Smart Surveillance: Person Recognition using Textual description

Mayank Bumb
University of Massachusetts Amherst
mbumb@umass.edu

Saransh Bhachawat
University of Massachusetts Amherst
sbhachawat@umass.edu

Abstract

The manual identification of suspects in video footage is a time-consuming and resource-intensive task in law enforcement and public safety. To improve this process, we propose an automated solution utilizing deep learning. Our approach integrates Faster R-CNN for object localization within the UCF Crime dataset, effectively detecting individuals in various scenarios. Complementing this, we fine-tune ResNet50 on the PETA Dataset, enabling precise attribute matching, crucial for suspect identification.

Positioned at the confluence of computer vision, and deep learning, our project offers significant implications for security and surveillance. It aims to alleviate the workload on human analysts and contribute to more effective law enforcement strategies. Adhering to ethical standards and privacy laws, our solution aspires to be a benchmark in smart surveillance technology, balancing innovation with responsible usage.

1. Introduction

In recent years, the proliferation of surveillance technologies, particularly CCTV cameras, has become a staple in ensuring public safety. These systems are now ubiquitous in public spaces, marking a significant advancement in security measures. However, despite the widespread adoption of CCTV systems, the process of person identification and retrieval from vast amounts of footage remains predominantly manual. This manual approach is not only time-intensive but also susceptible to human error, highlighting a critical need for more efficient and accurate methods.

Addressing this gap, our project proposes a deep learning-based solution, leveraging the capabilities of Faster R-CNN and ResNet50 models. The core objective of this endeavor is to transform the laborious task of manually sifting through surveillance footage into an automated, efficient, and error-resistant process. Our system is designed to identify individuals based on specific attributes, such as clothing or accessories, input by the user. By integrating advanced computer vision techniques, the model can swiftly and accurately locate the individual of interest within the surveillance data.

The implementation of this deep learning approach aims not only to enhance the effectiveness of public surveillance systems but also to significantly reduce the time and resources currently expended in manual analysis.

2. Dataset

For the scope of our project, we employ two datasets: the UCF Crime dataset[1] and the PETA dataset[2]. The UCF Crime dataset, with its extensive collection of real-world surveillance videos, serves as the foundation for our CCTV video data analysis using Faster R-CNN, enabling efficient localization and identification of individuals in various settings. Complementing this, we are using the PETA dataset, known for its detailed annotations of pedestrian attributes, for image analysis and attribute recognition. By fine-tuning the ResNet50 model on the PETA dataset, our system gains the ability to accurately identify specific characteristics such as clothing and accessories based on user inputs. These datasets together prove to be rich and comprehensive resources for our project.

3. Related work

Recent advancements in pedestrian attribute recognition and human-centric perceptions are rapidly transforming the landscape of visual models in industrial applications. Traditionally, the challenge in pedestrian attribute recognition has been the aggregation of multiple attributes within a single model, making it difficult to discern specific features. To address this, an innovative approach introduced in paper [3] proposed an end-to-end trainable framework, featuring two key modules: the Attribute Localization Module (ALM) and the Attribute Correlation Module (ACM). ALM utilizes an attention mechanism for precise localization of attribute-specific regions, while ACM leverages the Transformer architecture to explore attribute correlations. This marked the first application of the Transformer architecture in pedestrian attribute recognition, offering a novel way to model attribute correlations.

Another study [4] tackled the challenge of visual variations and spatial shifts in high-level semantic attribute descriptions by employing a fusion strategy that combines high-level learned features with low-level features. This approach effectively models the diverse appearances and spatial distributions of attributes. The introduction of a specific Convolutional Neural Network (CNN) architecture with 1D convolution layers captures attribute patterns across different body parts, while LOMO features address viewpoint-invariant pedestrian re-identification.

Building upon these, paper [5] proposed a modified ResNet architecture for training deeper networks. These deeper architectures capture more contextual information, enhancing classification accuracy and generalizability. Experiments using PA-100K, PETA, and CelebA datasets provided insights into automating suspect identification and localization.

Furthermore, the recognition of computationally expensive challenges in localizing attribute-specific regions in video surveillance led to the development of a Deep Template Matching (DTM) method [6], which reduces computational complexity while maintaining effective local characteristic extraction.

