Enhancing Privacy: Automated Detection and Blurring of Sensitive Information in Images and Video Feeds

Yanis Najy Miraoui
Stanford University
ymiraoui@stanford.edu

Paul Woringer
Stanford University
paulworinger@stanford.edu

Abstract

The monitoring of urban spaces has become a critical aspect of modern city management. However, these surveillance systems, which utilize extensive networks of cameras and sensors, pose significant privacy risks by capturing sensitive personal information, such as license plates and facial features. Our project addresses these privacy concerns by developing an innovative solution using advanced Computer Vision techniques. We focus on real-time detection and anonymization of sensitive information in urban surveillance feeds, specifically targeting license plates and human faces. The input to our system consists of high-definition images and video feeds from urban environments, processed using deep learning models to detect and blur sensitive data. We employ state-of-the-art models for Optical Character Recognition (OCR) and image segmentation, integrated into a robust pipeline to ensure privacy protection without compromising the utility of surveillance data. Our solution demonstrates high performance and efficiency, providing a viable approach to enhancing privacy in modern urban surveillance systems. Through extensive evaluation, we show that our system effectively safeguards privacy while maintaining the footage's intended purpose, addressing the complex challenges posed by dynamic urban environments.

1. Introduction

In an increasingly digital and connected world, the monitoring of urban spaces has become a critical aspect of modern city management. Surveillance systems are used for various purposes, including traffic management, public safety, and urban planning. These systems rely on extensive networks of cameras and sensors that continuously capture and process vast amounts of visual data. Although these technologies offer significant benefits in terms of efficiency and security, they also pose substantial privacy risks.

The inadvertent capture and dissemination of sensitive personal information, such as license plates, facial features, and other identifiable details, are common in monitored environments. This raises serious privacy concerns, as unauthorized access to such information can lead to misuse, identity theft, and other forms of harm. Moreover, the constant surveillance may infringe on individuals' rights to privacy, creating a sense of being perpetually watched.

However, addressing these privacy concerns is not straightforward due to the complex and dynamic nature of urban environments. Moving objects, varying lighting conditions (day and night), and diverse backgrounds make it challenging for conventional static image processing tools to effectively detect and mask sensitive information. Therefore, there is a need for the use of advanced Computer Vision techniques that can operate efficiently and accurately in real-time to ensure privacy.

Our project aims to develop an innovative solution that leverages state-of-the-art ML in optical character recognition (OCR), image segmentation, and deep learning to enhance privacy protection in urban surveillance. Our primary focus will be on detecting and anonymizing two critical types of sensitive information: license plates and human faces. By integrating these advanced techniques, we seek to create a robust system capable of detecting and anonymizing personal data in real-time video feeds and images. Our approach will not only safeguard privacy, but will also maintain the utility of surveillance for its intended purposes.

1.1. Problem Statement

The inadvertent capture and dissemination of personal data in monitored urban spaces violate privacy norms and expose individuals to potential misuse of their information. This is especially critical for sensitive data such as license plates and facial features. Conventional static image processing tools are inefficient in addressing these complex dynamics, creating the need for advanced Computer Vision techniques to detect and mask personal data in real-time video feeds and images.

1.2. Setup

The input to our system comprises high-definition images and video feeds collected from various urban environments, including busy streets and public parks, captured under diverse lighting conditions and weather scenarios. These images and videos are pre-processed and annotated to mark sensitive information such as license plates and human faces. Using advanced Computer Vision techniques, particularly object detection and Optical Character Recognition (OCR) for license plates and image segmentation for human faces, we train deep learning models, to detect and anonymize this sensitive information in real-time. The output of our system is an image or a video feed in which all detected license plates and faces are effectively anonymized and blurred, thus protecting individuals' privacy while maintaining the utility of the surveillance footage for its intended purposes.

2. Related Work

2.1. Face and object detection

Haar Cascades One of the first breakthroughs in object detection was brought about by Viola and Jones (2001) [20], who combined increasingly complex classifier models to discard background zones efficiently and only dedicate resources to zones that have been deemed likely to contain said objects. These models use Haar features to classify objects. This increase in efficiency allowed for face detection to be performed in real time. We will be using this pretrained model as our first baseline for face detection. This was a great step forward at the time of its publication, but its performance does not hold up well against more recent advances.

