MASTER'S DEGREE IN INDUSTRIAL ENGINEERING
+ MASTER'S DEGREE IN SMART INDUSTRY

MASTER'S THESIS

Design, development, and construction of an event identification system based on the processing of images of different natures

Author: María López-Chaves Estévez

Director: Emilio Manuel Domínguez Adán

Madrid July 2024

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Diseño, desarrollo y construcción de un sistema de identificación de eventos sobre el procesado de imágenes de diferente naturaleza

en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el curso académico 2023/24 es de mi autoría, original e inédito y no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido tomada de otros documentos está debidamente referenciada.

LOPEZ-CHAVES ESTEVEZ MARIA - 39467350V

Firmado digitalmente por LOPEZ-CHAVES ESTEVEZ MARIA - 39467350V
Fecha: 2024.07.14 22:36:17 +02'00'

Fdo.: María López-Chaves Estévez Fecha: 17/07/2024

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

DOMINGUEZ ADAN EMILIO MANUEL - 02620715A

Firmado digitalmente por DOMINGUEZ ADAN EMILIO MANUEL - 02620715A
Fecha: 2024.07.13 21:59:41 +02'00'

Fdo.: Emilio Manuel Domínguez Adan Fecha: 17/07/2024

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D. María López-Chaves Estévez DECLARA ser el titular de los derechos de propiedad intelectual de la obra: Diseño, desarrollo y construcción de un sistema de identificación de eventos sobre el procesado de imágenes de diferente naturaleza, que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, los derechos de digitalización, de archivo, de reproducción, de distribución y de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar "marcas de agua" o cualquier otro sistema de seguridad o de protección.
b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.
c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.
d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.
e) Asignar por defecto a estos trabajos una licencia Creative Commons.
f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente)*.

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

a) Que la Universidad identifique claramente su nombre como autor de la misma
b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.
c) Solicitar la retirada de la obra del repositorio por causa justificada.
d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

El autor se compromete a:

a) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.
b) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.
c) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.
d) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción

de derechos derivada de las obras objeto de la cesión.

6º. Fines y funcionamiento del Repositorio Institucional.

La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusive del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.
- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.
- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 17 de julio de 2024

ACEPTA

LOPEZ-CHAVES ESTEVEZ MARIA - 39467350V

Firmado digitalmente por LOPEZ-CHAVES ESTEVEZ MARIA - 39467350V
Fecha: 2024.07.14 22:37:06 +02'00'

Fdo. María López-Chaves Estévez

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:

MASTER'S DEGREE IN INDUSTRIAL ENGINEERING
+ MASTER'S DEGREE IN SMART INDUSTRY

MASTER'S THESIS

Design, development, and construction of an event identification system based on the processing of images of different natures

Author: María López-Chaves Estévez

Director: Emilio Manuel Domínguez Adán

Madrid July 2024

DESIGN, DEVELOPMENT, AND CONSTRUCTION OF AN EVENT IDENTIFICATION SYSTEM BASED ON THE PROCESSING OF IMAGES OF DIFFERENT NATURES

Author: López-Chaves Estévez, María.
Director: Domínguez Adán, Emilio Manuel.
Collaborating Entity: ICAI – Universidad Pontificia Comillas.

ABSTRACT

In a context where the physical security of industrial facilities is increasingly important, the need for an automated solution for detecting and identifying intruders is crucial.

This work presents a scalable and reproducible approach to identify intruders in industrial installations through object detection in surveillance camera videos, transmitting these detections to a centralized cluster, and employing various deep learning (DL) models within the CLIP (Contrastive Language-Image Pre-Training) framework, complemented with large language models (LLM).

The system effectively detects objects in recordings, sends them via MQTT (Message Queuing Telemetry Transport), and then they are classified using various algorithms.

Keywords: Industrial Security, Object Detection, Intruder Identification, MQTT, CLIP Framework, LLM.

1. Introduction

Critical infrastructures, defined as essential elements supporting the vital functions of our society, have evolved into assets whose security means constant challenges. This forces a reconsideration of traditional paradigms to adapt to a dynamic and complex environment.

The current situation is characterized by the presence of sophisticated technologies that not only improve efficiency and connectivity but also increase the attack surface because of the existence of a complex network of interdependencies.

In this dynamic and complex environment, automating intruder detection becomes an essential pillar for quickly and accurately identifying any intrusion attempt, as human response, while valuable, can be limited by factors such as reaction speed and the need for constant attention.

There are more such automated systems each day, although there are still issues with eliminating false alarms and accurately identifying objects, especially under challenging and changing environmental conditions.

The application of algorithms aims to redefine the response by increasing efficiency and reducing workload. If necessary, human security measures would be employed after their work.

This project will automate intruder detection using thermal cameras (technology that has already proven its convenience in complex environments) in an electrical sector company's facilities.

This approach aims to provide an automated and intelligent surveillance system capable of offering a more effective tool for protecting critical installations, which can also be extrapolated to other areas such as residential security.

2. State of the Art

Video surveillance systems have advanced significantly in recent decades, evolving from manual solutions with high human resource costs to complex systems based on automation.

Historically, Computer Vision (CV) depended on manually creating reference points. The introduction of Machine Learning (ML) revolutionized this field, eliminating the need to manually define these rules by using "features" that detect specific patterns in images, but whose creation is manual. Deep Learning (DL) algorithms extract these patterns themselves and convert them into mathematical equations, requiring only a large amount of labeled data. Additionally, the possibility of reusing pre-trained networks (Transfer Learning, TL) developed by large companies offers a solution to the hardware limitations of individual users.

On the one hand, to process images, there are multiple specialized libraries (Scikit-Image, Numpy, PIL, Mahotas, etc.), among which OpenCV stands out for its high performance and suitability for complex tasks.

On the other hand, there are a variety of algorithms for automatic individual detection characterized by the acquisition system, the number of potential subjects followed, and their categorization. For example, some algorithms focus on information about physical appearance, adding data acquired through tracking algorithms, while others reduce complexity and are independent of appearance variability by basing its analysis on motion information although they achieve slightly worse results.

These latter automated systems are called SVAD (Surveillance Video Anomaly Detection) and are based on analyzing frames at the pixel level to identify deviations in temporal and spatial behaviors compared to habitual behaviors. This process is divided into:

1. Identification of moving objects through Frame differencing (frame comparison), Optical Flow (2D velocity field that explains the movement between frames and that it can be sparse if it only analyses a limited number of features or dense if it uses all pixels), or Background subtraction (separates the background from the detected objects or foreground).

2. Classification of objects using ML algorithms (Random Forest Classifier, KNN, or Naive Bayes) or DL (Convolutional Neural Networks, CNN) with pre-trained models such as AlexNet, VGGNet, GoogLeNet/Inception, Resnet, etc.

2.1. Technologies

2.1.1. MQTT

MQTT is used as a communication protocol between devices, applied over TCP (Transmission Control Protocol). It is widely used due to its simplicity, efficiency, low latency, and minimal resource consumption.

It uses a publisher/subscriber methodology where clients can be

publishers, subscribers, or both. It has a broker or server that receives messages from clients, filters them, and sends them to the appropriate ones depending on the topic.

Different Quality of Service (QoS) levels can be specified: 0 (no reception confirmation), 1 (does not ensure no duplicates), and 2 (guarantees the message is delivered only once).

2.1.2. Docker

Docker is an open platform for developing and distributing applications. Applications created with it are packaged with all necessary dependencies in a standard and lightweight format called a container. These virtualized applications can run anywhere without modification.

2.1.3. CLIP Model

DL model characterized by an architecture with two encoders and contrastive learning to map images and text into a shared latent space.

On the one hand, for image encoding, different models can be used (typically based on CNN or Visual Transformers named, ViT) to convert a preprocessed image into a fixed high-dimensional vector representation. Two model families have been used: Resnet50 (a 50-layer CNN based on residual connections) and ViT (Transformer architecture over image patches), although any widely pre-trained model for object classification could be used but eliminating the last layer.

On the other hand, the text encoder transforms the semantic meaning of textual descriptions into the same latent space as the image encoder's output, usually using Transformer-based architectures.

In the shared latent space, embeddings from both encoders are compared using cosine similarity after normalization.

3. Existing Architecture

This project is based on the current architecture of an electrical sector company with the following elements:

- A network of thermal cameras monitoring the exterior of various facilities, communicating with a Control Center via a VPN.
- Sensors for precise detection, which can be motion or laser-based.