A recent breakthrough in human-centric perceptions is highlighted in a new study focusing on the unified semantic structure of the human body across various tasks. This work introduces UniHCP, a Unified Model for Human-Centric Perceptions [7], which simplifies a range of human-centric tasks in an end-to-end manner using a plain vision transformer architecture. UniHCP, trained on 33 human-centric datasets, outperforms strong baselines in several in-domain and downstream tasks, setting new state-of-the-art benchmarks in human parsing, attribute prediction, ReID, and pedestrian detection. The success of UniHCP demonstrates the potential of leveraging the homogeneity of human-centric tasks to design general-purpose models. Although finetuned UniHCP has the best accuracy for the Pedestrian attributes task, the biggest issue with UniHCP model is that for initial training it requires dataset which all the lables (Pedestrian attributes, Pose estimation, Pedestrian detection, Person ReID, Human parsing). Having all this is not always possible. Hence, we implement Learning Disentangled Attribute Representations for Robust Pedestrian Attribute Recognition (DAFL)[8] whose mean accuracy is the highest for the given task with training only on 1 dataset.

These advancements underscore the growing pursuit of efficient, accurate, and computationally feasible solutions in object localization and pedestrian attribute recognition for surveillance. Our project aims to build upon these foundations, incorporating insights from these studies while addressing the specific requirements of practical surveillance applications.

Method	PA-100K	RAPv2
SSC [9]	81.87	-
C-Tran [10]	81.53	-
Q2L [11]	80.72	-
L2L [12]	82.37	-
DAFL [8]	83.54	81.04
UniHCP (finetune) [7]	86.18	82.34

Table 1. State of the art Pedestrian attribute recognition models

4. Methodology Used

Below is the methodology we are using to implement the object detection and Attribute detection models:

4.1. Object Detection Using Faster R-CNN

The object detection component of our project is critical for the identification and localization of subjects within video frames. We utilize the Faster R-CNN algorithm, acclaimed for its proficiency in real-time object detection. Our methodology is comprehensive, spanning from dataset preparation to post-processing of detection outputs.

4.1.1 Dataset Preparation

Our study benefits from the diverse and extensive UCF Crime and PETA datasets. These datasets encompass a wide spectrum of imagery, capturing various activities, behaviors, and environments. This breadth of data is paramount for training a detection model with the robustness to accurately identify individuals in a multitude of contexts.

4.1.2 Preprocessing

The preprocessing phase lays the foundation for effective model training and inference. We systematically resize the extracted frames to a standard resolution, ensuring that the input to the Faster R-CNN model remains consistent. Moreover, pixel values across all images are normalized to enhance the model's ability to discern features and patterns pertinent to object detection.

4.1.3 Model Selection and Training

The selection of the Faster R-CNN model is deliberate, drawing upon its innovative integration of deep convolutional networks and region proposal mechanisms. We capitalize on the pretrained weights from the COCO dataset to inherit a wealth of feature representations, facilitating the detection of various objects.

4.1.4 Customization for Pedestrian Detection

Our model undergoes a meticulous fine-tuning process on the selected datasets, concentrating on the detection of pedestrians. Annotated bounding boxes provide supervised signals, instructing the model to localize the pedestrian figures within the frames accurately.

4.1.5 Training Regimen

We tailor the training parameters to optimize performance. The learning rate is methodically scheduled to decrease over epochs, promoting the convergence of the model. We implement Stochastic Gradient Descent for optimization, carefully observing the model's validation performance to mitigate the risk of overfitting.

4.1.6 Post-processing

The detections from the model are refined through confidence thresholding and non-maximum suppression. These post-processing techniques are pivotal in ensuring the precision and relevance of the bounding box annotations.

4.1.7 Output Generation

The end product of this phase is a collection of frames with accurately annotated bounding boxes. These annotations are essential inputs for the subsequent phase, transitioning from object detection to attribute identification.

4.1.8 Implementation Details

The localization process is streamlined through an automated script, which extracts frames from the video files at specified time intervals. The frames are then processed through the trained Faster R-CNN model. The localized regions are cropped from the frames and used for further attribute analysis.

4.2. Attribute detection using RESNET

In this section, our objective is to identify individuals wearing specific clothing and accessories using CCTV surveillance data. We start by employing person detection, which helps us mark each person independently. Subsequently, we proceed to attribute detection for each individual.

Our model has been trained on the PETA dataset, consisting of 19,000 images featuring segmented individuals and their associated attributes. These attributes include upper body color, lower body color, the presence of a hat, and whether the person is carrying a bag or not. The dataset is compiled from various sources, encompassing diverse angles, lighting conditions, and image quality.

