv2** removes the cross-layer time-consuming connections from the context path in the original version. And, redesign the overall architecture, especially, with an improved lightweight model component based on depth-wise convolutions. While adding an efficient aggregation module to enhance the fusion between the two paths.

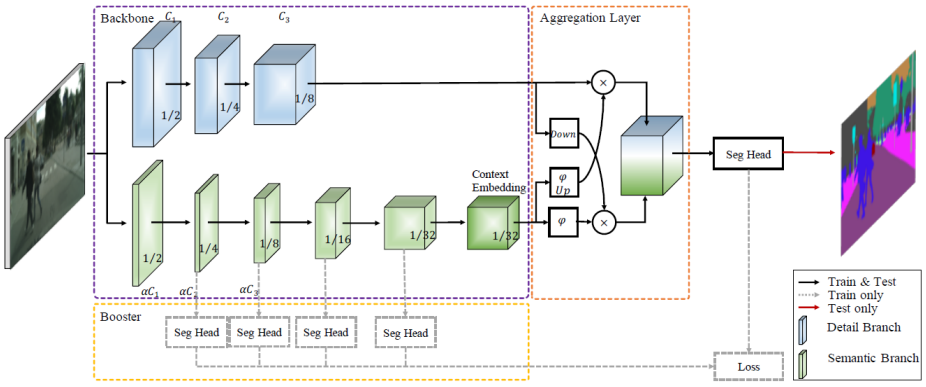


Figure 5: Obtained from [51]. BiSeNetv2 architecture. It's divided into three components: the blue dashed line represents the two-path way backbone, blue boxes for the detailed branch while green boxes are the semantic branch; Yellow dashed lines contain the training booster part; finally, the red dashed line represents the guided aggregation layer.

The semantic branch is the first pathway in the model. it is a lightweight model with low channel capacity combined with a fast down-sampling strategy to provide a large Receptive Field and employs global average pooling to obtain global contextual details. The first stage of the semantic branch is called stem block which uses two different down-sampling strategies, a pooling layer with stride = 2 and one max pooling layer in another path to shrink the feature representation and then concatenate both results as its output with an efficient computation cost. Lastly, the Context Embedding Block uses global average pooling and residual connections to embed global contextual information. The Gather and Expansion layer is a 3x3 convolution followed by a 3x3 depth-wise convolution and 1x1 convolution to project the output of the depth-wise convolution into a low channel capacity space. The Gather and Expansion layer is friendly to the computation and memory access cost.

The detail branch is meant to capture spatial information using its wide channels and shallow layers. Therefore, this branch has a small Receptive Field. It is composed of three stages. At each stage, each layer is a convolution

layer followed by batch normalization and activation function. The first layer has a stride of 2 while the successive layers at the same stage have the same number of filters and same output feature map size.

Guided Aggregation Layer up-samples the output feature map of the Semantic Branch to match the output of the Detail Branch and down-samples the output feature map of the Detail Branch to match the output of the Semantic Branch. This layer employs the contextual information of the Semantic Branch to guide the feature response of Detail Branch scale feature representation, which inherently encodes the multi-scale information.

The booster training strategy enhances the feature representation during the training phase which later can be discarded in the inference phase. It can be inserted in different positions in the Semantic branch. The complexity of the Booster head can be adjusted by changing the channel dimensions. The booster model is formed from 3x3 convolution + Batch Normalization + ReLU activation followed by one convolution later to up-sample the output feature map of the head.

4.4 STDC

Short-Term Dense Concatenate Module replaces the extra path of **BiSeNet** with the **Detail Aggregation Module** to guide the low-level layers for the learning of spatial details by generating a binary detail ground-truth without the extra computation cost during the inference time. **STDC** considers that **BiSeNet** extra path is time-consuming and the borrowed pretrained backbones from other computer vision tasks, like the Image classification task may affect poorly the model segmentation predictions. To preserve multi-scale respective fields and enrich the spatial information extraction. From the first layer to the last layer feature maps are concatenated as the output of the **STDC** module with skip connections. Before any concatenation, the different blocks in the **STDC** module are down-sampled to the same spatial size by average pooling operation with a 3x3 pooling size.

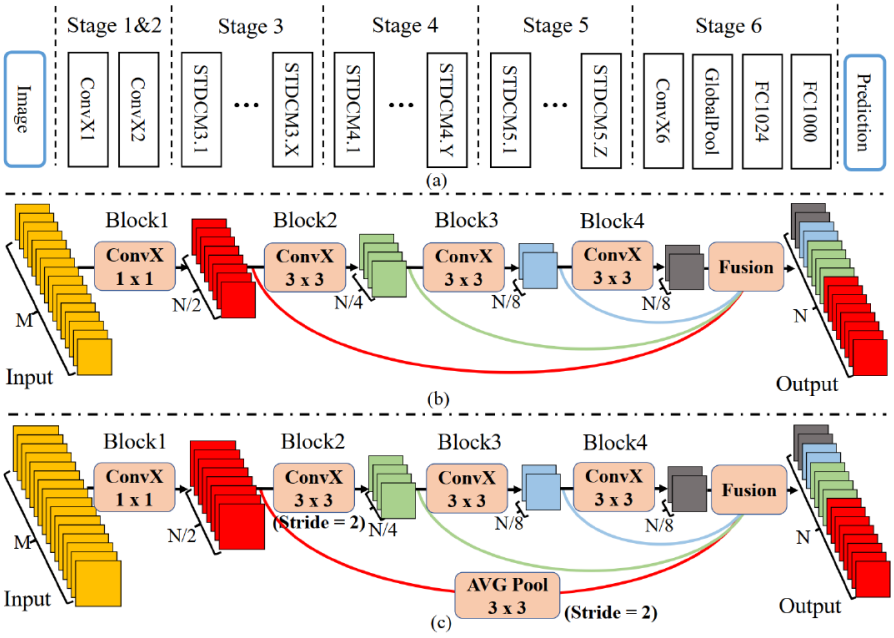


Figure 6: Obtained from [18]. (a) General STDC network architecture. (b) STDC module. (c) STDC with stride = 2.