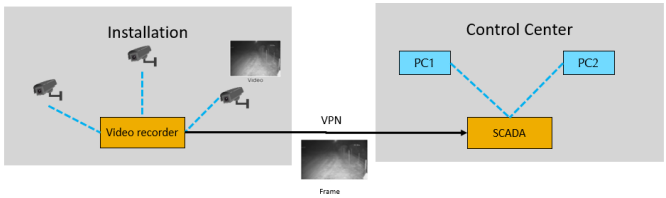


Figure 1 Part of the existing architecture in which we are interested

Each facility has a video recorder receiving videos from different cameras and transmitting these signals via VPN to a SCADA platform accessible from the Control Center.

The main problem with this infrastructure is its lack of automation, requiring a high manual workload from Control Center operators.

The developed solution aims to improve efficiency and reduce workload while achieving greater precision.

4. Solution Vision

The first step for analyzing surveillance camera recordings involves detecting objects in the videos, which has noise related issues. Once the object is detected, we must identify it. We use DL classification algorithms due to their ability to recognize complex patterns, with the added benefit of using TL models.

The real data we work with are black and white recordings from security cameras outside various facilities.

Assuming reliable communications between facilities and the Control Center with very low latency (less than a second in all cases), detected objects with the motion detection algorithm will be sent through MQTT to another algorithm for classification. If communications were poor, detection and classification would be done at each facility, sending only alarms.

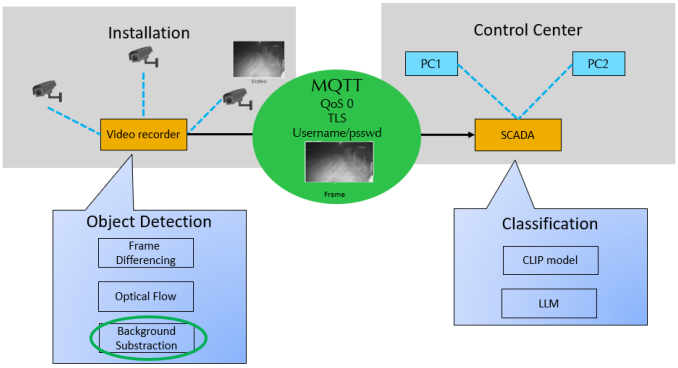
MQTT is chosen for its low latency, asynchronous communication, scalability, different QoS levels, and TLS (Transport Layer Security).

The DL algorithm used is the CLIP model, which overcomes the limitation of predicting a fixed number of categories and performs zero-shot classification as, due to the lack of diversity in our data, the possibility of using data augmentation techniques was rapidly eliminated.

A scheme of the developed solution would be the following:

Figure 2 Developed Solution

5. Intruder Detection



Object detection involves locating moving elements within a video sequence. Three different object detection methods will be compared using both MOG2 and KNN for one of them.

MOG2 (Mixture of Gaussians) represents a statistical approach for modeling the probability distribution of data that could be generated by multiple curves.

KNN (K-Nearest Neighbors) is based on the implementation of the KNN algorithm in CV.

Background Subtraction with MOG2 will be the most suitable solution due to having less noise, for its ability to detect gradual changes, and its lower computational cost.

5.1. Frame Differencing

This method subtracts the current video frame from the previous one, annotating the differences, which are supposed to correspond to motion (motion mask).

It begins by calculating the difference between frames in grayscale, but this produces some spots. These are eliminated by obtaining the motion mask (binary image with white pixels for the foreground and black for the background). A pixel will be based as foreground if its difference with the one of the previous

frame exceeds a defined adaptive threshold.

Then, object detectors are used to save points given by the Teh-Chinu approximation algorithm on contours in this mask if they exceed a threshold.

Finally, Non-maximum Suppression (NMS) is applied to filter predictions.

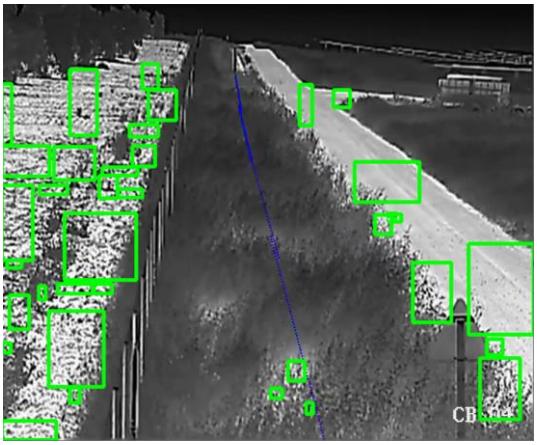


Figure 3 Frame Differencing

5.2. Optical Flow

Determines the 2D velocity field obtained by projecting 3D velocities onto the image plane, resulting in a displacement vector that explains point movement between frames.

Sparse methods like Lukas Kanade detect the most distinctive features of an image, tracking these points between consecutive frames to visualize movement. Dense methods like Gunnar Farneback consider all pixels between frames.

In this case, contours are searched while using the flow angle to filter unlikely detections.

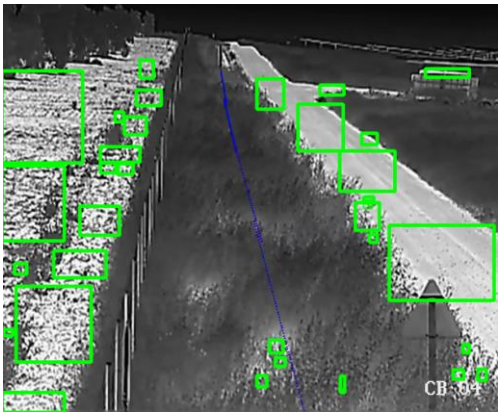


Figure 4 Dense Optical Flow

6. Background Subtraction

Calculates the foreground mask by subtracting the current frame from the dynamic background and applying a threshold to this differencing using the following steps:

1. Background Initialization: In our case, it is progressive as it is partially generated over time.

2. Background Update and Foreground Detection: This involves comparing the current image with the background to classify pixels as foreground or background. It includes the following steps:

    a. Preprocessing: To avoid detecting insignificant changes using geometric and intensity adjustments.

    b. Pixel Classification: Pixels are classified as background or foreground depending on whether the difference between the current image and the background exceeds a threshold (if it does, it is part of the foreground).

c. Post-processing: To enhance the consistency of the foreground mask using morphological operations and blurring the motion mask.

3. Selective Background Maintenance: Using a much higher learning rate for pixels classified as background than for those classified as foreground.

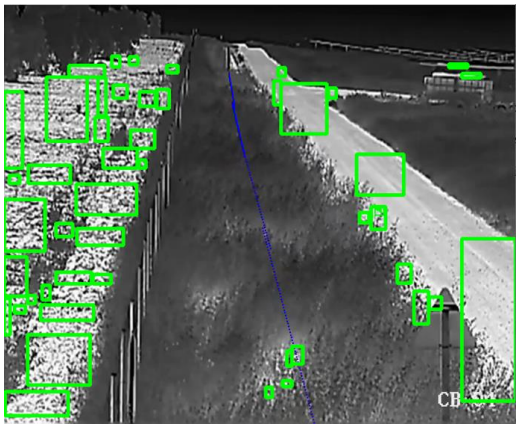


Figure 5 BS-KNN

Since changes in the background will be gradual and we have not a lot of pixels classified as foreground, we will use MOG2 instead of KNN.

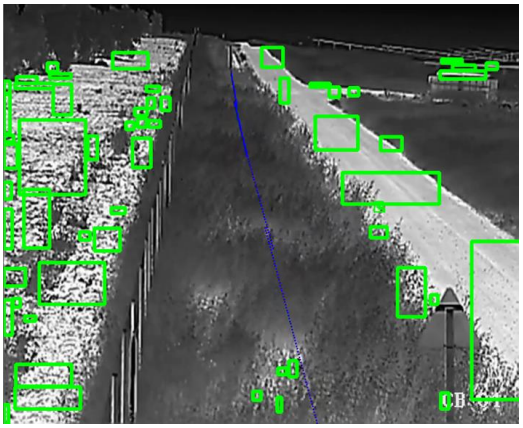


Figure 6 BS-MOG2

7. MQTT

For this project, we used an open-source broker: Mosquitto. This choice is based on its comprehensive documentation, large developer community, and low resource consumption.

Docker Compose will be used to create this server, facilitating scalability by allowing the addition of multiple applications or services, using volumes or bind mounts to store persistent data, and establishing an order for the startup of different containers based on dependencies.