4.2.1 Upper body color Attribute

This is essentially a multi-class image classification task. The input is an image of a person, and the label is the color of the upper body clothing they are wearing, with a total of 6 possible colors. To train our model, we preprocess the input images and split them into a 80/20 training and testing set. We utilize the ResNet50 architecture with pre-trained ImageNet weights, freezing the first 143 layers as non-trainable. The subsequent layers are fine-tuned based on our training data.

The output of the ResNet50 model yields a 7x7x2048-dimensional feature map, which we flatten and pass through a fully connected layer. This fully connected layer leads to 6 scores, with the highest score determining the predicted color. We have also experimented with making the entire ResNet non-trainable and training only the fully connected layers, but this approach yielded less favorable results.

4.2.2 Binary attributes Finetuned ResNet50

We utilize a refined version of the ResNet50 model to identify attributes of a person, performing 14 binary classifications. This approach mirrors our method for Upper body color attribution, where we maintain a training-to-validation data ratio of 80/20 and keep 143 layers of the network fixed during training. After obtaining the 7x7x2048 dimension embeddings, we proceed to flatten this output. The data then passes through several fully connected layers, culminating in a dense layer with 14 neurons. Each neuron employs softmax for evaluation, accommodating scenarios with multiple positive predictions. For evaluating model performance, we use BinaryAccuracy, and for loss calculation, BinaryCrossentropy is applied.

4.2.3 Binary attributes DAFL Framework

We also detect attributes of the person using DAFL framework. Here we consider 14 attributes with binary classification like wearing formals/casuals or male/female, etc.

DAFL focuses on identifying specific spatial regions related to each attribute and gathering unique regional features for every attribute. It also leverages semantic characteristics, which are especially useful for attributes that share similar spatial distributions. To capitalize on these consistent semantic characteristics across different samples, the framework incorporates learnable semantic queries. It also applies triplet loss to the features of each attribute, enhancing their distinctiveness.

The core components of the DAFL framework are the SSCA (Spatially Specific Channel Attention) module and

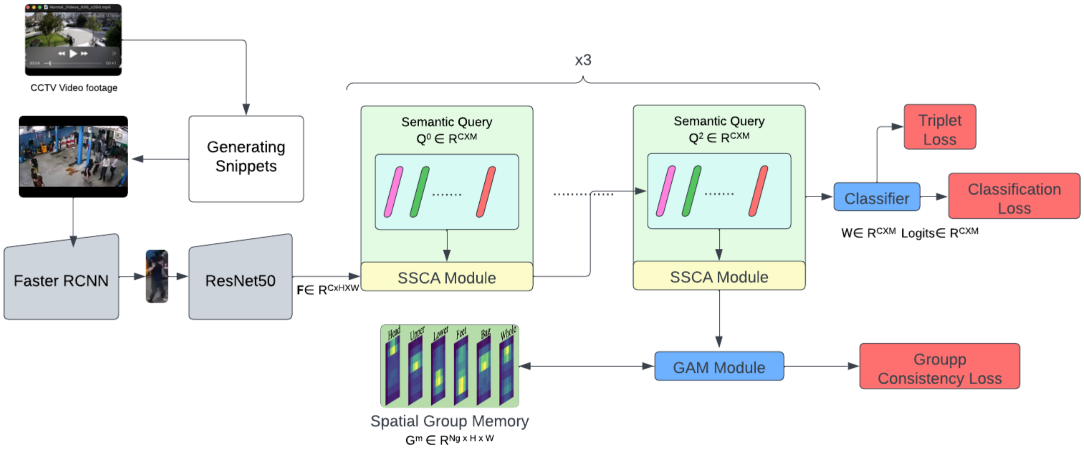


Figure 1. Model pipeline architecture

the GAM (Global Attribute Map) module. The SSCA module is designed to precisely locate spatial regions corresponding to each attribute and to aggregate these regions into attribute-specific features. The attention map for the m-th SSCA is calculated as follows:

$$A = \text{Softmax}\left(\frac{\theta(Q)^T \phi(F)}{\sqrt{C}}\right),$$

where Q is the query vector and F is the feature vector. We then calculate specific spatial feature which is as follows:

$$F^S = A\psi(F^T),$$

where A is the attention map evaluated above and F is the feature vector. This spatial feature vector acts like a query vector for the next SSCA block.