The model structure is formed from 6 stages. Stage 1 - 5 down-sample the spatial resolution of the input with a stride of 2, and the kernel size is 1 for the first layer and 3 for the rest of the layers. While stage 6 outputs the prediction maps by one convolution layer, one global average pooling, and two fully connected convolutional layers. Stages 1 and 2 are considered low-level layers. For efficiency's sake, there is only one convolutional block in each stage. Each convolutional layer is followed by Batch normalization and ReLU activation function. Stages 3 to 5 are used to produce the feature maps with a down-sample of 1/8, 1/16, and 1/32, respectively. Global average pooling to provide global context information with a large Receptive Field. A U-shape structure is used to up-sample the features derived from the global feature and combine them with the counterparts obtained from the last two stages 4 and 5. Feature Refinement module from **BiSeNet** is used to fuse the 1/8 down-sample from stage 3 in the encoder and the counterpart from the decoder. And the Segmentation Head includes a 3x3 convolution + Batch Normalization + ReLU activation function followed by a 1x1 convolution. This will set the output dimension as the number of classes in the dataset.

Detail Aggregation Module is designed to replace the extra detail path responsible for extracting spatial information and generating the detail ground truth from the segmentation ground truth using a Laplacian operator. The detail head gets inserted at stage 3 to generate the detail feature map. These detail ground truth maps are used to guide the low-level layers to learn the features of spatial details. It can encode more spatial information than **BiSeNet**. The learned detail features are fused with the context features from the deep block of the decoder for the segmentation prediction.

The **Detail Aggregation Module** generates detail ground truth from the semantic segmentation ground truth. This operation can be carried out by a 2D convolution kernel called Laplacian kernel and trainable 1x1 convolution. The Laplacian operator is used to produce soft, thin detail feature maps with different strides to obtain multi-scale detail information. Then up-samples the details feature maps to the original size and fuse it with a trainable 1 x 1 convolution for dynamic re-weighting. Finally, adopt a threshold of 0.1 to convert the predicted details to the final binary detail ground-truth with boundary and corner information. In the Detail Head, a detailed ground truth map is produced to guide the shallow layers (low-level layers) to encode spatial information. Includes a 3 x 3 convolution + batch normalization + ReLU operator followed by a 1 x 1 convolution to get the output detail map.

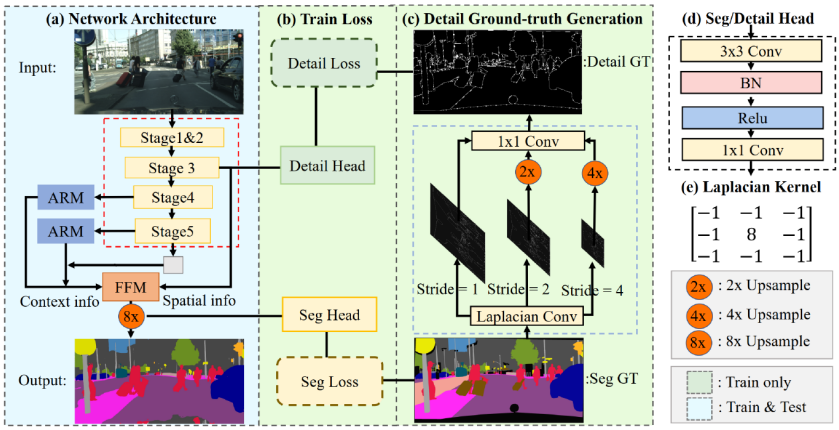


Figure 7: Obtained from [18]. STDC segmentation network. The red dashed line is the STDC network while the blue dashed box is the Detail Aggregation module. ARM is the Attention refinement module and FFN is Feature Fusion Module. The operation in the green box is used in training only.

Cross entropy and dice losses are combined to optimize the details learning. Dice loss measures the overlap between predicted maps and ground truth. Since it is insensitive to foreground/background pixels, it can alleviate the problem of class inequality.

4.5 SegFormer

A powerful encoder-decoder model. Inspired by ViT [13], MiT is more optimized for the semantic segmentation task. The SegFormor encoder part can be scaled from MiT-B0 to MiT-B5 to enhance performance and efficiency. Same architecture but different sizes. MiT-B0 is the fastest with the poorest results while MiT-B5 is the slowest yet has the best results. This segmentation model treats each image as a sequence of tokens and feeds them to a hierarchical Transformer encoder model and extracts multi-level fine features and feeds them to a lightweight MLP decoder model to fuse these multi-level features to produce the final semantic segmentation mask.

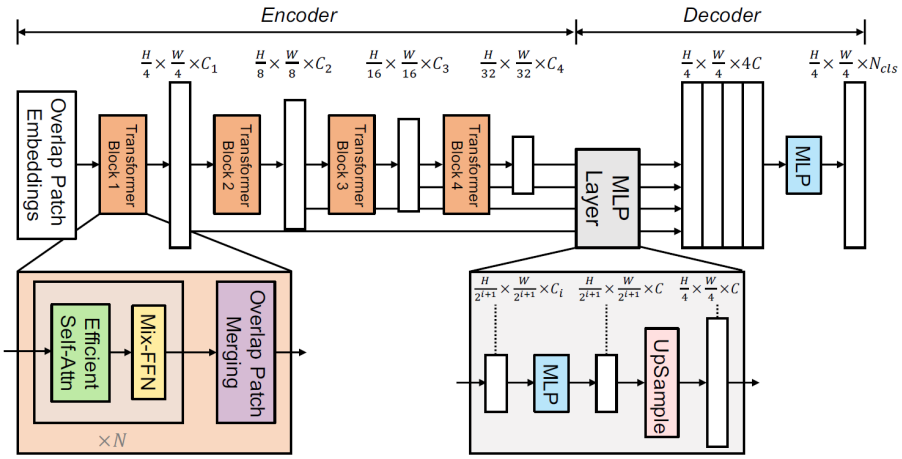


Figure 8: Obtained from [50]. SegFormer architecture. it's an Encoder-decoder architecture where the encoder part is a hierarchical transformer model and the decoder is a lightweight MLP model.

First, split the image into fixed size patches. Then, take each patch and flatten it into a vector using a trained flattened matrix to transform it into a linear vector or a sequence called patch embedding. These patch embeddings are the input token feed to the transformer encoder part [13].

Self-attention is the transformer's main component and the most computationally expensive. Using the traditional multi-head self-attention (MHA) comes at a quadratic cost, instead, **SegFormer** uses the spatial reduction attention (SRA) process. By using the progressive shrinking pyramid (proposed in x) to reduce the sequence length and generate multi-scale features. Thus, reduces its cost from O(N^2) to O($\frac{N^2}{R}$) where R is reduction ratio and N is the length of the sequence H x W [49].