Deep Learning and Convolutional Neural Networks (CNN) Haar cascades, as efficient as they were, relied on predefined Haar features that limited the expressivity of the models. More recently, the boom of deep learning has led to the development of convolutional neural networks, which automatically learn hierarchical feature representations from the data. They are also more robust to variations in lighting, pose, and occlusions. This allowed for significant improvements in performance, as displayed with Girshick et al. (2013) [4], Ren et al. (2016) [18], and Redmon et al. (2016) [17] (that we have studied in class). We will be using the YOLO model introduced by Redmon et al. in our pipeline to detect vehicles (see below). Approaches like Zhang et al. [23] leverage multitask training for increased performance in face detection and alignment. Transfer learning is also often used to take advantage of large pretrained models and applying them to more specific problems. This is an approach that will be central in our project.

Recent advances Recently, the focus seems to have shifted from CNNs to models with attention mechanisms and based on transformers, as introduced by Dosovitskiy et al. (2021) [2], as they have brought considerable increases in performance. Increasing attention is also given to lightweight models, which allow for increased portability to devices with limited resources (such as smartphones), as presented by Howard et al. (2017) [6]. There is also research being done in privacy-preserving face-detection in order to comply to regulations and ethics, using federated learning and differential privacy to ensure the privacy of the people displayed in the training set. While these techniques are all very promising and relevant to our task, as of now we are not considering including them in our project due to our limited time.

2.2. OCR

While there has been considerable focus on Optical Character Recognition long before face detection was even thought to be possible, recent trends in this task have followed closely the advancements made for object detection. However, specific research is also dedicated to this task, with a focus on robustness and multilingual systems. Some very powerful implementations such as Tesseract [11] and EasyOCR [8] are readily available. The Tesseract model has an architecture based on CNNs followed by a deep bidirectional LSTM. EasyOCR on the other hand uses a ResNet for feature extraction (also followed by an LSTM layer). Both models also include a postprocessing layer.

2.3. Applications to Privacy

The methods described above have been collated and applied to the task of blurring out sensitive information to preserve individual privacy. For example, the paper Frome et al. (2009) [3] which describes the initial process used in Google Street View was the inspiration for our project. The main objective of this paper is to maximize its recall on face and license-plate detection, in order to avoid leaking private information. To do so, they use a two-step process with a sliding-window detector tuned for high recall, followed by a fast post-processing stage that includes domain-specific information to mitigate the number of false positives given by the first detector. One limitation of this paper is that it is rather domain-specific, in that the paper only uses data from Google Street View, which is usually pretty controlled (for example images are only taken during the day in relatively clear weather).

3. Methods

Our objective is to accurately detect sensitive information, with a primary focus on identifying faces and license plates for the purpose of blurring them. We will address both static images and dynamic content such as

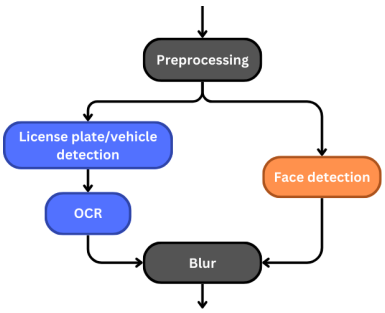

Figure 1. Our proposed pipeline to process an input image

video surveillance footage. To achieve this, we have designed a multi-component pipeline that includes a license plate/vehicle detection model, an OCR model, and a face detection model. This fully autonomous pipeline will be capable of processing inputs, such as street images and video clips, and producing outputs of blurred and desensitized images and video clips. Note that this pipeline architecture and modules is ultimately what we aim to achieve.

3.1. Pipeline Components

The pipeline, described on Figure 1, is structured into several key components, each with specific responsibilities:

1. License Plate/Vehicle Detection Model: This model uses pre-trained YOLO-V8 fine-tuned for our task to identify and locate vehicles and license plates in the input data.

2. OCR Model: The OCR model ensures the accurate identification of license plates by reading and recognizing alphanumeric characters, distinguishing them from other textual elements such as highway signs. This verification step allows precise anonymization and blurring.