Table 2. The six spatial groups of attributes in PETA Dataset

Group	**Attribute**
Head (G1)	Hair
UpperBody (G2)	LongSleeve, Casual, Stripes, UpperLogo, Other
LowerBody (G3)	Casual, Trousers, Skirt, Trousers, Shorts, Skirt&Dress
Feet (G4)	Sneaker
Whole (G5)	accessoryNothing, carryingNothing, personalLess30, personalLess60, personalMale

The framework tackles the challenge of uneven attribute distribution in data, where some attributes lack sufficient positive samples for learning query attention maps and developing effective features by leveraging the fact that some attributes, like ”Hat” and ”Glasses” or ”UpperLogo” and ”UpperPlaid,” often share similar spatial locations in images. We merge the query attention maps of attributes that share a similar spatial distribution into groups which is evaluated as:

$$G_k^a = \frac{1}{|G_k|}\sum_{m\in G_k}\frac{1}{|R|}\sum_{i=1}^{b_t}\nVdash_{\{R\}}A_{i,m},$$

To address the variability that arises due to the small size of batches and the randomness in sampling, the system employs a spatial group memory, denoted as Gm, which is updated using a momentum-based method. This approach ensures that the group attention remains stable and uniform across different batches.

$$G_k^m \leftarrow (1-\alpha)\times G_k^m + \alpha\times G_k^a,$$

Now coming to losses, we consider 3 losses. Group loss, triplet loss and the classification loss. Group loss is to rectify the imprecise spatial localization of minority attributes and is evaluated as follows:

$$L_{\text{group}} = \frac{1}{b_t}\left\|\sum_{i=1}^{b_t}\sum_{k=1}^{K}\sum_{m\in G_k} G_k^m - A_{i,m}\right\|_2,$$

Triplet loss is to solve the problem of the distance among similar (positive) samples being larger than the distance between dissimilar (positive and negative) samples.

$$L_{\text{pos},m} = \sum_{j \in N_m^P} \max(0, D(a_j^P, f_j^P) - D(a_j^P, f_j^n)),$$

$$L_{\text{neg},m} = \sum_{j \in N_m^n} \max(0, D(a_j^n, f_j^n) - D(a_j^n, f_j^P)),$$

These are the positive and the negative triplet losses and the classification loss is calculated as:

$$\text{BCE Loss} = -\frac{1}{N} \sum_{i=1}^{N} [y_i \cdot \log(p_i) + (1 - y_i) \cdot \log(1 - p_i)]$$

We have trained our model using ResNet50 to get embeddings and then the above mentioned equations and loss functions to get better results than normally finetuning ResNet50 model.

4.3. Query-Based Person Mapping

An innovative aspect of our project is the implementation of a query-based person mapping system. This system is designed to process natural language queries and map them to visual attributes observed in the detected individuals.

Natural Language Processing: The system begins by parsing the input query to extract relevant attributes. For instance, consider the sample query:

```
query_text = "carrying something,
hair short, male"
```

This query is indicative of three distinct attributes: an action (carrying something), a physical characteristic (short hair), and a demographic detail (male).

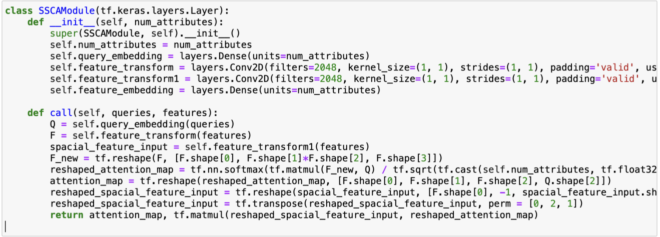
Attribute Matching: Once the attributes are extracted, the system iterates over all the people's attributes. It then tries to find if the attributes that have been inputted are present in the person or not. If not, the system moves to another person. If yes then the system maps the two.

Mapping and Output: The final mapping associates the query with the identified individual or individuals in the video frames. If a match is found, the output consists of the bounding box or boxes around the individual(s) that correspond to the query's description.

Thus query-based person mapping serves as a powerful tool for quickly identifying individuals based on descriptive queries, streamlining the process of locating persons of interest within extensive video data.