Transformers do not have any idea about each token's spatial location. To add spatial information to the transformer we need to add a positional encoding to the embedding vector. **SegFormer** Mix-Feed Forward Network uses a 3x3 convolution to provide positional information for Transformers. And adding 3x3 depth-wise convolutions reduces the number of parameters, improving performance [46].

The decoder part is a lightweight MLP model. The enabler of having lightweight models is the large size of the encoder's Effective Receptive Field (ERF). Where the key idea is to take advantage of the Transformer-induced

features. The MLP decoder's first layer unifies the multi-level features of the encoder. Then, the MLP decoder up-samples to 1/4th and concatenates them together after that, fuses the concatenated features using an MLP layer. Finally, another MLP layer predicts the segmentation mask from the fused feature.

5 Results & Experimental analysis

In this section, we first walk through the evaluation protocol that we applied to all the chosen models. First, outline the metrics we employed as well as the optimizer and data augmentation techniques we applied to the data for training and testing. And which hardware we used. Then, we review the experiments that we conducted to achieve these results. Later, we explain our findings by comparing the results for each model with the rest of the models. We also compare the results of our trained models with the pretrained models using the CityScapes dataset [12]. Finally, a visualization of the winner model results on our data to demonstrate how we extracted a privacy protection version of these data using semantic segmentation.

5.1 Evaluation protocol

We evaluated the models along three axes. Speed, complexity and performance. The most relevant evaluation metrics are: for speed we considered the number of frames or still images the inference can compute per second (FPS); for complexity, we took into account the number of parameters and floating points operations (FLOPs) required for a given input image shape; And finally, human category IoU for performance comparison.

Our evaluation tests were conducted using our dataset that we have cleansed and preprocessed accordingly to preserve only surveillance-like images (as mentioned in section 3). This dataset has 1 class for humans and another class labeled as background. The dataset has in total **48,530** images. 85% of the total size is taken up by the training set, 5% by the validation set, and 10% by the testing set. The three subsets have the same mean percentage of human pixels per image which is 11%.

5.2 Implementation details

To conduct our tests, we used the test pipeline, where images are cropped to 512x512, then normalized. Furthermore, the batch size is always equal to 1. We realized three experiments. 1) In the speed experiment, we used only one GPU. The experiment gets repeated five times with 200 batches each. We employ a batch size of 1 as noted earlier, which means one image is tested at each iteration. Finally, the average speed for all the repetitions is the final result for the speed test of the models. 2) The complexity experiment takes 512x512 as the model input shape, maps the model, and calculates the FLOPs for each layer in the model. 3) The performance experiment, the most relevant metric for this evaluation is the human IoU. In this test, each model is evaluated by running its inference for one testing epoch.

For a fair comparison. The same evaluation metrics, the same optimizer, and the same data augmentation techniques were applied to all models. This means the same batch size (or the same number of iterations per epoch) is used for training. We are using as the optimizer stochastic gradient descent (SGD) with a constant momentum equal to 0.9 and weight decay $5e^{-}4$.

The "poly" learning rate strategy [33] applied is $lr = lr0 * (1 - \frac{i}{T_i})^p$. Where lr0 is the initial rate and i denotes the number of iterations and T_i is the maximum number of iterations (Also, T_i can be calculated, as the total number of epochs per the number of iterations per epoch). The p value is 0.9. With this value the model learns faster and has better accuracy (according to [7] and [30]). The initial learning rate we used is $1e^{-}5$. As part of the training process, we applied a linear Warm-Up strategy starting with a low learning rate, increasing it up until it reaches the initial learning rate for the first epoch, and then decreasing it again with a specified step decay for the rest of the training. By doing so, the parameters can be tuned to achieve higher accuracy and avoid the chances of overshooting in the early stages of training. [33] The Cross Entropy loss function was applied to all models except **UNet**, which was combined with the Dice loss function [34].

For data augmentation, we follow **mmsegmentation** [11] recommendations for the data augmentation techniques in the training pipeline. We first, resize the input images to a scale preserving the aspect ratio. Then, crop the

image to size 512x512 and flip the image vertically or horizontally randomly with 50% probability. Using photometric distortion techniques such as Brightness distortion, Contrast distortion, Saturation distortion, and Hue distortion with a probability of 50% for each one of them to augment the dataset during training. Finally, normalize the image and convert it from BGR to RGB and apply a padding of the image to a fixed size of 512x512 for training [42].

All experiments are performed using PyTorch-1.11 in a 3.7 Python virtual environment on an NVIDIA GeForce GTX 1660 SUPER GPU with CUDA 10.2 and CUDNN 7.6.5.

5.3 Results

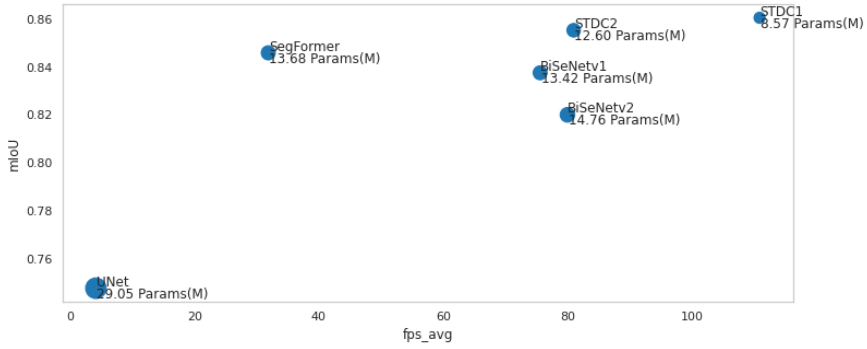
Three experiments were conducted on the selected models using our dataset to evaluate them over speed, complexity, and performance. Therefore, we compare the results of each test separately. Because we want to analyze them and find out which model will perform optimally in terms of accuracy on a surveillance dataset. However, this model should also be efficient in both complexity and speed.

Speed Analysis. For almost all the models included in this study, real-time segmentation is their target. Nevertheless, both low-level and high-level context information must be preserved for the semantic segmentation task to yield sufficient results. In our testing pipeline, images get cropped to 512x512 and normalized. We run the speed test five times for each model, where each time we run the inference for 200 batches.