We will create a Docker Compose file that uses the Mosquitto broker, with configurable ports set to 1883 (default port) and 8883 (TLS), which are mapped to the corresponding ports of the container. It will automatically restart and belong to a network called “my network”.

To ensure a secure communication channel, we will use TLS. We will generate a self-signed certificate from the certificate authority (CA) and additional certificates for the server and two clients using the SHA-256 cryptographic hash algorithm with a 2048-bit private key.

Additionally, four bind mounts will be used:

- Config: Broker configuration.
- Certs: Stores the CA and server certificates.
- Data: In-memory database.
- Log: Stores events.

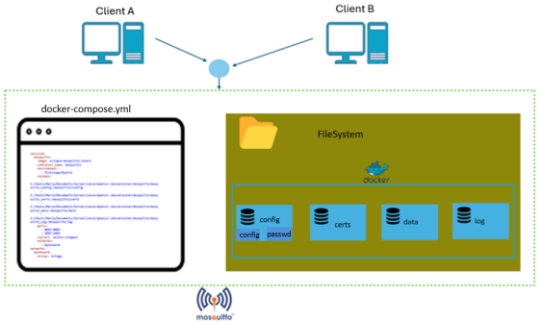


Figure 7 MQTT Architecture

We will use the *mosquitto_passwd* command to create passwords that will be saved in a file of the same name and referenced in *mosquitto_config*.

Clients will send and receive messages under a single topic called “ObjectDetected” (simplifying the solution to a single installation). These clients can be:

- Publisher: The object detection algorithm using Background Subtraction with QoS 0 for minimal latency.
- Subscriber: The CLIP model.

8. Intruder Classification

We will need a labeled database to compare the different TL models with quantifiable metrics. In our case, we will use CIFAR-100, a widely used dataset for classification tasks. It contains 60,000 color images of 32x32 pixels grouped into 100 classes, which are further grouped into 20 superclasses. There are about 600 images per class (500 for training and 100 for testing), and each image comes with a "fine" label representing the class it belongs to and a "coarse" label representing the superclass.

8.1. CLIP Model

The CLIP DL model, having been trained on the WebText dataset with 400,000,000 (image, text) pairs of publicly available information from the Internet, is computationally efficient and uses a task-agnostic and zero-shot method that has better results than specific supervised models.

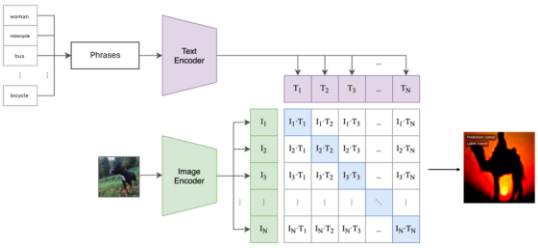


Figure 8 CLIP Model

However, one of the problems with this architecture is that most datasets, including CIFAR-100, are not suitable to this model because it needs phrases instead of classes. Therefore, an auxiliary LLM model will be required to describe these classes.

8.2. LLM Model

To provide phrases for the CLIP model, we have two alternatives: using the templates provided by the CLIP model repository or using an LLM model to generate descriptions from the classes.

Both alternatives can lead to three possible combinations:

- Using different templates for each of the classes.

- Using the LLM model to obtain descriptions of the classes based on prompts (instructions).

- Combining the templates with the descriptions given by the LLM model.

For the creation of descriptions, two pre-trained models are used:

- On the one hand, "*gpt-3.5-turbo-instruct*", belonging to the GPT-3.5 family, can generate natural language or code. However, it has the limitation of being a paid service with interaction limits, so results provided by another developer had to be used.

- On the other hand, "*EleutherAI/gpt-j-6b*" transformer model trained using Ben Wang's Mesh Transformer JAX. "GPT-J" refers to the class of model, while "6B" represents the number of trainable parameters.

One or the other model will be chosen based on the results obtained when using them with the different tested CLIP models.

8.3. Model Results

Before applying the model to our real use case, we evaluate it by obtaining images with the predicted and correct class to calculate the metrics.

It is important to note that we have evaluated different models using the following text encoder approaches: OpenAI's provided templates, the outputs from the two LLM models when given prompts, and both the templates and the outputs from the two LLM models.

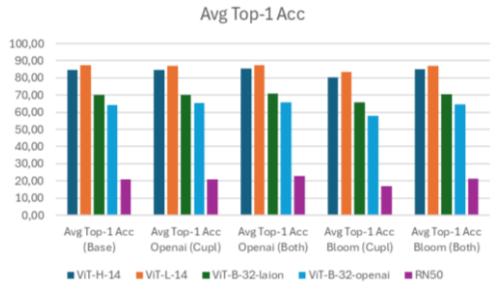


Figure 9 Top-1 Accuracy

Regarding top-1 accuracy, we can observe that for all the mentioned approaches, the model that achieves the best results is VIT-L-14. Its accuracy is almost invariant to differences in text encoder inputs, although slightly better results are obtained when using the *gpt-3.5-turbo-instruct* model along with the templates.

The next best-performing model with the same approach is VIT-H-14, followed by VIT-B-32.

Lastly, it's noteworthy that the RN50 model shows poor results.

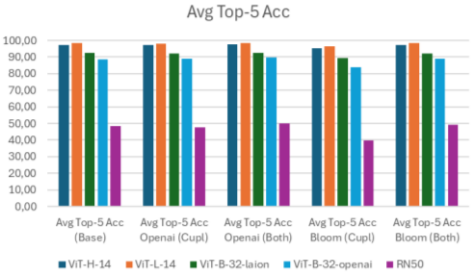


Figure 10 Top-5 Accuracy

On the other hand, the top-5 accuracy results are consistent and significantly better because the search is extended to the five classes that the model predicts with the highest probability.

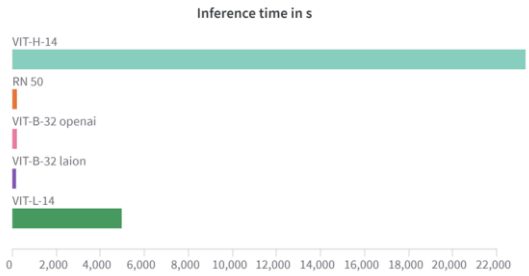


Figure 11 Inference Time

If we compare the processing time of the different models, we can see a significant difference in the time required to process all the data from the test set. The VIT-H-14 and VIT-L-14 models have a much longer inference time than the others, by several orders of magnitude. However, it's also worth noting that the best accuracy results are obtained with the VIT-L-14 model, which has a much shorter processing time than the VIT-H-14.

In conclusion, the results indicate that the VIT-L-14 model, despite having a high inference time, delivers much more robust results compared to the next best model in terms of accuracy and with a shorter inference time.

9. Results

To apply the CLIP model with the result of object detection and transmission via MQTT, the slower processing speed of the algorithm needs to be adapted to the transmission of objects in movement. While threads and queues could be used, since first ones have race conditions and the latter ones have a fixed capacity, it was decided to split the subscription to the topic (the reception of messages) and the classification of these messages into two separate files using an intermediate shared folder (a possible extension could involve using a database) where the subscriber stores the sent images and the algorithm reads them to classify them at a different processing rate.

Additionally, the only output considered from the algorithm is the class with the highest assigned probability.

As final structure, we will have an object detection model for each installation, which communicates via MQTT with the classification model (along with the LLM).

In summary, this project compares three object detection methods: Frame Differencing (simple, fast, and efficient, but prone to generating false detections due to noise), Optical Flow (more accurate but requires high computational resources), and Background Subtraction (more complex but with fewer false positives).

The Mosquitto broker is used for MQTT data transmission between installations and the Control Center due to its low latency and scalability, using Docker Compose and TLS.

Regarding the Deep Learning models used, the VIT-L-14 model, despite having a high processing time, has much more solid results compared to the next best model with a shorter processing time in terms of accuracy (VIT-B-32). On the other hand, it is also notable that the VIT-H-14 model has a significant difference in processing time, making it impractical for low-latency requirements. Additionally, the RN50 model, despite being a reference in image classification, shows very poor results for this project.

The main issue is that when detecting objects, the image to be classified has fewer pixels, which qualitatively reduces its resolution and leads to unreliable results.

To address this problem, techniques for enhancing image quality (super-resolution) could be used before passing them through the classification model, increasing the resolution of the images, or using an ensemble method that combines the predictions of multiple models.

10. Conclusions

In this project, recordings from thermal cameras were used to develop and implement the following:

1. An object detection model: Techniques such as Frame Differencing, Optical Flow, and Background Subtraction (using KNN and MOG2) were compared, ultimately selecting Background Subtraction with MOG2.

