Distributed Scalable Face Recognition System

Dokumentacija

Kolegij: Distribuirani sustavi
Fakultet informatike u Puli

Autor: Antonio Labinjan
Mentor: doc. dr. sc. Nikola Tanković

November 6, 2025

Contents

1. Sažetak

Ova dokumentacija opisuje dizajn i implementaciju distribuiranog, skalabilnog sustava za prepoznavanje lica u realnom vremenu temeljenog na modernim metodama računalnog vida i umjetne inteligencije. Sustav se sastoji od više udaljenih nodeova (edge kamera) koji lokalno izvode ekstrakciju značajki lica koristeći CLIP model, te centralnog servera koji koristi FAISS za vektorsko pretraživanje i klasifikaciju osoba. Arhitektura omogućuje horizontalno skaliranje, load balancing i failover mehanizme između nodeova, čime se postiže visoka pouzdanost i rad u stvarnom vremenu.

Komunikacija između nodeova i servera odvija se putem Redis posrednika, koji služi kao message broker i load balancing sloj. Redis omogućuje asinkronu razmjenu embeddinga, redoslijedno procesiranje zahtjeva i stabilan prijenos podataka čak i u slučajevima privremene mrežne nestabilnosti.

Na strani nodea implementiran je mehanizam shouldClassify() koji optimizira prijenos podataka - embedding se šalje prema serveru samo ako je detekcija dovoljno različita ili ako je prošlo određeno vrijeme od zadnjeg slanja. Time se značajno smanjuje mrežno opterećenje bez gubitka performansi sustava.

Sustav također uključuje unknown alert system koji prepoznaje višestruke pokušaje neuspjele identifikacije s različitih nodeova u kratkom vremenskom periodu, uz potpuno poštivanje GDPR smjernica i bez otkrivanja identiteta korisnika.

Rješenje pokazuje kako se principi distribuiranih sustava mogu učinkovito primijeniti na prepoznavanje lica u stvarnom vremenu, uz naglasak na skalabilnost, efikasnost i privatnost u edge-to-server arhitekturi.

2. Opis aplikacije

Razvijeni sustav predstavlja distribuirano rješenje za prepoznavanje lica u stvarnom vremenu koje kombinira snagu compter visiona i distribuiranih sustava. Sastoji se od više **nodeova** (edge kamera) koji lokalno obrađuju video u stvarnom vremenu, te centralnog **servera** koji objedinjuje rezultate, vrši klasifikaciju i vodi evidenciju detekcija. Također, vrši se dvostruka segmentacija i uklanjanje pozadine i na server-side djelu i na nodevima kako bi se smanjio utjecaj pozadine na klasifikaciju. U trenutnoj verziji, server je implementiran u FastAPI-ju, dok su nodevi implementirani u čistom Pythonu te koriste ugrađene kamere računala i dodatne vanjske kamere (trenutno USB priključak). Nodeovi koriste **CLIP** model za ekstrakciju značajki lica (embeddinga), dok server koristi **FAISS** za vektorsko pretraživanje i usporedbu embeddinga s postojećom bazom poznatih osoba. Komunikacija između nodeova i servera odvija se putem **Redis** posrednika koji služi kao message broker i load balancing sloj, čime se omogućuje skalabilna i asinkrona razmjena podataka.

Sustav podržava horizontalno skaliranje - dodavanjem novih nodeova automatski se povećava kapacitet obrade bez promjene konfiguracije servera. Osim toga, uključeni su **failover mehanizmi** koji omogućuju nastavak rada i u slučaju kvara pojedinog nodea ili pada servera. Nodesi su lako zamjenjivi i lako ih je dodati (potrebno je samo spojiti kameru i pokrenuti Python script). Ukoliko server padne, embeddinzi se svejedno šalju u Redis te mogu biti dohvaćeni kad server opet proradi.

Jedna od prednosti sustava je mehanizam `should_classify()`, koji značajno smanjuje promet u mreži jer osigurava da se embedding šalje samo kad je stvarno potreban (promjena lica ili protek vremena). Time se postiže optimalna ravnoteža između performansi i učinkovitosti.

Ciljano tržište i korisnici

Primarna ciljna skupina su **organizacije, poduzeća i institucije** kojima je potreban automatizirani, brzi i pouzdani sustav za identifikaciju i evidenciju osoba. Sustav se može koristiti u različitim scenarijima:

- **Evidencija prisutnosti zaposlenika** u tvrtkama, laboratorijima ili fakultetima.
- **Sigurnosni nadzor** u poslovnim i javnim prostorima, s mogućnošću detekcije nepoznatih osoba (unknown alert system).
- **Autonomni ulazni sustavi** koji reagiraju na prepoznato lice i odobravaju pristup bez potrebe za fizičkim kontaktom.

Za razliku od centraliziranih rješenja, ovaj sustav pruža **distribuiranost, otpornost i privatnost** – obrada se odvija lokalno na nodeovima, a prema serveru se šalju samo embeddingi, ne i slike lica. To znači da je sustav u skladu s **GDPR** smjernicama i može se primijeniti u okruženjima s visokim sigurnosnim i etičkim zahtjevima. Time se postiže moderna kombinacija umjetne inteligencije i distribuiranih principa koja otvara put prema **lokalnim edge-to-server** rješenjima spremnima za buduće proširenje u cloud okruženje.