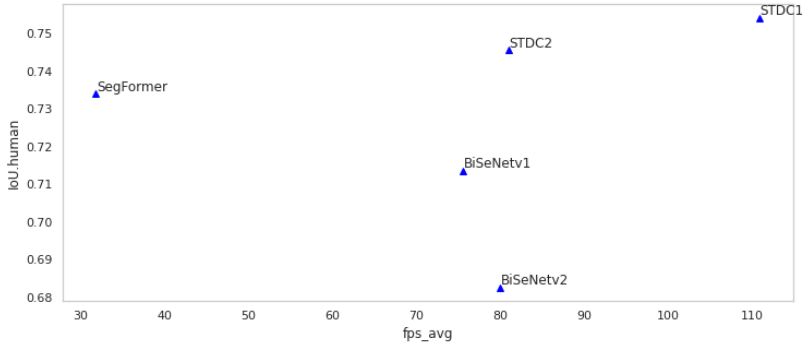
Method	FPS
UNet	4.13
SegFormer-b1	31.85
BiSeNet(ResNet18)	75.58
BiSeNetv2	79.98
STDC2	80.97
STDC1	110.88

Table 1: Speed comparison. The table is sorted in ascending order according to the FPS column. The speed results for the selected models. These results were obtained by running the inference for each model separately on one NVIDIA GeForce GTX 1660 SUPER GPU with CUDA 10.2 and CUDNN 7.6.5. Batch size equal to 1 and input image size equal to 512x512.

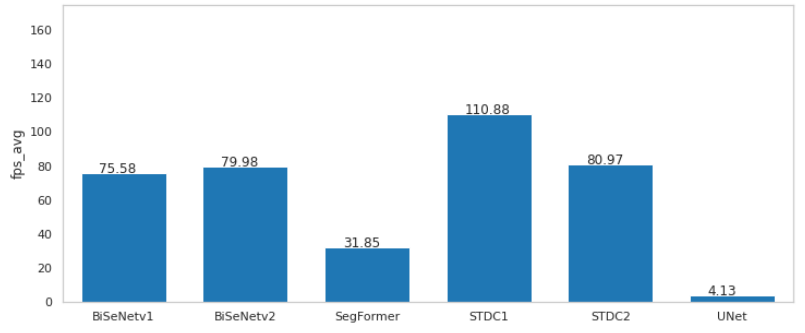
The results in Table 1 illustrate how **STDC** outperforms the rest of the models in terms of speed with almost 30 frames per second. Since **STDC1** is lighter than **STDC2** as the latter has more repeated blocks. So, this difference between **STDC1** and **STDC2** was expected. Meanwhile, **SegFormer-B1** as a transformer-based model is the slowest after **UNet** and this is due to the complexity of the hierarchical transformer encoder. **BiSeNet** results are reasonable due to its underlying ResNet18 complexity [44]. According to [52] comparison between ResNet18 and Xception39 [10] on CityScapes dataset [12], ResNet18 has 8.3 GFLOPs while Xception39 has 185.5 Million FLOPs for 3x640x360 input shape. **BiSeNet v2** semantic branch inspired by the Xception model is extremely efficient. Despite this, the Detail branch of this model uses a residual structure in addition to the layers' high channel capacity and its high-resolution output of 1/8 of the input image shape, which consumes more memory and leads to a longer execution time. [51].



(a)



(b)



(c)

Figure 9: Visualization graphs for the efficiency comparison. (a) Shows the Speed according to mIoU while the size of each point denotes the number of parameters. (b) Shows speed according to human IoU discarding UNet to provide better visualization. (c) Comparison between the models in terms of speed where we can visualize the real difference in speed between each one of them.

Complexity Analysis. In this experiment we are interested in visualizing how the complexity of each model affects the inference speed. The number of parameters for **STDC** compared to the rest of the models is astonishingly low. Observing only **BiSeNetv2**, this model has a significant amount of learnable parameters than **BiSeNet** and yet it has fewer FLOPs. Most of the **BiSeNetv2** Semantic Branch consists of Gather and Expansion Layers that increase the Receptive Field size. Hence the Depth-wise Convolution layers are followed by a low-capacity convolution layer. This way it is optimized in terms of computation and memory access cost reducing FLOPs while extracting semantic information. Since the ViT [13] backbone has quadratic complexity, **SegFormer-B1** transformers MiT lower the complexity of the process significantly to be just enough for real-time semantic segmentation.

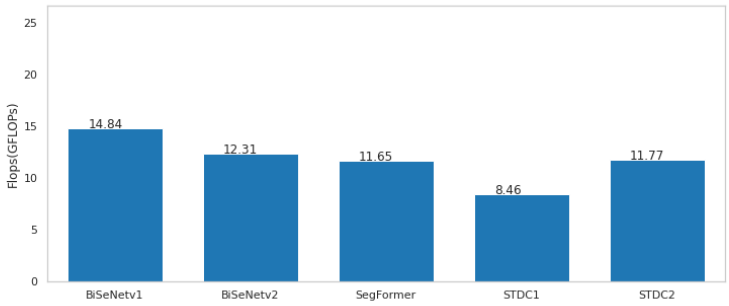
Method	FLOPs G	Params M	FPS
STDC1	8.46	8.57	110.88
SegFormer-b1	11.65	13.68	31.85
STDC2	11.77	12.60	80.97
BiSeNetv2	12.31	14.76	79.98
BiSeNet(ResNet18)	14.84	13.42	75.58
UNet	198.08	29.05	4.13

Table 2: Complexity comparison. The table is sorted in ascending order according to the FLOPs column. The FLOPs are in GIGAS and the numbers of parameters are in Millions. The speed measurement is included to add more visualization on how the speed of each model is affected by its complexity.

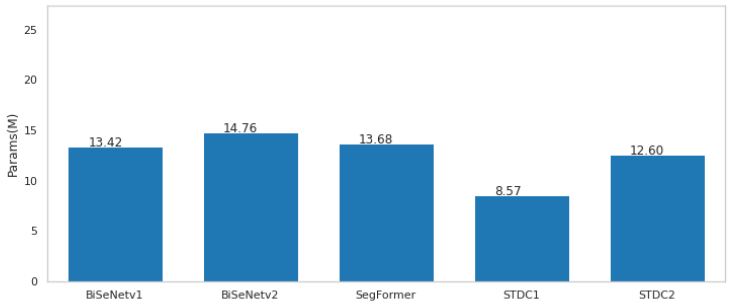
Accuracy Analysis. Like in any classification task, accuracy along with precision is what determines the quality of the model for a specific task. This is the measure that puts the model in any benchmark table.