4.4. Interesting code snippets

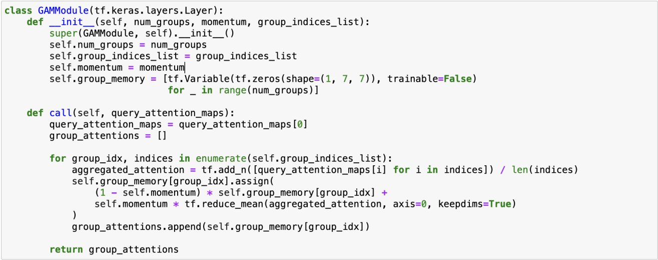
Figure 2, 3, and 4 are code snippets of a few interesting concepts deployed in our model.



```
class SSCAModule(tf.keras.layers.Layer):
    def __init__(self, num_attributes):
        super(SSCAModule, self).__init__()
        self.num_attributes = num_attributes
        self.query_embedding = layers.Dense(units=num_attributes)
        self.feature_transform = layers.Conv2D(filters=2048, kernel_size=(1, 1), strides=(1, 1), padding='valid', us
        self.feature_transform1 = layers.Conv2D(filters=2048, kernel_size=(1, 1), strides=(1, 1), padding='valid', u
        self.feature_embedding = layers.Dense(units=num_attributes)

    def call(self, queries, features):
        Q = self.query_embedding(queries)
        F = self.feature_transform(features)
        spacial_feature_input = self.feature_transform1(features)
        F_new = tf.reshape(F, [F.shape[0], F.shape[1]*F.shape[2], F.shape[3]])
        reshaped_attention_map = tf.nn.softmax(tf.matmul(F_new, Q) / tf.sqrt(tf.cast(self.num_attributes, tf.float32
        attention_map = tf.reshape(reshaped_attention_map, [F.shape[0], F.shape[1], F.shape[2], Q.shape[2]])
        reshaped_spacial_feature_input = tf.reshape(spacial_feature_input, [F.shape[0], -1, spacial_feature_input.sh
        reshaped_spacial_feature_input = tf.transpose(reshaped_spacial_feature_input, perm = [0, 2, 1])
        return attention_map, tf.matmul(reshaped_spacial_feature_input, reshaped_attention_map)
```

Figure 2. SSC module implementation



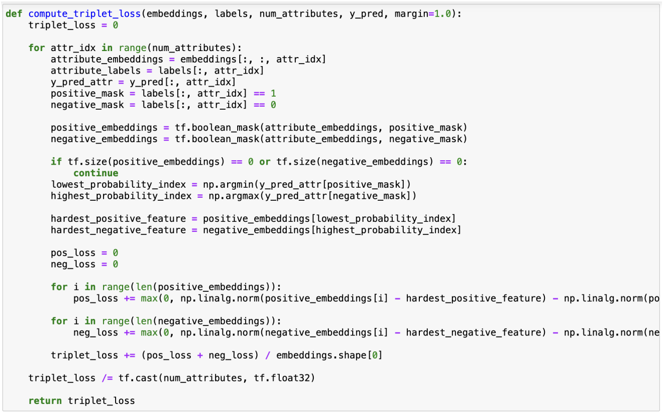
```
class GAMModule(tf.keras.layers.Layer):
    def __init__(self, num_groups, momentum, group_indices_list):
        super(GAMModule, self).__init__()
        self.num_groups = num_groups
        self.group_indices_list = group_indices_list
        self.momentum = momentum
        self.group_memory = [tf.Variable(tf.zeros(shape=(1, 7, 7)), trainable=False)
                             for _ in range(num_groups)]

    def call(self, query_attention_maps):
        query_attention_maps = query_attention_maps[0]
        group_attentions = []

        for group_idx, indices in enumerate(self.group_indices_list):
            aggregated_attention = tf.add_n([query_attention_maps[i] for i in indices]) / len(indices)
            self.group_memory[group_idx].assign(
                (1 - self.momentum) * self.group_memory[group_idx] +
                self.momentum * tf.reduce_mean(aggregated_attention, axis=0, keepdims=True)
            )
            group_attentions.append(self.group_memory[group_idx])

        return group_attentions
```