2. Docker Compose with the MQTT broker, including bind mounts and security.

3. MQTT communication using the topic "ObjectDetected": The Background Subtraction models act as publishers, while the CLIP model acts as a subscriber.

4. LLM models for obtaining descriptions of CIFAR-100 classes to feed the text encoder of the CLIP model.

5. CLIP image classification models using different image encoders and feeding the text encoder with the results from the LLMs.

6. Application of the results from the best LLM and CLIP model for predicting objects detected by Background Subtraction, using an intermediate folder to decouple the reception of messages from their processing by the CLIP model.

7. Use of Wandb for result analysis.

11. Future Work

As a major potential improvement, which was attempted but not achieved, is the integration of an improved CLIP model through fine-tuning techniques. A code was generated to fine-tune the projections of the text encoder and the image encoder into a common latent space, incorporating techniques to implement early stopping and avoid overfitting. Additionally, given that we have several descriptions for each class, when selecting a (class, image) pair, a random description would be chosen as input to the text encoder.

The main problem with this implementation was that very high loss values were obtained in both training and validation (higher than 40%), leading to its abandonment.

Regarding the Background Subtraction method, the following could be implemented:

- In foreground detection, normalization or spatial filters could be used to reduce noise, while in post-processing, methods for analyzing the foreground mask based on the region or even using information from previous frames could be employed.

- The learning rate for background maintenance, although fixed in our case, could be dynamically adjusted.

- Incorporating a set of counters to set a temporal limit on the duration of a pixel in the foreground.

Additionally, web data processing could be used instead of the CIFAR-100 database.

References

[1] F. Bahri and N. Ray, "Dynamic Background Subtraction by Generative Neural Networks," in *2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2022, pp. 1–8, doi: 10.1109/AVSS56176.2022.9959543.

[2] T. Bouwmans, "Traditional Approaches in Background Modeling."

[3] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Computer Science Review*, vol. 11-12, pp. 31–66, 2014, doi: 10.1016/j.cosrev.2014.04.001.

[4] T. Bouwmans, F. Porikli, B. Höferlin, and A. Vacavant, "Evaluation of Background Models with Synthetic and Real Data," in *Background Modeling and Foreground Detection for Video Surveillance*, Chapman and Hall/CRC, 2014, pp. 601–615, doi: 10.1201/b17223-35.

[5] O. Elharrouss, N. Almaadeed, and S. Al-Maadeed, "A review of video surveillance systems," *Journal of Visual Communication and Image Representation*, vol. 77, p. 103116, 2021, doi: 10.1016/j.jvcir.2021.103116.

[6] V. Fernández-Carbajales, M. Á. G. García, and J. M. Martínez, "Robust People Detection by Fusion of Evidence from Multiple Methods," in *2008 Ninth International Workshop on Image Analysis for Multimedia Interactive Services*, 2008, pp. 55–58, doi: 10.1109/WIAMIS.2008.8.

[7] A. Garcia-Martin and J. M. Martinez, "Robust Real Time Moving People Detection in Surveillance Scenarios," in *2010 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2010, pp. 241–247, doi: 10.1109/AVSS.2010.33.

[8] T. Hamada, T. Minematsu, A. Simada, F. Okubo, and Y. Taniguchi, "Background Subtraction Network Module Ensemble for Background Scene Adaptation," in *2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2022, pp. 1–8, doi: 10.1109/AVSS56176.2022.9959316.

[9] J. Zhou and J. Hoang, "Real Time Robust Human Detection and Tracking System," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, vol. 3, 2005, pp. 149–149, doi: 10.1109/CVPR.2005.517.

[10] M. Paul, S. M. E. Haque, and S. Chakraborty, "Human detection in surveillance videos and its applications—A review," *EURASIP Journal on Advances in Signal Processing*, vol. 2013, no. 1, p. 176, 2013, doi: 10.1186/1687-6180-2013-176.

[11] A. Radford et al., "Transferable Visual Models From Natural Language Supervision," *arXiv preprint arXiv:2103.00020*, 2021, doi: 10.48550/arXiv.2103.00020.

[12] N. Quan, "Difference Between Computer Vision and Image Processing," Eastgate Software, November 2023. [Online]. Available at eastgate.

[13] R. Rajab Asaad, R. Ismael Ali, Z. Arif Ali, and A. Ahmad Shaaban, "Image Processing with Python Libraries," *Academic Journal of Nawroz University*,

vol. 12, no. 2, pp. 410–416, 2023, doi: 10.25007/ajnu.v12n2a1754.

[14] E. Şengönül, R. Samet, Q. Abu Al-Haija, A. Alqahtani, B. Alturki, and A. A. Alsulami, "An Analysis of Artificial Intelligence Techniques in Surveillance Video Anomaly Detection: A Comprehensive Survey," *Applied Sciences*, vol. 13, no. 8, p. 4956, 2023, doi: 10.3390/app13084956.

[15] A. Shimada, Y. Nonaka, H. Nagahara, and R. Taniguchi, "Case-based background modeling: Associative background database towards low-cost and high-performance change detection," *Machine Vision and Applications*, vol. 25, no. 5, pp. 1121–1131, 2014, doi: 10.1007/s00138-013-0563-4.

[16] GitHub, "TFM Repository," https://github.com/201811017/TFM (accessed Jul. 13, 2024).

[17] W&B, "W&B Report: Model comparison," https://api.wandb.ai/links/deeplearningcai/npu6jp5p (accessed Jul. 13, 2024).

[18] OpenAI, "OpenAI Platform - Models Documentation," https://platform.openai.com/docs/models (accessed Jul. 13, 2024).

[19] Hugging Face, "EleutherAI/gpt-j-6b," https://huggingface.co/EleutherAI/gpt-j-6b (accessed Jul. 13, 2024).

[20] W&B, "W&B Report: Predicting detected objects," https://api.wandb.ai/links/deeplearningcai/cbkjhokg (accessed Jul. 13, 2024).

DISEÑO, DESARROLLO Y CONSTRUCCIÓN DE UN SISTEMA DE IDENTIFICACIÓN DE EVENTOS SOBRE EL PROCESADO DE IMÁGENES DE DIFERENTE NATURALEZA

Autor: López-Chaves Estévez, María.
Director: Domínguez Adán, Emilio Manuel.
Entidad Colaboradora: ICAI – Universidad Pontificia Comillas.

RESUMEN DEL PROYECTO

En un contexto donde la seguridad física de las instalaciones industriales es cada vez más importante, la necesidad de una solución automatizada de detección e identificación de intrusos es crucial.

Este trabajo presenta un enfoque escalable y reproducible para identificar intrusos en instalaciones industriales mediante la detección de objetos en vídeos de cámaras de videovigilancia, la transmisión de estas a un clúster centralizado y el empleo de diversos modelos de aprendizaje profundo supervisado dentro del marco CLIP (Contrastive Language-Image Pre-Training), complementados con modelos lingüísticos de gran tamaño (LLM).

El sistema detecta eficazmente objetos en grabaciones, envía estas detecciones mediante MQTT (Message Queuing Telemetry Transport) y las clasifica con el uso de varios algoritmos.

Palabras clave: Seguridad Industrial, Detección de Objetos, Identificación de Intrusos, MQTT, Framework CLIP, LLM.

1. Introducción

Las infraestructuras críticas, definidas como elementos esenciales que sustentan las funciones vitales de una sociedad, han evolucionado hacia activos cuya seguridad presenta constantes desafíos. Esto obliga a una reconsideración de los paradigmas tradicionales para adaptarse a un entorno dinámico y complejo.

La situación actual está marcada por la omnipresencia de tecnologías sofisticadas que no sólo mejoran la eficiencia y la conectividad, sino que también aumentan la superficie de ataque al existir una red compleja de interdependencias. En este entorno dinámico y complejo, la automatización de la detección de intrusos emerge como un pilar esencial para identificar de forma rápida y precisa cualquier intento de intrusión ya que la respuesta humana, aunque valiosa, puede verse limitada por diferentes factores como la velocidad de reacción y la necesidad de atención constante. Cada vez existen más de estos sistemas, aunque se siguen teniendo problemas de falta de eliminación de falsas alarmas y de identificación precisa de objetos, especialmente bajo condiciones ambientales adversas y cambiantes.

Con la aplicación de algoritmos se pretende redefinir la respuesta al aumentar la eficiencia y reducir la carga de trabajo. Tras su trabajo, en caso de ser necesario, se emplearían las medidas de seguridad humanas necesarias.