Adding an extra path to extract spatial information is very useful to improve the performance on accuracy as **BiSeNet** has demonstrated. According to [18], using pretrained backbones borrowed from different classification tasks can harm the performance. Nevertheless, BiSeNetv2 adapts a booster training strategy and replaces the borrowed backbone with a robust lightweight model designed using depth-wise convolutions to enhance performance with less computation and memory access costs [51]. **STDC** demonstrates how this added extra path in **BiSeNet** architecture is time-consuming [18]. Furthermore, adds Detail Guidance of spatial information (low-level features) which can be discarded later in the inference phase which is very effective and induces more precise pixel classification. However, the spatial path can be added to enhance the performance, at the expense of higher computation costs.

SegFormer, in contrast, uses a lightweight decoder model that has only 0.4 M parameters. Meanwhile, the rest of the model is the hierarchical transformer encoder that outputs both high-level and low-level features providing a larger Effective Receptive Field than traditional CNN encoders can (as mentioned here [50]). For better accuracy **SegFormer-B5** is a better candidate while **SegFormer-B0** is the fastest **SegFormer** implementation. Nevertheless, we used **SegFormer-B1** for this test since we are looking for the fastest and most accurate model.



(a)



(b)

Figure 10: Visualization graphs for the complexity comparison. (a) Visualization of the graph of FLOPs on 512x512 input shape for all the selected models except UNet due to it elevated FLOPs when compared to the rest of the models. (b) Visualization of the graph positioning the selected models by their number of parameters.

Method	**human IoU**	**parameters**	**FPS**
UNet	0.56	29.05	4.13
BiSeNetv2	0.68	14.76	79.98
BiSeNet(ResNet18)	0.71	13.42	75.58
SegFormer-b1	0.73	13.68	31.85
STDC2	0.75	12.60	80.97
STDC1	0.75	8.57	110.88

Table 3: Comparison on Accuracy using our testing set for evaluation. This table is sorted in ascending order according to human category IoU. It shows that UNet has the least successful results, while STDC1 and STDC2 have the most successful results.

Comparison with models trained with CityScapes. After all, our final dataset had one class only. Hence, training these selected state-of-the-art approaches with this dataset considered them binary classifiers. As a result, the accuracy of the selected models in classifying human pixels only improved significantly. Meaning, that reducing the number of classes to one improves classification performance for this specific class. We evaluated the segmentation accuracy of the selected models on our dataset and CityScapes dataset [12]. **UNet** was removed as we are interested in evaluating only real-time semantic segmentation models. As illustrated in Figure 12, using our surveillance-like dataset generated better segmentation results for human category improving the performance on accuracy twice, almost for all the models.

Method	ours	CityScapes
BiSeNetv2	0.68	0.25
BiSeNet(ResNet18)	0.71	0.22
SegFormer-b1	0.73	0.29
STDC2	0.75	0.41
STDC1	0.75	0.43

Table 4: Human IoU for the selected models but UNet on both our dataset and CityScapes benchmark dataset.

5.4 Privacy-preserving results

The General Data Protection Regulation (GDPR) of the European Parliament and the Council of the European Union in respect of the protection of individuals' fundamental rights and freedom outlines the principles and rules on the protection of natural persons with regards to the processing of their data in the Union, regardless of whether the processing takes place in the Union or not.

Artificial Intelligence (AI) is a powerful tool that should be harnessed for the benefit of society and individuals. AI algorithms should be built with a responsibility to provide a transparent, accurate, and inclusive AI algorithm. A privacy-protected surveillance video visualization using DL approaches for real-time semantic segmentation to eliminate the risk of personal data being exposed in real-time. Specifically, the data that can be used to identify the person visible in a surveillance sequence, such as their physical characteristics, clothing, and colors of skin, hair, and eyes. An AI system that provides irreversible automated methods for safeguarding personal and sensitive data is in line with GDPR principles [35].

Integrity protection of a video-surveillance sequence is ensuring that each frame of this sequence is unalterable. As for Video Management Systems (VMS) or Video Surveillance systems, confidentiality can only be guaranteed by ensuring the inaccessibility of video data when it is captured by a camera device, stored in the VMS database, or when the video retention period ends. Therefore, the privacy of users is protected by giving limited information to the system operator and insiders while protecting sensitive data. Because the operator of the system can identify and visualize the monitored individuals. Moreover, video data can be shared and exposed in certain situations, as well as used as evidence in court.

Using a privacy safeguard technique, an additional object detection method is necessary to detect and identify regions of interest (ROI) in video data. You can choose to hide person-sensitive information (or ROI) or to hide the person's location.

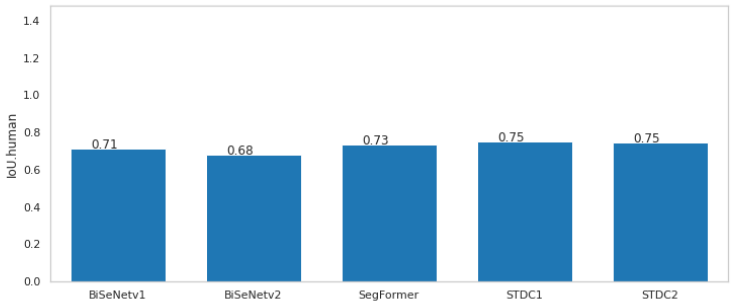


Figure 11: Visualization graph for the accuracy comparison. Shows STDC as the winner approach in terms of accuracy among the rest of the selected models.

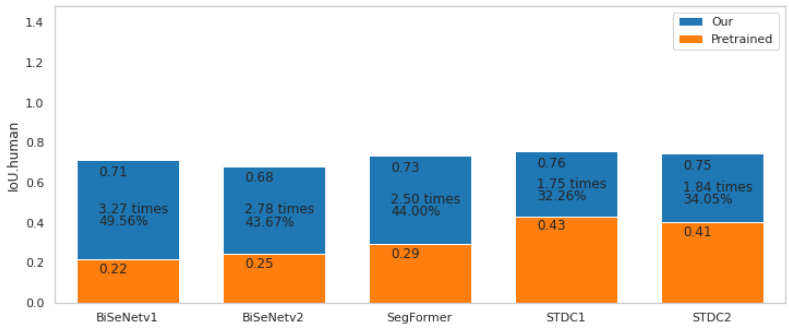


Figure 12: Visualization graph for the accuracy comparison. The difference between the segmentation accuracy for human category using our created dataset optimized for surveillance purposes and CityScapes dataset.