3. Face Detection Model: This component utilizes deep learning techniques, specifically Convolutional Neural Networks (CNNs), to detect and locate human faces within images and video frames. We will consider models like Haar Cascades but also fine-tuned pre-trained model and compare their accuracy as well as efficiency for our task.

4. Blurring Module: After detecting sensitive information, the blurring module processes these areas to obscure identifiable features using methods such as Gaussian blurring, pixelation, or synthetic overlays, ensuring privacy without compromising the footage's overall utility.

3.2. Baselines

Each component of our pipeline are evaluated against a baseline model. These baselines include:

- For detecting vehicles: Haar Cascades and a pre-trained YOLO-V8 model.
- For detecting persons and faces: Haar Cascades and MTCNN.
- For Optical Character Recognition (OCR): EasyOCR and Tesseract OCR.

These baselines have been established in past work and are described in details in the Literature Review section.

In the following subsections, we will provide detailed explanations of each component of our pipeline and their functionalities.

3.3. License plate and vehicle detection

As described earlier, our first module is composed of a license plate and vehicle detection model. This component takes as input an image or video and returns the location and bounding box of the license plate or vehicle if one those is present in the image. For this purpose, we trained and evaluated a fine-tuned YOLO-V8 model. We compared its performance to other baseline models such as Haar Cascades and a pre-trained only YOLO-V8 model. In fact, the YOLO-V8 model operates by dividing the input image into a grid and predicting bounding boxes and class probabilities for each grid cell, allowing it to accurately detect and localize vehicles and license plates in real-time. Mathematically, the YOLO model optimizes the following loss function:

$$\mathcal{L} = \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$
$$+ \lambda_{\text{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2 \right]$$
$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(C_i - \hat{C}_i)^2 \right]$$
$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left[(C_i - \hat{C}_i)^2 \right]$$

Where:

- S is the number of grid cells.
- B is the number of bounding boxes per grid cell.
- $\mathbb{1}_{ij}^{\text{obj}}$ indicates if the j-th bounding box in the i-th grid cell is responsible for the object.
- x_i, y_i, w_i, h_i are the coordinates and dimensions of the bounding box.
- C_i is the confidence score for the bounding box.

- λ_{coord} and λ_{noobj} are hyperparameters to balance the loss terms.

The loss function ensures that the predicted bounding box coordinates and dimensions match ground truth values (localization loss), penalizes incorrect confidence scores (confidence loss), and includes a classification loss component to penalize incorrect class predictions, ensuring accurate object classification. Hyperparameters λ_{coord} and λ_{noobj} balance the importance of localization accuracy and background detection. Overall, By optimizing this composite loss function, YOLO effectively learns to predict accurate bounding boxes, high confidence scores for objects, and correct object classifications. All of these properties are particularly valuable for our task.

3.4. Optical Character Recognition (OCR)

To verify and read license plates, we employ OCR models such as EasyOCR and Tesseract OCR. These models are crucial for distinguishing license plates from other textual elements like highway signs. EasyOCR and Tesseract OCR use Convolutional Neural Networks to extract features from the input image and a sequence-to-sequence model for character recognition. The performance of OCR models is evaluated using metrics such as accuracy, precision, and recall (as described in Section 3.7).

3.5. Face Detection

The face detection component leverages advanced deep learning techniques, particularly CNNs. We compare various models, including Haar Cascades, MTCNN, and DeepFace, to identify the most performant model for our task. These models mainly work by detecting facial landmarks and using these landmarks to predict the bounding box of the face. Mathematically, the face detection models generally optimize the following loss functions:

1. A localization loss using the Mean Squared Error (MSE):

$$\mathcal{L}_{\text{MSE}} = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

   where y_i is the ground truth coordinate and $\hat{y}_i$ is the predicted coordinate for the i-th landmark.

2. A classification loss using the Cross-Entropy loss:

$$\mathcal{L}_{\text{CE}} = -\sum_{i=1}^{N} y_i \log(\hat{y}_i)$$

   where y_i is the ground truth label and $\hat{y}_i$ is the predicted probability for the i-th class.