Analiza tržišta i konkurencija

Motivacija za razvoj ovog projekta proizašla je iz prethodnog rada na jednostavnim web aplikacijama baziranim na arhitekturi **1-client–1-server**. Takav pristup bio je dovoljan za manje projekte, ali pokazao je svoja ograničenja u situacijama gdje je potrebno obraditi veći broj ulaza

(kamera) i postići real-time sinkronizaciju između više uređaja. Cilj je bio unaprijediti postojeću aplikaciju pretvaranjem je u **distribuirani sustav**, sposoban za obradu podataka s više kamera u stvarnom vremenu uz istovremeno održavanje stabilnosti, točnosti i brzine.

Glavni motiv je bio **automatizirati brojne svakodnevne procese bilježenja prisutnosti** — eliminirati ručni unos, liste i kartice te omogućiti sustavu da sam prepoznaje osobu pri ulasku ili izlasku. Takav pristup ne samo da štedi vrijeme, već i uklanja ljudske pogreške te omogućuje analitiku u stvarnom vremenu.

Na tržištu već postoje sustavi za prepoznavanje lica, poput **Azure Face API-ja**, **AWS Rekognitiona** ili **OpenCV/DeepFace** rješenja, ali svi oni imaju svoja ograničenja:

- **Cloud ovisnost:** većina rješenja zahtijeva konstantnu internetsku vezu i vanjsku infrastrukturu, što stvara ovisnost o trećim servisima.
- **Privatnost:** podaci o licima šalju se na vanjske servere, što je neprihvatljivo u GDPR osjetljivim okruženjima.
- **Centraliziranost:** tradicionalna rješenja koriste jedan server, što otežava skaliranje i povećava rizik od kvara.

Razvijeni sustav uklanja te probleme uvođenjem **distribuirane arhitekture** s lokalnom obradom podataka na **edge nodeovima** i centralnim serverom koji upravlja klasifikacijom i sinkronizacijom putem **Redis** brokera. Time je postignut balans između brzine, sigurnosti i fleksibilnosti, bez ovisnosti o vanjskim servisima.

Table 1: SWOT analiza distribuiranog sustava za prepoznavanje lica

Snage (S)	**Slabosti (W)**
• Distribuiran i skalabilan sustav • Lokalna obrada podataka (GDPR-friendly) • Real-time prepoznavanje i logging • Modularnost i lako dodavanje novih nodeova	• Zahtjevnija implementacija i konfiguracija • Potrebno održavanje više uređaja
Prilike (O)	**Prijetnje (T)**
• Primjena u pametnim zgradama i industriji 4.0 • Integracija s postojećim sustavima kontrole pristupa • Potencijal za komercijalizaciju i SaaS model	• Brz razvoj konkurentskih AI rješenja • Moguće regulatorne promjene (GDPR, AI Act) • Tehnički problemi s pouzdanošću mreže

Projekt time ne samo da rješava konkretan problem evidencije prisutnosti, već otvara vrata prema **budućim edge-to-server sustavima** koji kombiniraju računalni vid, distribuirano računalstvo i automatizaciju poslovnih procesa. Ovo rješenje demonstrira kako se napredni AI modeli mogu učinkovito raspodijeliti između rubnih uređaja i centralnog sustava, što predstavlja važan korak prema modernim, skalabilnim i etičkim sustavima prepoznavanja lica.

1 4. Razrada funkcionalnosti *(max. 8–15 stranica)*

1. Funkcionalnosti

- Pokretanje FastAPI servera s REST API endpointima
- Učitavanje dataset-a lica iz foldera

- Dodavanje pojedinačnih lica u dataset i ekstrakcija CLIP embeddinga
- Normalizacija i spremanje embeddinga u memoriju
- Gradnja FAISS indeksa za brzu pretragu najbližih susjeda
- Klasifikacija lica pomoću FAISS indeksa i CLIP embeddinga
- Testiranje različitih threshold vrijednosti za klasifikaciju
- Praćenje neprepoznatih pokušaja (Unknown) i intruder alert sustav
- Logiranje detekcija u memoriju
- Globalno logiranje preko Redis streama
- Redis queue za slanje embeddinga u worker thread
- Worker thread koji čita embeddinge iz Redis queue-a i klasificira ih
- Dead-letter queue u Redisu za neispravne ili višestruko neuspjele poruke
- Pametno odlučivanje kada klasificirati embedding po node-u (`should_classify`)
- Praćenje posljednjeg embeddinga i timestamp-a po node-u
- API endpointi za pregled loga detekcija u JSON i HTML formatu
- API endpointi za pregled queue-a, threshold statistike i aktivnih node-ova
- Vizualizacija threshold statistike i aktivnih node-ova u HTML tablicama
- API endpointi za intruder alert log u JSON i HTML formatu
- Endpoint za ponovno učitavanje dataset-a i rebuild FAISS indeksa
- Ping endpoint za provjeru zdravlja servera
- Logging svih važnih operacija s vremenskim oznakama

Funkcionalnosti Dijagrami (use case, class) Prototip

2 5. Implementacija *(max. 3–5 stranica)*

Kako je isprogramirano Graf arhitekture Detaljno objasnit Koje su klase i komponente unutar aplikacije Docker compose => zašto (screenovi) Zašto su nodesi lokalni (kamera ne dela u dockeru) active nodes intruder alerts queue contents => poć na /docs od fastapija i vidit sve rute

3 6. Korisničke upute *(max. 4–6 stranica)*

3.1 Instalacija / Pokretanje

3.2 Korištenje - korak po korak

Figure 1: Screenshot - primjer korištenja.

Stavit slike setupa (hardverskega)

Reference

References

[1] Primjer reference.

A Dodatak A: Tehničke specifikacije

B Dodatak B: Logovi i testovi