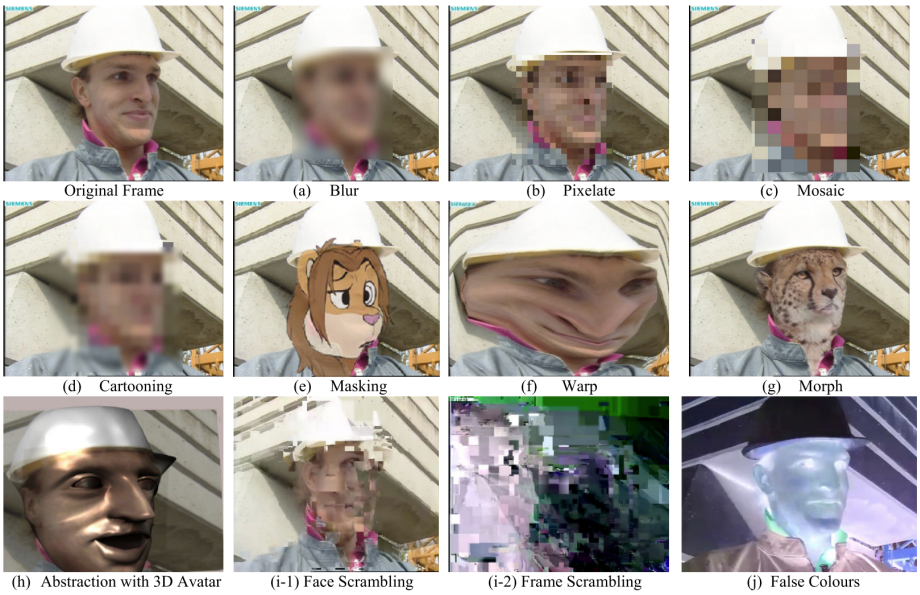


Figure 13: Obtained from [3] Existing data privacy safeguard techniques in image processing. (a) blurring: Correlation of a filter (Mean filter, weighted filter, Gaussian filter) over the region of interest (ROI) [19]. (b) Pixelation: Similar to Blurring but applied through pixel interpolation [20]. (c) Mosaic: Replacing ROI's pixels with a combination of Small blocks of pixels taken from different areas of the image [27]. (d) Cartooning: blurring superimposed on Sobel edge detection on the input image [16]. (e) Masking: Replaces ROI pixels with unknown pixels values taken from a different image. (f) Warp: The ROI's pixels are transformed geometrically to different plane based on a function to be later mapped to the image [47]. (g) Morph: Similar to warping but the transformation is from one image to another [47]. (h) Visual Abstraction: Replacing ROI with a cartoon 3D avatar [9]. (i) Scrambling: Scrumbles the AC coefficient of the blocks of pixels of the ROI by pseudorandomly inverting their signs [14]. (j) False Colors: Mapping different color palettes to the input image [1].

Figure 14: Privacy preservation in video surveillance using DL approaches for real-time semantic segmentation.

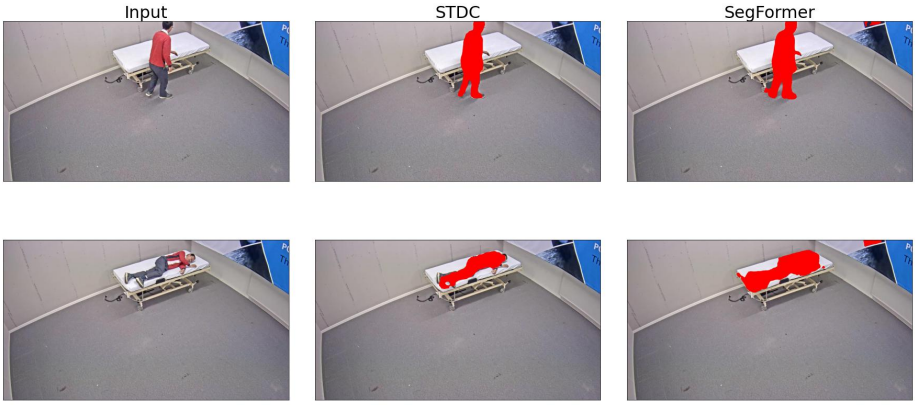


Figure 15: A comparison of STDC and SegFormer in providing privacy-protected video.

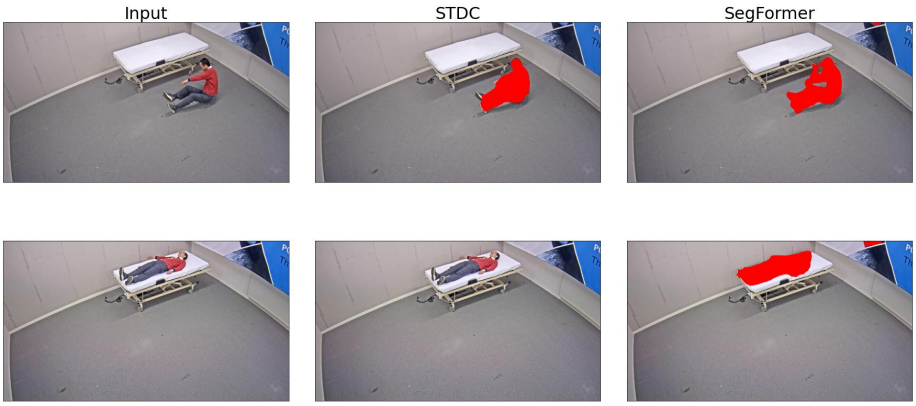


Figure 16: A comparison of STDC and SegFormer in providing privacy-protected video.

In figure 14. we can visualize an example of how privacy-protected video data is generated. In this case, all the person's sensitive information has been covered by the segmentation mask.

In the comparison between **STDC** and **SegFormer** in figure 15. **STDC** results are precise and accurate while **SegFormer's** shows the human is over-segmented. In addition, **SegFormer** output includes some pixels on the edges that are falsely detected as human pixels.

In figure 16. Shows **SegFormer** Falsely detecting the leg of the bed as part of the human body when they are relatively close but not overlapped. In the image in the second row, **STDC** miss-detects the human in the sequence especially when the human is completely straight on the bed. This miss-detection of sensitive data in this particular frame leads to a failure in providing a complete privacy-protected video surveillance sequence.