Figure 3. GAM module implementation



```
def compute_triplet_loss(embeddings, labels, num_attributes, y_pred, margin=1.0):
    triplet_loss = 0

    for attr_idx in range(num_attributes):
        attribute_embeddings = embeddings[:, :, attr_idx]
        attribute_labels = labels[:, attr_idx]
        y_pred_attr = y_pred[:, attr_idx]
        positive_mask = labels[:, attr_idx] == 1
        negative_mask = labels[:, attr_idx] == 0

        positive_embeddings = tf.boolean_mask(attribute_embeddings, positive_mask)
        negative_embeddings = tf.boolean_mask(attribute_embeddings, negative_mask)

        if tf.size(positive_embeddings) == 0 or tf.size(negative_embeddings) == 0:
            continue
        lowest_probability_index = np.argmin(y_pred_attr[positive_mask])
        highest_probability_index = np.argmax(y_pred_attr[negative_mask])

        hardest_positive_feature = positive_embeddings[lowest_probability_index]
        hardest_negative_feature = negative_embeddings[highest_probability_index]

        pos_loss = 0
        neg_loss = 0

        for i in range(len(positive_embeddings)):
            pos_loss += max(0, np.linalg.norm(positive_embeddings[i] - hardest_positive_feature) - np.linalg.norm(po

        for i in range(len(negative_embeddings)):
            neg_loss += max(0, np.linalg.norm(negative_embeddings[i] - hardest_negative_feature) - np.linalg.norm(ne

        triplet_loss += (pos_loss + neg_loss) / embeddings.shape[0]

    triplet_loss /= tf.cast(num_attributes, tf.float32)

    return triplet_loss
```

Figure 4. Triplet Loss implementation

5. Evaluation Metric

- For the Attribute detection task we have the training and testing dataset from which we evaluate our model based on accuracy %, F1 score, Precision and Recall. Outputs attached in the Results section.
- For the complete pipeline from snapshot of the video and text as input to marking a person matching the attributes, we don't have a complete dataset as such. Hence, we run it on 50 examples and manually evaluate the predictions which gave us an accuracy of correctly marked people.

6. Results

6.1. Classification Performance

	Multiple Binary		Multi-class
	ResNet50	DAFL	
Accuracy	0.8235	0.84682715	0.5278
F1-Score	0.8210	0.83401084	0.4469
Precision	0.8310	0.85873735	0.2966
Recall	0.8113	0.8106684	0.9030

Table 3. Performance Measures across classification models.

Our comparative analysis of classification models employed Multiple Binary Classification and Multi-class Classification approaches. The performance metrics, which include Accuracy, F1-Score, Precision, and Recall, provide a comprehensive understanding of the models' predictive capabilities.

6.1.1 Binary Classification Analysis

Finetuning ResNet50: This binary classifier set a benchmark with a substantial accuracy rate, showcasing its ability to make correct predictions in a majority of test cases. The F1-Score, a harmonic mean of precision and recall, corroborates the model's balanced detection capabilities. High precision indicates the model's robustness in identifying positive cases, while a notable recall rate reflects its adeptness at capturing a significant proportion of actual positive instances.

DAFL Framework: An advancement over finetuned ResNet50, this model marked an improvement in all performance metrics, especially precision. This enhancement signifies a refined prediction model, where a higher percentage of identified positives are true positives, hence elevating the model's reliability.

6.1.2 Multi-class Classification Analysis

The Multi-class Classification model, while demonstrating a high recall rate, indicated a propensity for overfitting, as evidenced by its low precision and accuracy scores. This suggests the model's sensitivity in detecting positive cases, albeit at the cost of incorrectly labeling negative cases as positive, which is indicative of a model that has memorized the training data rather than learned to generalize.

6.2. Loss Analysis

The training and validation loss curves for these models, illustrated in Figures 5, 6, and 7, offer visual insights into the training dynamics and generalization tendencies of the models.

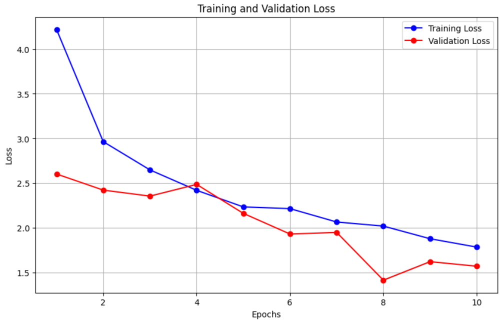


Figure 5. Training and Validation Loss for Multiple Binary Classification models. The convergence of the two curves suggests effective learning and validation performance.