En este proyecto, se automatizará la detección de intrusos usando cámaras térmicas (tecnología que ha demostrado su conveniencia en entornos complejos) de las instalaciones de una empresa del sector eléctrico. Este enfoque tiene como objetivo proporcionar un sistema de vigilancia automatizado e inteligente, capaz de ofrecer una herramienta más efectiva para la protección de instalaciones críticas que además pueda ser extrapolable a otros ámbitos, como la seguridad residencial.

2. Estado del Arte

El desarrollo e implementación de sistemas de videovigilancia ha avanzado significativamente en las últimas décadas, evolucionando desde soluciones manuales con altos costes de recursos humanos hacia sistemas complejos que se basan en la automatización.

Históricamente, la visión artificial (CV, Computer Vision) dependía de la creación manual de puntos de referencia. La introducción de Machine Learning (ML) revolucionó este campo, eliminando la necesidad de definir manualmente estas reglas gracias al uso de "features" que detectan patrones específicos en las imágenes, pero cuya creación es manual. Los algoritmos de Deep Learning (DL) extraen estos patrones por sí mismos y los convierten en ecuaciones matemáticas, requiriendo solo una gran cantidad de datos etiquetados. Además, la posibilidad de reutilizar redes preentrenadas (TL, Transfer Learning), desarrolladas por grandes compañías aporta una solución a las limitaciones hardware de los usuarios individuales.

Por un lado, para poder procesar imágenes, se han creado numerosas librerías especializadas (Scikit-Image, Numpy, PIL, Mahotas, etc.), de entre las que destaca OpenCV al presentar un alto rendimiento y ser adecuado para tareas complejas.

Por otro lado, existen una gran variedad de algoritmos para la detección automática de personas que se caracterizan según el sistema de adquisición, el número de potenciales sujetos seguidos y su categorización. Por ejemplo, hay algoritmos que se centran en información sobre la apariencia física añadiendo la adquirida mediante algoritmos de seguimiento mientras que hay otros que consiguen reducir la complejidad y ser independientes de la variabilidad de la apariencia al basarse en la información dada por el movimiento, pero con los que se obtienen ligeramente peores resultados.

Estos últimos sistemas automatizados se llaman SVAD (Detección de Anomalías en Vídeos de Vigilancia) y se basan en analizar los fotogramas o frames a nivel de píxel para identificar desviaciones a nivel temporal y espacial respecto a comportamientos habituales. Este proceso se divide en:

1. Identificación de objetos en movimiento mediante Frame differencing (comparación de frames), Optical Flow (campo de vectores 2D que expresa el movimiento entre frames y que puede ser sparse si solamente analiza un número limitado de features o dense si usa todos los píxeles) o Background substraction (separar el fondo o background de los objetos detectados o foreground).

2. Clasificación de objetos mediante algoritmos de ML (Random Forest Classifier , KNN o Naive Bayes) o DL con las Convolutional Neural Networks (CNN) donde hay que

destacar algunos modelos ya preentrenados como AlexNet, VGGNet, GoogLeNet/Inception, Resnet, etc.

2.1. Tecnologías

2.1.1. MQTT

Como protocolo de comunicación se utiliza MQTT, empleado para la comunicación entre equipos. Se aplica sobre TCP (Protocolo de Control de Transmisión) y es ampliamente usado debido a su simplicidad, eficiencia, baja latencia y escaso consumo de recursos.

Utiliza una metodología publicador/subscriptor dónde hay clientes que pueden ser publicadores, subscriptores o los dos a la vez. Cuenta con un broker o servidor que recibe los mensajes de los clientes, los filtra y envía a los adecuados en función del topic.

Se pueden especificar diferentes niveles de niveles de Calidad de Servicio (QoS): 0, sin confirmación de recepción, 1 que no asegura que no haya duplicados y 2 que garantiza que el mensaje llega solamente una vez.

2.1.2. Docker

Docker es una plataforma abierta que se utiliza para desarrollar y distribuir aplicaciones. Las aplicaciones creadas con Docker se empaquetan con todas las dependencias necesarias en un formato estándar y ligero llamado contenedor. Estas aplicaciones virtualizadas se pueden ejecutar en cualquier lugar sin necesidad de modificarlas.

2.1.3. Modelo CLIP

Modelo de DL caracterizado por una arquitectura con dos codificadores y aprendizaje contrastivo para mapear imágenes y texto en un espacio latente compartido

Por un lado, como codificador de imágenes se pueden utilizar diferentes modelos (normalmente basados en CNN o Visual Transformers, ViT) para convertir una imagen preprocesada en una representación vectorial de un espacio dimensional fijo. Se han utilizado dos familias de modelos: Resnet50 (CNN de 50 capas que se basa en conexiones residuales) y ViT (arquitectura de Transformer sobre patches o trozos de una imagen) aunque podría usarse cualquiera de los modelos ya preentrenados y ampliamente utilizados en la clasificación de objetos, pero eliminando la última capa.

Por otro lado, el codificador de texto transforma el significado semántico de las descripciones textuales al mismo espacio latente que la salida del codificador de imágenes. Se suelen utilizar arquitecturas basadas en transformadores.

Como punto de unión entre ambos, usamos el espacio latente compartido en el que los embeddings producidos por los dos codificadores se comparan en un espacio vectorial común usando la similitud del coseno después de haber pasado por una capa de normalización.

3. Arquitectura existente

Este proyecto se basa en la arquitectura existente de una empresa del sector eléctrico que presenta los siguientes elementos:

- Una red de cámaras térmicas que vigilan el exterior de las diferentes instalaciones y se comunican con un Centro de Control mediante una VPN.

- Sensores que permiten la detección precisa y que pueden ser de movimiento o láser.

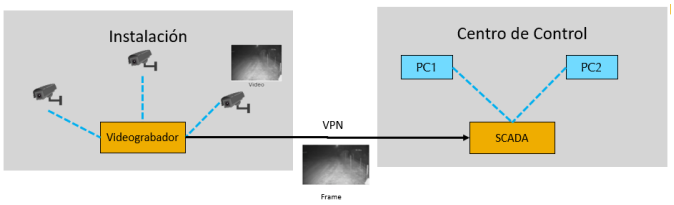


Figura 1 Arquitectura existente.

Cada instalación tiene un videograbador al que le llegan los vídeos de las diferentes cámaras. Estas señales son transmitidas a través de una VPN hacia una plataforma SCADA a la que se accede desde el Centro de Control.

El principal problema que presenta dicha infraestructura es la falta de automatización al requerir una alta carga de trabajo manual por parte de los operarios del Centro de Control.

La solución desarrollada en este proyecto pretende mejorar la eficacia y disminuir la carga de trabajo a la vez que se consigue una mayor precisión.

4. Visión de la solución

El primer paso para el análisis de las grabaciones de las cámaras de vigilancia consiste en la detección de objectos en los vídeos, lo que presenta problemas relacionados con el ruido. Una vez se ha detectado el objeto, queremos identificarlo. Para ello se ha decidido utilizar algoritmos de clasificación de DL por la posibilidad que ofrecen, al además poder utilizar modelos de TL, para reconocer patrones complejos.

Los datos reales con los que trabajamos son grabaciones, en blanco y negro, de las cámaras de seguridad del exterior de diferentes instalaciones

Como suponemos que las comunicaciones entre las instalaciones y el Centro de Control son fiables y dotadas de una latencia muy contenida (inferior al segundo en todos los casos), enviaremos los objetos detectados mediante el algoritmo de detección de movimiento a través de MQTT a otro algoritmo que permitirá clasificarlos, Si hubiese malas comunicaciones, en cada instalación estaríamos detectando y clasificando objetos y sólo enviaríamos las alarmas.

En lo referente al protocolo de comunicación, se ha decidido usar MQTT por su baja latencia y comunicación asíncrona, por su escalabilidad, además de por la posibilidad de definir diferentes QoS y de utilizar TLS (Seguridad de la Capa de Transporte).

Respecto al algoritmo de DL, se ha usado el modelo CLIP que supera la limitación de predecir un número fijo de categorías además de contar con capacidad de realizar una clasificación zero-shot ya que, debido a la falta de diversidad en nuestros datos, la posibilidad de usar técnicas de aumento de ellos fue rápidamente eliminada.