6 Discussion and future work

The dataset created for this study is not yet inclusive of all cases. We faced some limitations in obtaining accurate segmentation results testing on videos with illumination changes or the existence of reflective surfaces in the background like windows, mirrors, or glasses. The dataset we cleansed and processed in this study is not a complete dataset that we could consider an inclusive surveillance dataset. This dataset has only been used for this study to make a fair comparison to the selected models. We will investigate further in an extended study to find out more precisely the definition of surveillance-like images. We are looking into creating this brand-new dataset more precise and inclusive with higher resolution and better quality annotations. Furthermore, this dataset would open the possibility for future work, such as segmenting humans in the healthcare context in lying or sitting positions.

7 Conclusions

We have studied human image semantic segmentation for privacy-preserving in surveillance videos. Our contributions are listed as follows:

1. We have prepared and cleansed a combined dataset based on three open datasets, COCO_stuff164K, PASCAL VOC 2012, and ADE20K, focusing on surveillance-like human images.
2. We have explored, trained, and compared both transformer-based and FCN-based SoTA approaches for real-time semantic segmentation.
3. Our trained models with our dataset achieve significantly higher accuracy than off-the-shelf models pretrained on the CityScapes benchmark dataset.
4. A video from the open surveillance video dataset by Milestone Systems is used in testing our models. The video is not used for training or validation. Our trained models effectiveness is verified.
5. **STDC1** achieves the highest IoU and has the best balance in speed and accuracy (FPS =110, mIoU=86%). Its excellence for extracting spatial information (low-level features) and context details (high-level features) contributes to its success in high accuracy and efficiency. However, adding more layers to the **STDC** architecture does not result in high accuracy in our study.
6. We can provide real-time privacy preservation in video surveillance while still being able to observe the behavior of the monitored person.
7. We faced some limitations that may be resolved in future work such as; By segmenting humans in a video, privacy is safeguarded only and only when all sensitive regions in all the frames of a video sequence are detected by the DL model. Relevant studies include alternative masking approaches and a more reliable segmentation pipeline, such as context and hardware adaptive segmentation; We were limited in terms of data such as training the models with poor quality annotations. Additionally, the lack of training data with illumination changes and including segmentation of reflections of humans on glasses or mirrors could enhance the segmentation results using our dataset.

Acknowledgment:
This study was supported by Milestone Systems. We thank Jorge Fandos Abadia for providing the hardware and support for this study. And thanks to Zenjie Li for the support and insightful discussions.

Bibliography

[1] Serdar ÃiftÃ§i, Ahmet OÄuz AkyÃ¼z, and Touradj Ebrahimi. "A Reliable and Reversible Image Privacy Protection Based on False Colors". In: *IEEE Transactions on Multimedia* 20.1 (2018), pp. 68–81. DOI: 10.1109/TMM.2017.2728479.

[2] Pablo Arbelaez et al. "Contour Detection and Hierarchical Image Segmentation". In: *IEEE Trans. Pattern Anal. Mach. Intell.* 33.5 (May 2011), pp. 898–916. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2010.161. URL: http://dx.doi.org/10.1109/TPAMI.2010.161.

[3] Mamoona N. Asghar et al. "Visual Surveillance Within the EU General Data Protection Regulation: A Technology Perspective". In: *IEEE Access* 7 (2019), pp. 111709–111726. DOI: 10.1109/ACCESS.2019.2934226.

[4] P. T. V. Bhuvaneswari and A. Brintha Therese. "Edge Detection Techniques in Digital and Optical Image Processing". In: 2014.

[5] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. "COCO-Stuff: Thing and stuff classes in context". In: *Computer vision and pattern recognition (CVPR), 2018 IEEE conference on*. IEEE. 2018.

[6] John Canny. "A Computational Approach to Edge Detection". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-8.6 (1986), pp. 679–698. DOI: 10.1109/TPAMI.1986.4767851.

[7] Liang-Chieh Chen et al. "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40 (2018), pp. 834–848.

[8] Xianjie Chen et al. "Detect What You Can: Detecting and Representing Objects Using Holistic Models and Body Parts". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition* (2014), pp. 1979–1986.

[9] Kenta Chinomi et al. "PriSurv: Privacy Protected Video Surveillance System Using Adaptive Visual Abstraction". In: *Advances in Multimedia Modeling*. Ed. by Shin'ichi Satoh, Frank Nack, and Minoru Etoh. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 144–154. ISBN: 978-3-540-77409-9.

[10] FranÃ§ois Chollet. "Xception: Deep Learning with Depthwise Separable Convolutions". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), pp. 1800–1807.

[11] MMSegmentation Contributors. *MMSegmentation: OpenMMLab Semantic Segmentation Toolbox and Benchmark*. 2020.

[12] Marius Cordts et al. "The Cityscapes Dataset for Semantic Urban Scene Understanding". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 3213–3223.

[13] Alexey Dosovitskiy et al. "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *ArXiv* abs/2010.11929 (2021).

[14] Frederic Dufaux and Touradj Ebrahimi. "Ebrahimi, T.: Scrambling for privacy protection in video surveillance systems. IEEE Trans. Circuits Syst. Video Technol. 18(8), 1168-1174". In: *Circuits and Systems for Video Technology, IEEE Transactions on* 18 (Sept. 2008), pp. 1168–1174. DOI: 10.1109/TCSVT.2008.928225.

[15] Vincent Dumoulin and Francesco Visin. "A guide to convolution arithmetic for deep learning". In: *ArXiv* abs/1603.07285 (2016).

[16] ÃdÃ¡m ErdÃ©lyi et al. “Adaptive cartooning for privacy protection in camera networks”. In: *2014 11th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. 2014, pp. 44–49. DOI: 10.1109/AVSS.2014.6918642.

[17] M. Everingham et al. *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html.

[18] Mingyuan Fan et al. “Rethinking BiSeNet For Real-time Semantic Segmentation”. In: *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2021), pp. 9711–9720.

[19] Jan Flusser et al. “Recognition of Images Degraded by Gaussian Blur”. In: *CAIP*. 2015.

[20] Timothy Gerstner et al. “Pixelated image abstraction with integrated user constraints”. In: *Comput. Graph.* 37 (2013), pp. 333–347.

[21] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. http://www.deeplearningbook.org. MIT Press, 2016.

[22] Stephen Gould, Richard Fulton, and Daphne Koller. “Decomposing a Scene into Geometric and Semantically Consistent Regions”. In: *Proceeding of International Conference on Computer Vision (ICCV)*. 2009.

[23] Murilo Gustineli. “A survey on recently proposed activation functions for Deep Learning”. In: *ArXiv* abs/2204.02921 (2022).