These loss functions help ensure that the detected facial landmarks and bounding boxes closely match the ground truth values. By focusing on accurate localization of facial landmarks and minimizing classification errors, these face detection models can reliably identify human faces in various lighting and pose conditions. All of these properties are particularly valuable for our task, making these models essential for accurate and reliable face detection in images and videos.

3.6. Anonymizing and Blurring Sensitive Information

Once sensitive information such as faces and license plates are detected, the anonymizing and blurring module processes these regions to ensure privacy. Methods such as Gaussian blurring and pixelation are applied to obscure identifiable features without compromising the overall utility of the footage. Gaussian blurring uses a Gaussian function to smooth the image, defined as:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}$$

The pixelation method, on the other hand, involves reducing the resolution of the detected regions by averaging the pixel values within a block. This can be represented as:

$$P_{ij} = \frac{1}{n^2}\sum_{m=0}^{n-1}\sum_{n=0}^{n-1} I_{(i+m)(j+n)}$$

where P_{ij} is the pixel value in the pixelated image and $I_{(i+m)(j+n)}$ are the pixel values in the original image within the $n \times n$ block.

These anonymization techniques ensure that the sensitive regions are obscured efficiently while maintaining the integrity of the surrounding image.

3.7. Evaluation and Metrics

We evaluate our pipeline in 2 steps:

1. The first step is to evaluate each component and model of our pipeline separately only based on the specific task they aim to achieve. For instance, we evaluate the performance of the License Plate/Vehicle Detection model alone on its own for its specific task first: detecting when there are and where are the vehicles.

2. The second step consists of testing and evaluating our whole pipeline as a whole. This involves a more qualitative analysis of our results and allows us to conclude on the overall performance of our system.

For comparing the models to the baselines for each component, we used multiple different metrics:

- Accuracy $= \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}}$

Task	Model	Accuracy	Precision	Recall
License Plate Detection	Haar Cascades	0.62	0.82	0.58
	YOLO-V8	0.91	0.91	0.90
	YOLO-V8 (fine-tuned)	*0.94*	*0.95*	*0.94*
Vehicle Detection	Haar Cascades	0.56	0.85	0.62
	YOLO-V8	0.84	0.88	0.75
	YOLO-V8 (fine-tuned)	*0.86*	*0.90*	*0.87*
Person and Face Detection	Haar Cascades	0.44	0.62	0.58
	MTCNN	0.82	0.80	0.79
	DeepFace	*0.84*	*0.81*	*0.80*
OCR	Tesseract OCR	0.76	0.58	0.82
	EasyOCR	*0.81*	*0.64*	*0.85*

Table 1. Performance metrics of baseline models and our method *in italic* for each task.

- Precision $= \frac{\text{True Positives}}{\text{True Positives}+\text{False Positives}}$
- Recall $= \frac{\text{True Positives}}{\text{True Positives}+\text{False Negatives}}$
- F1 Score $= 2 \times \frac{\text{Precision}\times\text{Recall}}{\text{Precision}+\text{Recall}}$
- Mean Average Precision (mAP) $= \sum_{k=1}^{M}(R_k - R_{k-1})P_k$ where M is the number of recall levels, R_k is the recall at the k-th threshold, and P_k is the precision at the k-th threshold.

These metrics are computed using the Intersection over Union (IoU) between the ground truth bounding boxes and the ones created by our different models. In fact, IoU measures the overlap between the predicted bounding box and the ground truth bounding box. Using these metrics allowed us to have a better understanding of where our proposed methods were successful or failing compared to the baselines.

We evaluated our pipeline comprehensively using these metrics, and also conducted a qualitative analysis of our results. This approach is justified by the complexity of our pipeline, which is composed of multiple different components that contribute to a single output. As a result, evaluating each component quantitatively alone might not fully capture the effectiveness and nuances of the overall system.

4. Datasets

For our analysis, we utilized and compiled multiple datasets. Again, this was necessary to evaluate both the individual components of our pipeline and the pipeline as a whole.

We have used the COCO dataset [14] for face and car detection (not license-plates directly). It contains 117266 training images and 4952 validation images. They have varying resolutions (480x640, 335x500, 640x480, 335x500, etc.). Please refer to Figure 7 in the appendix for examples.