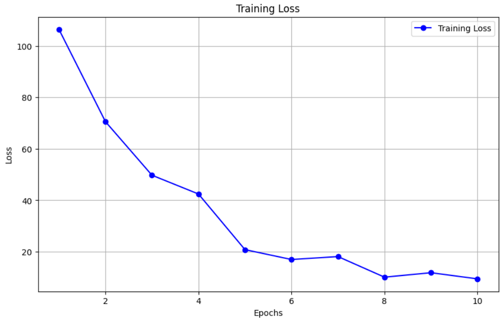


Figure 6. Training Loss for the Enhanced Attribute Recognition Model. A steady decline in loss indicates consistent learning across epochs.

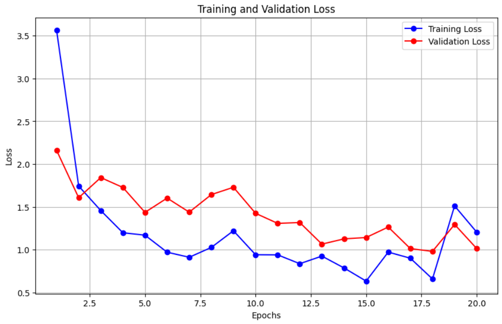


Figure 7. Training and Validation Loss for Multi-class Classification. The divergence of loss curves reflects overfitting and poor generalization to the validation set.

For the binary classifiers, the downward trajectory of loss over epochs suggests effective learning, with DAFL Framework demonstrating a particularly strong alignment between training and validation loss. This is indicative of its superior generalization capabilities when applied to unseen data. In contrast, the multi-class model's loss curves reveal discrepancies between training and validation, with a higher degree of fluctuation in the validation loss. This

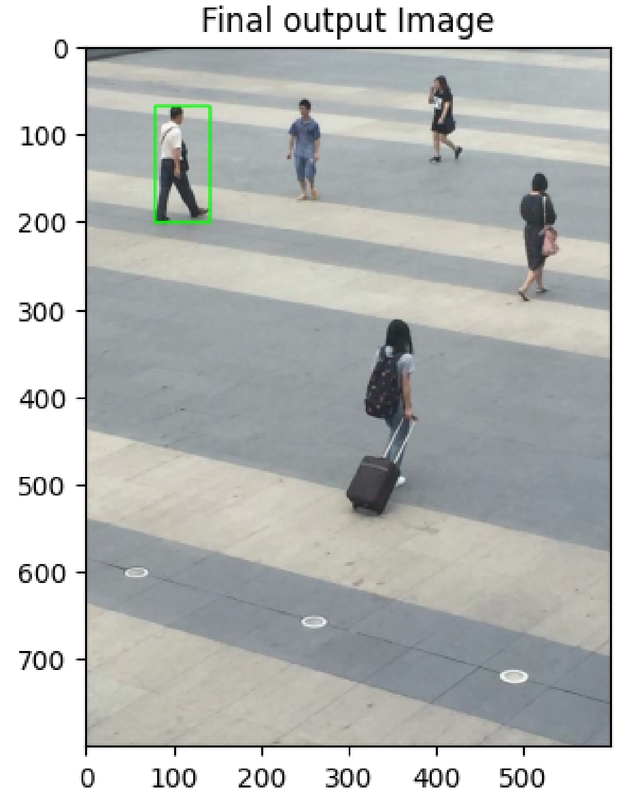


Figure 8. Final output

variability raises concerns regarding the model's stability and suggests a need for additional tuning or regularization techniques to enhance its predictive performance across diverse datasets.

Figure 8 demonstrates a successful identification instance of a person via a query describing their attributes such as short hair and gender.The green bounding box signifies the model's precision in isolating the individual from the broader scene.

This result validates our model's utility in practical applications, where rapid and reliable person identification is paramount. It also underscores the potential of our approach in enhancing the efficiency of video surveillance analysis and related domains.

7. Limitations

Our model lacks few challenges like

- In case of extreme brightness which occurs during summers, we can see that the model predicts more people with black color upper body clothes since because of the light all darker colors look more like black color.
- The model is little biased towards men. It predicts more male than female. Using a Marginal Interventional Mixture (MIM) models will help tackle this
- Long time problem of occlusion still persists in our model as well. We have just implemented it on snippets of videos, if we run it across video, the results do get better.

8. Conclusion

In conclusion, our project embarked on a mission to enhance surveillance capabilities through the automation of suspect identification and localization within CCTV footage. Leveraging the power of advanced machine learning algorithms, specifically Faster R-CNN for object detection and ResNet50 for attribute recognition, we developed a system capable of analyzing complex video data efficiently and accurately.