Un esquema de la solución desarrollada se presenta a continuación:

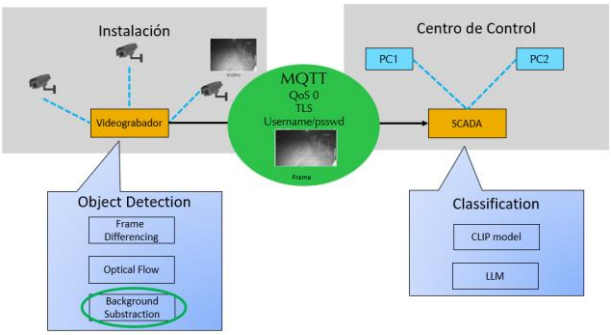


Figura 2 Solución desarrollada.

5. Detección de intrusos

La detección de objetos consiste en localizar los elementos que se mueven dentro de una secuencia de un vídeo.

Para ello se ha decidido comparar la eficacia de tres métodos diferentes de detección de objetos y con uno de ellos utilizando tanto MOG2 como KNN.

Por un lado, el modelo MOG2 pertenece a la familia de Mezcla de Gaussianas que consiste en un enfoque estadístico para representar la distribución de probabilidad de datos que podrían ser generados por múltiples curvas.

Por otro lado, el modelo KNN se basa en la implementación del algoritmo K-Nearest Neighbors en CV.

Como se podrá observar en los siguientes subapartados, Background Substraction con MOG2 es la solución que mejor se adapta por presentar menos ruido, poder detectar cambios graduales y ser menos costosa computacionalmente.

5.1. Frame Differencing

Este método consiste en sustraer el frame actual del video del anterior y anotar las diferencias ya que se supone que estas corresponden al movimiento (máscara de movimiento o motion mask).

Se comienza calculando la diferencia entre frames en escala de grises, pero esto no es suficiente ya que se obtienen algunas manchas. Para eliminarlas se calcula la motion mask, una imagen binaria cuyos píxeles estarán en blanco si forman parte del foreground y en negro si son del background o fondo. Se clasificación como foreground si la diferencia entre píxeles es mayor que un umbral definido como adaptativo.

El siguiente paso consiste en usar detectores de objetos para guardar los puntos dados por el algoritmo de aproximación de cadenas Teh-Chinu sobre los contornos en esta máscara si sus valores son mayores que un umbral.

Por último, postprocesamos lo proporcionado por lo detectores de objetos usando el algoritmo de Supresión No Máxima (NMS) para filtrar estas predicciones al eliminar las que sean poco probables de contener a un objeto en movimiento.

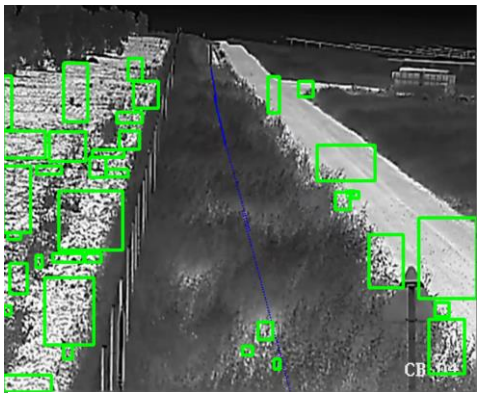


Figura 3 Frame Differencing

5.2. Optical Flow

Consiste en la determinación del campo de velocidades bidimensional que se obtiene al proyectar las velocidades tridimensionales en el plano de la imagen.

Esto resulta en un vector de desplazamiento que explica el movimiento de los puntos entre un frame y el siguiente.

Si usamos un método sparse como Lukas Kanade, dónde detectamos las características más distintivas de una imagen, solamente seguiremos a estos puntos entre frames consecutivos para visualizar el movimiento. Si en cambio usamos un método dense, concretamente Gunnar Farneback, tenemos en cuenta todos los píxeles entre dos frames.

A continuación, se buscan los contornos a la vez que se utiliza el ángulo del flujo para filtrar las detecciones que tienen poca probabilidad de ser correctas.

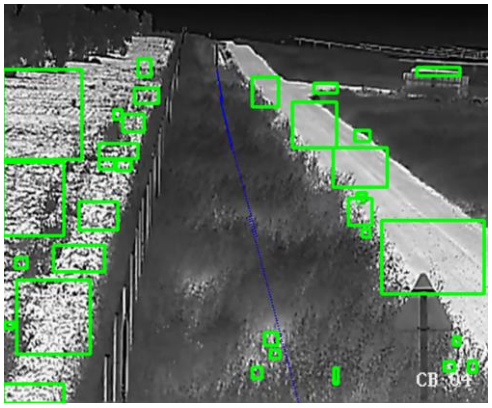


Figura 4 Dense Optical Flow

6. Background Subtraction

Se basa en calcular la máscara del foreground restando el frame actual del background dinámico y aplicando un umbral a esta resta según los siguientes pasos:

1. Inicialización del background. En nuestro caso es progresivo ya que se va parcialmente generando.

2. Actualización del background y detección del foreground. Se basa en comparar la imagen actual con la del background para clasificar los píxeles como background o foreground. Tiene los siguientes pasos:

    a. Preprocesado para evitar detectar cambios no importantes usando ajustes geométricos y de intensidad.

    b. Clasificación de pixeles en background o foreground dependiendo de si la diferencia de la imagen actual respecto a la del background es superior a un umbral (si es mayor, formaría parte del foreground).

    c. Postprocesado para aumentar la consistencia de la máscara de foreground utilizando operaciones morfológicas y difuminando la motion mask.

3. Mantenimiento del background selectivo, es decir, con una learning rate mucho mayor a un píxel clasificado como background que a uno que es foreground.

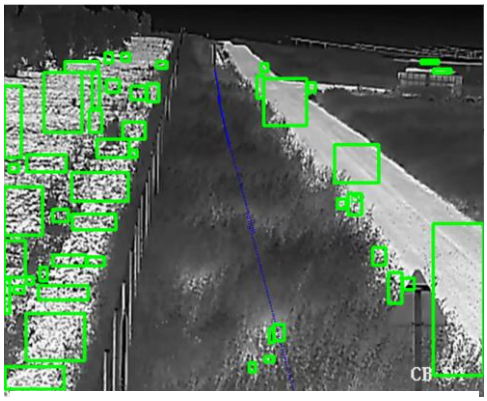


Figura 5 BS-KNN

Dado que los cambios que se producirán en el background serán graduales y como no tenemos pocos píxeles clasificados como foreground, utilizaremos MOG2 en vez de KNN.

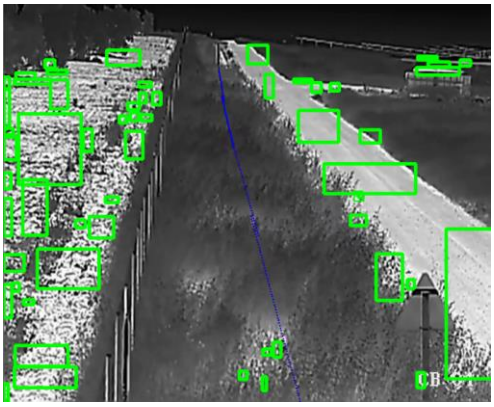


Figura 5 BS-MOG2

7. MQTT

Para desarrollar este proyecto se ha usado un bróker de código abierto: Mosquitto. Esta elección se ha basado en su completa documentación, su gran comunidad de desarrolladores y su poca consumición de recursos.

Para crear este servidor se usará Docker Compose ya que esto facilita la escalabilidad al poder añadir múltiples aplicaciones o servicios, poder usar volúmenes o bind mounts para almacenar datos persistentes y establecer un orden de inicio de funcionamiento de los diferentes contenedores.

Para ello crearemos un Docker Compose que utiliza el bróker Mosquitto cuyos puertos son configurables, pero hemos escogido el 1883 (puerto por defecto) y 8883 (TLS) los cuales están mapeados con los correspondientes del contenedor. Se reinicia automáticamente y pertenece a la red llamada “my network”.

Para tener un canal de comunicaciones seguro usaremos TLS. Para ello generaremos un certificado de la autoridad certificadora (CA) autofirmado y, a continuación, otros tantos para el servidor y para dos clientes con el algoritmo hash criptográfico SHA-256 usando una clave privada de 2048 bits.

Además, se usan cuatro bind mounts: Config (configuración de funcionamiento del bróker), Certs (guarda los certificados de la CA y del servidor), Data (base de datos en memoria) y Log (almacena eventos).