We have used a license-plate specific dataset available on HuggingFace [1] for the license-plate detection task. Some examples are displayed in the appendix in Figure 8. Note that this is a dataset where the license plates are most of the time relatively well displayed and not too obstructed. It contains images from different settings: security camera above the entrance of a parking garage, outside parking lot, cars and motorcycles. Note also that some images may contain text outside of the license plate, as shown on one of the example images, which might lead to false positives for OCR-based approaches to license-plate detection. It is already split in a three train/validation/test datasets of 6176, 1765, and 882 entries respectively. The majority of the images have dimensions 472x303, but some larger resolutions also appear (1024x768, 1024x608, 764x428, etc.).

We have used the TImes Square Intersection (TISI) video surveillance dataset [2]. Note that this dataset contains videos. In our tests, to make training more efficient and the visualization of our results simpler, we chose to sample these videos at regular intervals. The dataset overall contains 1465 folders, each containing 1000 frames. The videos are taken at approximately 10Hz. We sample one image every 50 frames (5 seconds between each image). This gives the pedestrians and the cars in the images enough time to move, thus generating diverse examples, while still keeping a large number of data points (29,300 total). The resolution of each of these images is 550x960. Please refer to Figure 9 in the appendix for examples.

[1] https://huggingface.co/datasets/keremberke/license-plate-object-detection

[2] https://xiatian-zhu.github.io/downloads_qmul_TISI_dataset.html

5. Results and Discussion

We have built and evaluated each of the models and methods described in earlier sections. We have compiled the results and metrics obtained on a separate test set in Table 1.

5.1. License plates and vehicles detection

First of all, we notice that as expected, the fine-tuned YOLO-V8 model performs best for license plate and vehicle detection compared to our baselines. It achieves very high performance for our task. Comparing our method to the baseline, we notice that Haar Cascades performs poorly in terms of accuracy and recall. Note that we have used Haar Cascades with a scale factor equal to 1.1 and a minNeighbors of 5. These hyperparameters were the highest performing ones on a separate validation set. Additionally, we can see on Figure 2 that Haar Cascades performs only relatively well qualitatively: it is not able to accurately detect a car and the bounding boxes are much larger than the size of the vehicle.

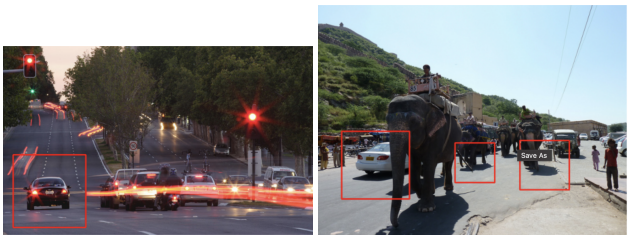

Figure 2. Outputs from Haar Cascades and examples of common unsuccessful predictions.

5.2. Optical Character Recognition (OCR)

We obtained very promising results for the two OCR models that we have considered in our analysis. Both Tesseract OCR (with Legacy Engine and Page segmentation mode 6 which assumes a uniform block of text) and EasyOCR hold very high accuracy and recall scores. However, they obtained low precision scores because they detected a lot of false positives. This should not be an issue in our case for our pipeline as the OCR will only be applied after, we have detected and determined a bounding box around the vehicles and license plates present in the image or video. Some examples of correct license plates detection as well as OCR detection are presented on Figure 3.

5.3. Face detection

Regarding person and face detection, our experiments reveal that the DeepFace model significantly outperforms the baseline models, especially in scenarios involving multiple faces in a single frame. DeepFace demonstrates robust accuracy and recall rates, ensuring reliable detection even in complex scenes. On the other hand, Haar Cascades show noticeable limitations in detecting multiple faces simultaneously. This inadequacy is illustrated in Figure 4, where Haar Cascades often miss several faces or produce imprecise bounding boxes around detected faces. In our evaluations, we used Haar Cascades with a scale factor of 1.2 and a minNeighbors value of 3, which were the optimal settings determined from our validation set. Despite these tuned hyperparameters, Haar Cascades frequently underperformed, highlighting the superiority of DeepFace in face detection tasks.

Figure 3. Output from YOLO-V8 (left) and EasyOCR (right) on the same image for license plate detection. Note that EasyOCR also flags the timestamp of the image.