The binary classification models, finetuning ResNet50 and DAFL framework, demonstrated strong performance in detecting and classifying individuals in video frames, with DAFL outperforming finetuned ResNet in terms of accuracy, precision, and F1-score. Despite the multi-class classification model's high recall, it fell short in other performance metrics, indicating a propensity for overfitting and a need for improved precision.

The analysis of the training and validation loss curves provided further insights into the models' behavior, with both binary classification models showing stable convergence, whereas the multi-class model exhibited fluctuations that suggested a gap in generalization to unseen data.

Our project has laid a solid foundation for the application of deep learning in public safety and surveillance. The results underscore the potential of machine learning in augmenting human efforts with automated systems, opening avenues for more advanced research and practical applications in the field of security and surveillance technology.

9. Future Work

We plan to explore more in this topic. Future work which we plan to do are:

- Right now we have just implemented on the PETA dataset, whereas a lot of state of the art models evaluate their models on PA-100k dataset. Since, PA-100k wasn't available for free, we mailed the creator and we just got the access to the dataset. We could you evaluate our model on such short notice, hence we plan to carry out results on other datasets. In case we achieve results as mentioned in the DAFL framework paper, we plan to upload the code on github for public use since the DAFL framework code isn't available publically.
- The recall of multi-class classification is very high and the precision is very low. We plan to augment the data by varying lighting conditions to get more balanced outputs.

- DAFL uses ResNet50 for getting the initial embeddings. We plan to try out ResNet101 and ResNet152 architectures to generate the initial embeddings.
- We have divided our pipeline which has been independently trained for different sub tasks. We plan to create a single architecture pipeline. Disadvantage of having different pipelines joined together is that, 1st model works best for a particular kind of data and the cropping that it provides is different from the PETA dataset with which we have trained our 2nd model. This discrepancy will vanish if we have a single architecture.

Explore the incorporation of additional datasets to further diversify the training, enhancement of the attribute recognition phase to include more granular attributes, and the implementation of real-time processing capabilities for live video streams.

References

[1] Sultani, W., Chen, C., & Shah, M. (2018). Real-world anomaly detection in surveillance videos. *Proceedings of the IEEE conference on computer vision and pattern recognition*, (pp. 6479-6488).

[2] Deng, Y., Luo, P., Loy, C. C., & Tang, X. (2014). Pedestrian attribute recognition at far distance. *Proceedings of the 22nd ACM International Conference on Multimedia*, 789-792.

[3] Weng, D., Tan, Z., Fang, L., & Guo, G. (2023). Exploring attribute localization and correlation for pedestrian attribute recognition. *Neurocomputing*, 531, 140-150.

[4] Chen, Y., Duffner, S., Stoian, A., Dufour, J. Y., & Baskurt, A. (2018). Pedestrian attribute recognition with part-based CNN and combined feature representations. *VISAPP 2018*.

[5] Bekele, E., & Lawson, W. (2019). The deeper, the better: Analysis of person attributes recognition. *IEEE International Conference on Automatic Face & Gesture Recognition (FG 2019)*, 1-8.

[6] Zhang, J., Ren, P., & Li, J. (2020). Deep template matching for pedestrian attribute recognition with the auxiliary supervision of attribute-wise keypoints. *arXiv preprint arXiv:2011.06798*.

[7] Ci, Y., Wang, Y., Chen, M., Tang, S., Bai, L., Zhu, F., ... & Ouyang, W. (2023). UniHCP: A Unified Model for Human-Centric Perceptions. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 17840-17852.

[8] Jia, J., Gao, N., He, F., Chen, X., & Huang, K. (2022). Learning disentangled attribute representations for robust pedestrian attribute recognition. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(1), 1069-1077.

[9] Jia, J., Chen, X., & Huang, K. (2021). Spatial and semantic consistency regularizations for pedestrian attribute recognition. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 962–971.

[10] Lanchantin, J., Wang, T., Ordonez, V., & Qi, Y. (2020). General multi-label image classification with transformers. *arXiv preprint arXiv:2011.14027.*

[11] Liu, S., Zhang, L., Yang, X., Su, H., & Zhu, J. (2021). Query2label: A simple transformer way to multi-label classification. *arXiv preprint arXiv:2107.10834*.

[12] Li, W., Cao, Z., Feng, J., Zhou, J., & Lu, J. (2022). Label2label: A language modeling framework for multi-attribute learning. *European Conference on Computer Vision (ECCV)*, 562-579.