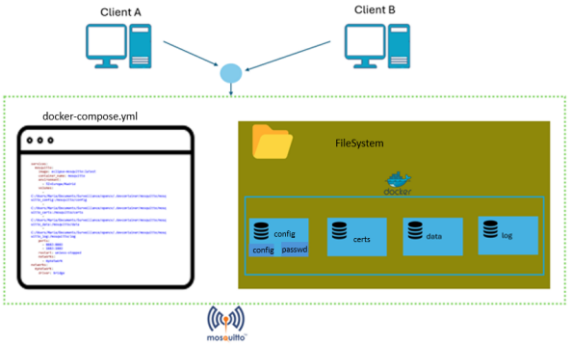


Figura 6 Arquitectura MQTT

Usaremos el propio comando *mosquitto_passwd* para crear las contraseñas que se guardarán en un archivo con el mismo nombre y referenciado en *mosquitto_config*.

Los clientes enviarán y recibirán los mensajes bajo un único topic llamado “ObjectDetected” (estamos simplificando la solución a una única instalación) y estos clientes pueden ser:

- Publicador (el algoritmo de detección de objetos que utiliza Background Substraction con QoS 0 para tener mínima latencia)
- El modelo CLIP como suscriptor.

8. Clasificación de intrusos

Necesitaremos una base de datos etiquetada para poder comparar los diferentes modelos de TL con métricas cuantificables. En nuestro caso usaremos CIFAR-100, un dataset ampliamente utilizado para tareas de clasificación. Contiene 60,000 imágenes a color de 32x32 píxeles agrupadas en 100 clases y

estas a su vez en 20 superclases. Hay unas 600 imágenes por cada clase (500 de entrenamiento y 100 de test) y cada una de ellas viene con una etiqueta "fine" que representa la clase a la que pertenece y una etiqueta "coarse" que representa la superclase.

8.1. Modelo CLIP

El modelo de DL CLIP, al haber sido entrenado en el WebText dataset de 400.000.000 pares de (imagen, texto) de información pública de Internet, es computacionalmente eficiente además que utiliza un método task-agnostic y zero-shot con mejores resultados que modelos supervisados específicos.

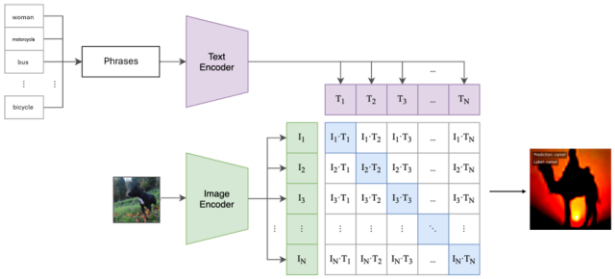


Figura 7 Modelo CLIP.

No obstante, uno de los problemas que presenta esta arquitectura es que la mayor parte de los datasets, incluyendo CIFAR-100, no son adecuados ya que, debido a que se necesita suplir al modelo con frases en vez de clases, se necesitará un modelo auxiliar LLM que describa a estas clases.

8.2. Modelo LLM

Para proveer al modelo CLIP de frases tenemos dos alternativas: usar las plantillas que proporciona el propio repositorio del modelo CLIP o utilizar un modelo LLM que genere descripciones a partir de las clases.

Ambas alternativas pueden dar lugar a tres combinaciones posibles:

- Utilizar las diferentes plantillas para cada una de las clases.
- Usar el modelo LLM para obtener descripciones de las clases en función de prompts o instrucciones.
- Combinar las plantillas junto a las descripciones dadas por el modelo LLM.

Para la creación de descripciones se utilizan dos modelos ya preentrenados.

Por un lado, "*gpt-3.5-turbo-instruct*", perteneciente a la familia GPT-3.5, y que puede generar lenguaje natural o código, aunque presenta el problema de que, al ser un servicio de pago, tiene un límite de interacciones, por lo que se han tenido que utilizar los resultados proporcionados por otro desarrollador.

Por otro lado, "*EleutherAI/gpt-j-6b*", un Transformer clase "GPT-J" entrenado con Mesh Transformer JAX de Ben Wang y que tiene 6B de parámetros.

Se escogerá uno u otro modelo en función de los resultados que se obtengan al utilizarlos con los diferentes modelos CLIP probados.

8.3. Resultados de los modelos

El modelo (antes de aplicarlo a nuestro caso de uso real) nos dará como resultado imágenes con la predicción y la clase correcta ya que queremos calcular las métricas.

Se ha de tener en cuenta que se han evaluado los diferentes modelos usando los siguientes enfoques del codificador de texto: las plantillas dadas por OpenAI, los resultados de los dos modelos LLM al pasarle las prompts y tanto las plantillas como los resultados de los dos modelos LLM.

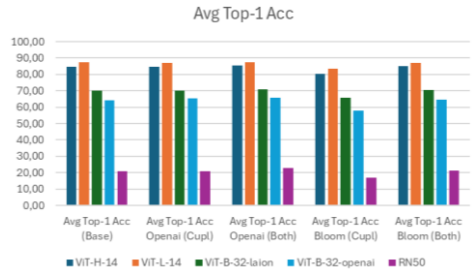


Figura 8 Top-1 Accuracy

En lo referente a la top-1 accuracy, podemos observar que, para todos los anteriores enfoques, el modelo que obtiene mejores resultados es el VIT-L-14. Su accuracy es prácticamente invariante a las diferencias en las entradas del codificador de texto, a pesar de que se obtienen ligeramente mejores resultados cuando se usa el modelo *gpt-3.5-turbo-instruct* junto a las plantillas.

El siguiente modelo que da mejores resultados y además con el mismo enfoque es VIT-H-14 y después VIT-B-32. Por último, es de resaltar los pobres resultados del RN50.

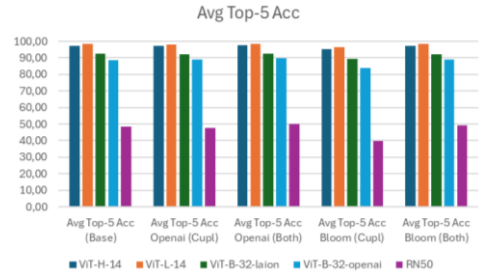


Figura 9 Top-5 Accuracy.

Por otro lado, los resultados con la top-5 accuracy son consistentes, aunque mucho mejores por ampliar la búsqueda a las cinco clases que el modelo predice con mayor probabilidad.

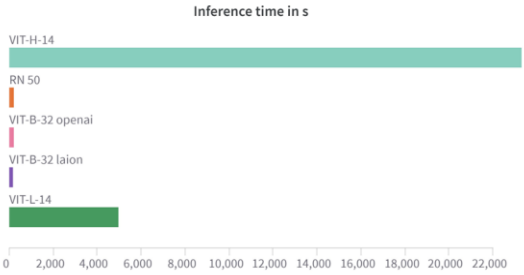


Figura 10 Tiempo de procesamiento.

Si comparamos el tiempo de procesamiento de los diferentes modelos, podemos observar que, para poder procesar todos los datos del conjunto de test, hay una gran diferencia entre los modelos Vit-H-14 y VIT-L-14 y el resto de varios órdenes de magnitud. No obstante, también es interesante comentar que los mejores resultados a nivel de accuracy se obtienen con el modelo VIT-L-14, el cual tiene un tiempo de procesamiento infinitamente menor que el VIT-H-14.

Como conclusión, los resultados indican que el modelo VIT-L-14, a pesar de tener un elevado tiempo de procesamiento, presenta resultados mucho más sólidos que el siguiente modelo con mejor accuracy y con menor tiempo de procesamiento.

9. Resultados

Para poder aplicar el modelo CLIP a la detección de objetos y envío por MQTT, se ha de adaptar la menor velocidad del procesamiento del algoritmo al envío de objetos detectados. Se podrían usar hilos y colas, pero, debido a las condiciones de carrera de las primeras y a que las últimas tienen una capacidad fija, se ha decidido separar en dos ficheros la suscripción al topic (la llegada de mensajes) y la

clasificación de estos mensajes mediante el uso de una carpeta intermedia y compartida (una posible ampliación sería usar una base de datos) en el que el suscriptor almacena las imágenes enviadas y el algoritmo, a un ritmo diferente de procesamiento, iría clasificándolas.

Además, sólo se entiende como resultado del algoritmo la clase que tiene la mayor probabilidad asignada.

Como estructura final tendremos un modelo de detección de objetos por cada una de las instalaciones que se comunica por MQTT con el modelo de clasificación (junto al LLM) localizado en el Centro de Control.