Figure 4. Face detection examples using Haar Cascades and a common failure faced. On the left, a face is correctly detected, while on the right, there is a false positive and a false negative. In the false positive, an incorrect bounding box is constructed in the background, and in the false negative, the main face is not correctly detected.

5.4. End-to-end pipeline

Before, putting all the components together, we implemented two blurring methods as described earlier in Section 3.6: Gaussian blurring and Pixelation. We tested it on multiple license plates and faces and an example of its results can be found on Figure 5.

Finally, our last step was to put all the different components and modules together in order to build and evaluate our end-to-end pipeline. As it is very hard, in our case, to obtain quantitative metrics of how well our end-to-end pipeline performs as a whole, we performed a more quantitative analysis of our results. In fact, as observed on Figure 6, our pipeline performs well. Qualitatively, it is able to

successfully detect and blur both the license plates as well as the faces present on the image.

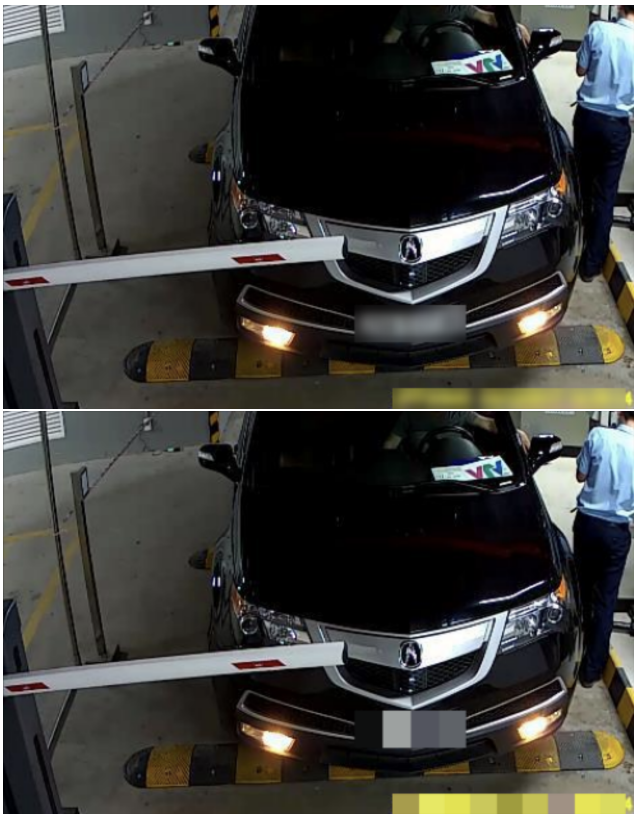

Figure 5. Output of our pipeline using Gaussian blur (top) or blur by pixelation (bottom). Note that we also blurred the text from the timestamp as it was also detected by EasyOCR.

Figure 6. Example of final outputs obtained from our end-to-end pipeline: all of the license plates and identifiable faces have been successfully blurred.

6. Conclusion and Next Steps

In this project, we addressed the critical issue of privacy protection in urban surveillance systems by developing an innovative and complete end-to-end anonymization tool. Our comprehensive solution leverages advanced Computer Vision techniques and models in order to detect and anonymize sensitive information, specifically license plates and human faces, in real-time video feeds and images. Our innovative multi-component pipeline integrates state-of-the-art models for license plate/vehicle detection, Optical Character Recognition (OCR), and face detection, demonstrating high performance in safeguarding privacy while maintaining the utility of surveillance footage.

Our experimental results indicate that fine-tuned YOLO-V8 models for license plate and vehicle detection, combined with DeepFace for face detection, provide robust and accurate identification of sensitive information. The integration of Gaussian blurring and pixelation techniques ensured effective anonymization, further enhancing the privacy-preserving capabilities of our system.

The proposed pipeline successfully addressed the challenges posed by dynamic urban environments, varying lighting conditions, and diverse backgrounds. By optimizing for both accuracy and efficiency, our solution offers a viable approach to privacy protection in modern urban surveillance systems, potentially mitigating the privacy risks associated with extensive visual data capture and processing.