[24] Geremy Heitz et al. “Cascaded Classification Models: Combining Models for Holistic Scene Understanding”. In: *22nd Neural Information Processing Systems (NIPS)*. 2008.

[25] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: *CoRR* abs/1412.6980 (2015).

[26] Alexander Kirillov et al. “Panoptic Segmentation”. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2019), pp. 9396–9405.

[27] Yuichi Kusama, Hyunho Kang, and Keiichi Iwamura. “Mosaic-based privacy-protection with reversible watermarking”. English. In: *SIGMAP 2015 - 12th International Conference on Signal Processing and Multimedia Applications, Proceedings; Part of 12th International Joint Conference on e-Business and Telecommunications, ICETE 2015*. Ed. by Mohammad S. Obaidat, Pascal Lorenz, and Enrique Cabello. SIGMAP 2015 - 12th International Conference on Signal Processing and Multimedia Applications, Proceedings; Part of 12th International Joint Conference on e-Business and Telecommunications, ICETE 2015. 12th International Conference on Signal Processing and Multimedia Applications, SIGMAP 2015 ; Conference date: 20-07-2015 Through 22-07-2015. SciTePress, Jan. 2015, pp. 98–103.

[28] Ping-Sung Liao, Tse-Sheng Chen, and P. C. Chung. “A Fast Algorithm for Multilevel Thresholding”. In: *J. Inf. Sci. Eng.* 17 (2001), pp. 713–727.

[29] Beyang Liu, Stephen Gould, and Daphne Koller. “Single image depth estimation from predicted semantic labels”. In: *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition* (2010), pp. 1253–1260.

[30] Wei Liu, Andrew Rabinovich, and Alexander C. Berg. “ParseNet: Looking Wider to See Better”. In: *ArXiv* abs/1506.04579 (2015).

[31] *Microsoft Research - Image Understanding*. URL: https://www.microsoft.com/en-us/research/project/image-understanding/.

[32] M. Minsky and S.A. Papert. “Perceptrons: An Introduction to Computational Geometry.” In: (1969).

[33] Purnendu Mishra and Kishor Sarawadekar. “Polynomial Learning Rate Policy with Warm Restart for Deep Neural Network”. In: *TENCON 2019 - 2019 IEEE Region 10 Conference (TENCON)*. 2019, pp. 2087–2092. DOI: 10.1109/TENCON.2019.8929465.

[34] Thomas Brox Olaf Ronneberger Philipp Fischer. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: (2015). DOI: http://dx.doi.org/10.1002/andp.19053221004.

[35] THE EUROPEAN PARLIAMENT and THE COUNCIL OF THE EUROPEAN UNION. *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)*. 2016.

[36] Ning Qian. "On the momentum term in gradient descent learning algorithms". In: *Neural networks : the official journal of the International Neural Network Society* 12 1 (1999), pp. 145–151.

[37] Muthukrishnan R. "Edge Detection Techniques For Image Segmentation". In: *International journal of computer science and information technology* 3 (Dec. 2011), pp. 259–267. DOI: 10.5121/ijcsit.

[38] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: *ArXiv* abs/1609.04747 (2016).

[39] Ashutosh Saxena, Sung Chung, and Andrew Ng. "Learning Depth from Single Monocular Images". In: *Advances in Neural Information Processing Systems*. Ed. by Y. Weiss, B. Schölkopf, and J. Platt. Vol. 18. MIT Press, 2005. URL: https://proceedings.neurips.cc/paper/2005/file/17d8da815fa21c57af9829fb0a869602-Paper.pdf.

[40] Ashutosh Saxena, Min Sun, and Andrew Y Ng. "Make3D: learning 3D scene structure from a single still image". In: *IEEE transactions on pattern analysis and machine intelligence* 31.5 (May 2009), 824â840. ISSN: 0162-8828. DOI: 10.1109/tpami.2008.132. URL: https://doi.org/10.1109/TPAMI.2008.132.

[41] Evan Shelhamer, Jonathan Long, and Trevor Darrell. "Fully Convolutional Networks for Semantic Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39 (2017), pp. 640–651.

[42] Taghi M. Shorten Connor. Khoshgoftaar. "A survey on Image Data Augmentation for Deep Learning". In: *Journal of Big Data* (2019). DOI: 10.1186/s40537-019-0197-0.

[43] T. V. Srikanth, Proff. Pradeep Kumar, and Ashwin Kumar. "Color Image Segmentation using Watershed Algorithm". In: 2011.

[44] Sasha Targ, Diogo Almeida, and Kevin Lyman. "Resnet in Resnet: Generalizing Residual Architectures". In: *ArXiv* abs/1603.08029 (2016).

[45] S. Thylashri, Udutha Yadav, and T. Chowdary. "Image Segmentation Using K- Means Clustering Method for Brain Tumour Detection". In: *International Journal of Engineering and Technology(UAE)* 7 (Apr. 2018), pp. 97–100. DOI: 10.14419/ijet.v7i2.19.15058.

[46] Ashish Vaswani et al. "Attention is All you Need". In: *ArXiv* abs/1706.03762 (2017).

[47] Luiz Velho, Alejandro Frery, and Jonas Gomes. "Warping and Morphing". In: *Image Processing for Computer Graphics and Vision*. London: Springer London, 2009, pp. 387–412. ISBN: 978-1-84800-193-0. DOI: 10.1007/978-1-84800-193-0_15. URL: https://doi.org/10.1007/978-1-84800-193-0_15.

[48] *Visual Object Classes Challenge 2012*. 2017. URL: https://groups.csail.mit.edu/vision/datasets/ADE20K/.

[49] Wenhai Wang et al. "Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions". In: *2021 IEEE/CVF International Conference on Computer Vision (ICCV)* (2021), pp. 548–558.

[50] Enze Xie et al. "SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers". In: *NeurIPS*. 2021.

[51] Changqian Yu et al. "BiSeNet V2: Bilateral Network with Guided Aggregation for Real-time Semantic Segmentation". In: *Int. J. Comput. Vis.* 129 (2021), pp. 3051–3068.

[52] Changqian Yu et al. "BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation". In: *Proceedings of the European Conference on Computer Vision (ECCV)*. Sept. 2018.

[53] Bolei Zhou et al. “Scene Parsing through ADE20K Dataset”. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 5122–5130. DOI: 10.1109/CVPR.2017.544.

[54] Bolei Zhou et al. “Semantic Understanding of Scenes Through the ADE20K Dataset”. In: *International Journal of Computer Vision* 127 (2018), pp. 302–321.