Distributed real-time system for video-surveillance

Matteini Mattia
Paganelli Alberto

February 10, 2023

Abstract

Si vuole sviluppare un sistema di video-sorveglianza real-time che permette di controllare da remoto diverse videocamere. Sarà presente una dashboard di controllo dalla quale si potranno vedere tutti i video in tempo reali catturati dalle videocamere. Oltre alla semplice consultazione dei video streaming, vi sarà anche la possibilità di ricavare delle informazioni attraverso un sistema di object-recognition. Questo verrà effettuato da nodi appositamente adibiti. Si potrà fare uso di queste informazioni per poter impostare notifiche di allarme personalizzate.

1 Goals/Requirements

Lo sviluppo di questo progetto prevede le seguenti milestones da raggiungere in maniera incrementale:

- Realizzazione dei nodi che si occupano della generazione di stream video (Producer).
- Realizzazione web server con relativi client (Consumer) grazie ai quali si potrà monitorare il sistema attraverso una dashboard.
- Realizzazione di nodi che consumano lo streaming, lo elaborano ed estraggono features utili per il sistema di monitoraggio.
- Realizzazione di sistemi di allarme personalizzati in base alle informazioni ricavate dai nodi "elaboratori".

2 Architecture

Si prevede di utilizzare il framework Apache Kafka per la gestione degli stream unbounded (essendo real-time non se ne conosce il termine). I nodi verranno quindi dotati di un client Kafka in grado di potersi interfacciare con il cluster Kafka.

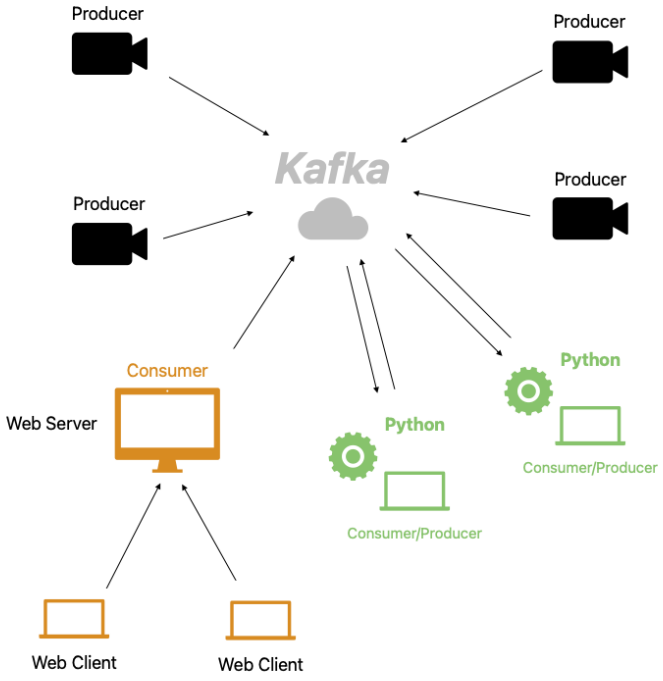

Figure 1: Architettura generale del sistema.

I componenti forniti di videocamera hanno l'unico compito di generare lo stream video catturando ciò che accade nella loro area di competenza, dopodiché lo stream verrà inviato al cluster Kafka. Al cluster si interfacceranno anche i nodi adibiti a fare object-recognition i quali in questo sistema saranno la tipologia di nodo più complesso essendo sia Consumer che Producer. Questo dovuto al fatto che consumeranno lo stream video per poterlo elaborare ed estrarne delle determinate features. Un'altra tipologia di nodo sarà il client, che attraverso una dashboard, visualizzerà in real-time tutti gli stream e le eventuali analitiche ricavate grazie alle features estratte. Attraverso questa tipologia di architettura anche aggiungere eventuali sensori utili al fine di sorveglianza rimane un'azione facilmente percorribile.

2.1 Programming platforms

- **Apache Kafka** per la gestione degli stream di dati.
- **NodeJS** per la realizzazione di:
  - Web Server (Consumer) in grado di interrogare il cluster Kafka e di fornire un interfaccia web all'utente.
  - Nodi (Producer) per generare ed inviare lo stream video.
- Ambiente **Python** per la realizzazione dei nodi "elaboratori", che hanno il compito di estrarre features e informazioni utili dallo streaming video attraverso l'utilizzo di OpenCV.
- **Docker** per il deploy automatico dei nodi del sistema distribuito.

References

[1] Documentazione Apache Kafka, site: `https://kafka.apache.org/`

[2] Documentazione NodeJS, site: `https://nodejs.org/`

[3] OpenCV per riconoscimento oggetti, site: `https://opencv.org/`.