While our project has made significant strides in enhancing privacy protection in urban surveillance, several areas for future work remain. Future work could involve further testing and validation in real-world environments, incorporating user-controlled privacy settings into the system in order for users to specify their privacy preferences or extending our pipeline to take into account the temporal aspect of dynamic video feeds.

7. Contributions and Acknowledgements

This project was solely carried for this course and was not combined with any other course. We wrote and implemented all the code ourselves and did not use any public GitHub repository. Regarding the distribution of work, Paul focused on the OCR, the blurring methods and the evaluation of our end-to-end pipeline while Yanis focused on the license plates, vehicles and faces detection models. The write-up and the figures were collaboratively created and edited by both of us to ensure consistency and clarity throughout the project report.

8. Appendices

8.1. Examples of images from our dataset

Figure 7. Example images from the COCO dataset

8.2. Packages Used

Note that we only cite the packages directly imported into our project, not their dependencies.

Data preprocessing

- Numpy [5]
- Pandas [21]
- Pytorch [15]
- Pillow [19]
- Transformers pipeline [22]

Figure 8. Example images from the license plate dataset from Huggingface

Figure 9. Example images from the Times Square Intersection dataset

Datasets

- Huggingface datasets [13]

Models and evaluation

- Segment Anything [12]
- YOLO-V5 [9]
- YOLO-V8 [10]
- Open CV (cv2) [1]
- Scikit-learn [16]
- EasyOCR [8]
- PyTesseract [11]

Miscellaneous

- Matplotlib.pyplot [7]
- Built-in libraries:

  time

  collections

References

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

[2] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[3] A. Frome, G. Cheung, A. Abdulkader, M. Zennaro, B. Wu, A. Bissacco, H. Adam, H. Neven, and L. Vincent. Large-scale privacy protection in google street view. In *2009 IEEE 12th International Conference on Computer Vision*, pages 2373–2380, 2009.

[4] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation, 2014.

[5] C. R. Harris, K. J. Millman, S. J. van der Walt, R. Gommers, P. Virtanen, D. Cournapeau, E. Wieser, J. Taylor, S. Berg, N. J. Smith, R. Kern, M. Picus, S. Hoyer, M. H. van Kerkwijk, M. Brett, A. Haldane, J. F. del Río, M. Wiebe, P. Peterson, P. Gérard-Marchant, K. Sheppard, T. Reddy, W. Weckesser, H. Abbasi, C. Gohlke, and T. E. Oliphant. Array programming with NumPy. *Nature*, 585(7825):357–362, Sept. 2020.

[6] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications, 2017.

[7] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95, 2007.

[8] jaided.ai. Easyocr, 2020.

[9] G. Jocher. YOLOv5 by Ultralytics, May 2020.

[10] G. Jocher, A. Chaurasia, and J. Qiu. Ultralytics YOLO, Jan. 2023.

[11] A. Kay. Tesseract: an open-source optical character recognition engine. *Linux J.*, 2007(159):2, jul 2007.

[12] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment anything, 2023.

[13] Q. Lhoest, A. Villanova del Moral, Y. Jernite, A. Thakur, P. von Platen, S. Patil, J. Chaumond, M. Drame, J. Plu, L. Tunstall, J. Davison, M. Šaško, G. Chhablani, B. Malik, S. Brandeis, T. Le Scao, V. Sanh, C. Xu, N. Patry, A. McMillan-Major, P. Schmid, S. Gugger, C. Delangue, T. Matussière, L. Debut, S. Bekman, P. Cistac, T. Goehringer, V. Mustar, F. Lagunas, A. Rush, and T. Wolf. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics.

[14] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[15] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. In *NIPS-W*, 2017.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[17] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection, 2016.

[18] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks, 2016.

[19] P. Umesh. Image processing in python. *CSI Communications*, 23, 2012.

[20] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I, 2001.

[21] Wes McKinney. Data Structures for Statistical Computing in Python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 56 – 61, 2010.

[22] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. L. Scao, S. Gugger, M. Drame, Q. Lhoest, and A. M. Rush. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, Oct. 2020. Association for Computational Linguistics.

[23] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao. Joint face detection and alignment using multi-task cascaded convolutional networks. *CoRR*, abs/1604.02878, 2016.