En resumen, en este proyecto se comparan tres métodos de detección de objetos: Frame Differencing (sencillo, rápido y eficiente, pero propenso a generar falsas detecciones con ruido), Optical Flow (más preciso, pero requiere de elevados recursos computacionales) y Background Subtraction (más complejo, pero con menor cantidad de falsos positivos).

Se utiliza el bróker Mosquitto para la transmisión de datos MQTT entre las instalaciones y el Centro de Control debido a su baja latencia y escalabilidad, utilizando Docker Compose y TLS.

Respecto a los modelos de Deep Learning utilizamos, podemos observar que el modelo VIT-L-14, a pesar de tener un elevado tiempo de procesamiento, presenta resultados mucho más sólidos que el siguiente modelo con mejor accuracy y con menor tiempo de procesamiento (VIT-B-32). Por otro lado, también es notable destacar la gran diferencia de tiempo de procesamiento del modelo VIT-H-14, lo que lo hace poco práctico si se quiere tener una baja latencia, además es importante resaltar que el modelo RN50, a pesar de ser un referente en la clasificación de imágenes, presenta unos resultados muy pobres para este proyecto.

El principal problema que presenta esta solución es que, al detectar objetos, la imagen que se quiere clasificar tiene menos píxeles lo que hace que se reduzca cualitativamente su resolución y que se obtengan resultados poco fiables.

Para abordar este problema se podrían utilizar técnicas de aumento de calidad de las imágenes (de superresolución) antes de pasarlo por el modelo de clasificación, aumentar la propia resolución de las imágenes o usar un ensemble method que combine las predicciones de múltiples modelos.

10. Conclusiones

En este proyecto se ha partido de las grabaciones de unas cámaras térmicas para desarrollar y aplicar lo siguiente:

1. Un modelo de detección de objetos. Se han comparado las técnicas de Frame Differencing, Optical Flow y Background Substraction (con KNN y MOG2) para al final escoger Background Substraction con MOG2.

2. Docker Compose con el bróker mqtt, con bind mounts y securizado.

3. Comunicación MQTT usando el topic “ObjectDetected” y teniendo como publicadores los modelos de Background Substraction y como suscriptor el modelo CLIP.

4. Modelos LLM para la obtención de descripciones de las clases de CIFAR-100 que alimenten al codificador de texto del modelo CLIP.

5. Modelos CLIP de clasificación de imágenes usando diferentes codificadores de imágenes y alimentando el codificador de texto con los resultados de los LLM.

6. Aplicación de los resultados del mejor LLM y modelo CLIP para la predicción de los objetos detectados por Background Substraction usando una carpeta intermedia para desacoplar la recepción de mensajes del procesamiento de estos por parte del modelo CLIP.

7. Uso de Wandb para analizar resultados.

11. Trabajo futuro

Como principal posible mejora, que se ha intentado implementar, pero no se ha conseguido, es la integración de un modelo CLIP mejorado mediante técnicas de fine-tuning. Para ello se ha generado un código en el que se está modificando las proyecciones del codificador del texto y del de imagen en un espacio común además de incorporar técnicas para implementar early stopping y evitar el overfitting. También hay que tener en cuenta que, dado que para cada clase tenemos varias descripciones, al seleccionar un par (clase, imagen), se escogería de manera aleatoria alguna de las descripciones como entrada al codificador de texto.

El principal problema de esta implementación es que se obtienen unos valores de loss muy alta (más de 40%) tanto en training como en validation por lo que se ha decidido descartarlo.

En lo referente al método de Background substraction se podrían implementar:

- En la detección del foreground se podría usar una normalización o filtros espaciales para reducir ruido mientras que en el postprocesado se podrían usar métodos de análisis de la máscara del foreground en función de la región o incluso usar información de frames anteriores.

- Respecto al learning rate del mantenimiento del background, a pesar de que en nuestro caso es fija, podría ser dinámicamente ajustada.

- Incorporar un set de contadores para tener un límite temporal de permanencia de un píxel en el foreground.

Por otro lado, se podría usar el procesamiento de datos web en vez de usar la base de datos CIFAR-100.

Bibliografía

[1] F. Bahri and N. Ray, "Dynamic Background Subtraction by Generative Neural Networks," in *2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2022, pp. 1–8, doi: 10.1109/AVSS56176.2022.9959543.

[2] T. Bouwmans, "Traditional Approaches in Background Modeling."

[3] T. Bouwmans, "Traditional and recent approaches in background modeling for foreground detection: An overview," *Computer Science Review*, vol. 11-12, pp. 31–66, 2014, doi: 10.1016/j.cosrev.2014.04.001.

[4] T. Bouwmans, F. Porikli, B. Höferlin, and A. Vacavant, "Evaluation of Background Models with Synthetic and

Real Data," in *Background Modeling and Foreground Detection for Video Surveillance*, Chapman and Hall/CRC, 2014, pp. 601–615, doi: 10.1201/b17223-35.

[5] O. Elharrouss, N. Almaadeed, and S. Al-Maadeed, "A review of video surveillance systems," *Journal of Visual Communication and Image Representation*, vol. 77, p. 103116, 2021, doi: 10.1016/j.jvcir.2021.103116.

[6] V. Fernández-Carbajales, M. Á. G. García, and J. M. Martínez, "Robust People Detection by Fusion of Evidence from Multiple Methods," in *2008 Ninth International Workshop on Image Analysis for Multimedia Interactive Services*, 2008, pp. 55–58, doi: 10.1109/WIAMIS.2008.8.

[7] A. Garcia-Martin and J. M. Martinez, "Robust Real Time Moving People Detection in Surveillance Scenarios," in *2010 7th IEEE International Conference on Advanced Video and Signal Based Surveillance*, 2010, pp. 241–247, doi: 10.1109/AVSS.2010.33.

[8] T. Hamada, T. Minematsu, A. Simada, F. Okubo, and Y. Taniguchi, "Background Subtraction Network Module Ensemble for Background Scene Adaptation," in *2022 18th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2022, pp. 1–8, doi: 10.1109/AVSS56176.2022.9959316.

[9] J. Zhou and J. Hoang, "Real Time Robust Human Detection and Tracking System," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Workshops*, vol. 3, 2005, pp. 149–149, doi: 10.1109/CVPR.2005.517.

[10] M. Paul, S. M. E. Haque, and S. Chakraborty, "Human detection in surveillance videos and its applications—A review," *EURASIP Journal on Advances in Signal Processing*, vol. 2013, no. 1, p. 176, 2013, doi: 10.1186/1687-6180-2013-176.

[11] A. Radford et al., "Transferable Visual Models From Natural Language Supervision," *arXiv preprint arXiv:2103.00020*, 2021, doi: 10.48550/arXiv.2103.00020.

[12] N. Quan, "Difference Between Computer Vision and Image Processing," Eastgate Software, November 2023. [Online]. Available at eastgate.

[13] R. Rajab Asaad, R. Ismael Ali, Z. Arif Ali, and A. Ahmad Shaaban, "Image Processing with Python Libraries," *Academic Journal of Nawroz University*, vol. 12, no. 2, pp. 410–416, 2023, doi: 10.25007/ajnu.v12n2a1754.

[14] E. Şengönül, R. Samet, Q. Abu Al-Haija, A. Alqahtani, B. Alturki, and A. A. Alsulami, "An Analysis of Artificial Intelligence Techniques in Surveillance Video Anomaly Detection: A Comprehensive Survey," *Applied Sciences*, vol. 13, no. 8, p. 4956, 2023, doi: 10.3390/app13084956.

[15] A. Shimada, Y. Nonaka, H. Nagahara, and R. Taniguchi, "Case-based background modeling: Associative background database towards low-cost and high-performance change detection," *Machine Vision and Applications*, vol. 25, no. 5, pp. 1121–1131, 2014, doi: 10.1007/s00138-013-0563-4

[16] GitHub, "TFM Repository," https://github.com/201811017/TFM (accessed Jul. 13, 2024).

[17] W&B, "W&B Report: Model comparison," https://api.wandb.ai/links/deeplearningicai/npu6jp5p (accessed Jul. 13, 2024).

[18] OpenAI, "OpenAI Platform - Models Documentation," https://platform.openai.com/docs/models (accessed Jul. 13, 2024).

[19] Hugging Face, "EleutherAI/gpt-j-6b," https://huggingface.co/EleutherAI/gpt-j-6b (accessed Jul. 13, 2024).

[20] W&B, "W&B Report: Predicting detected objects," https://api.wandb.ai/links/deeplearningicai/cbkjhokg (accessed Jul. 13, 2024).