FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

**Projekt iz predmeta Raspoznavanje uzoraka**

**Ak. god. 2020/21**

**Detekcija i lokalizacija nenormalnih događaja uporabom metode dubokog učenja**

Autori:

Marin Rađa

Ivona Farkaš

Fran Pugelnik

Branimir Ričko

Antonio Kuzminski

**siječanj, 2021.**

Sadržaj

[1. Projektni zadatak 3](#_Toc60936158)

[1.1 Opis projektnog zadatka 3](#_Toc60936159)

[1.2 Pregled i opis srodnih rješenja 3](#_Toc60936160)

[1.3 Konceptualno rješenje zadatka 4](#_Toc60936161)

[1.3.1 Arhitektura prostorno vremenske konvolucijske neuronske mreže 6](#_Toc60936162)

[2. Postupak rješavanja zadatka 7](#_Toc60936163)

[2.1 Priprema skupova podataka 7](#_Toc60936164)

[2.2 Izvlačenje SVOI-ja iz skupa podataka 7](#_Toc60936165)

[2.3 Učenje prostorne vremenske konvolucijske neuronske mreže 9](#_Toc60936166)

[3. Ispitivanje rješenja 10](#_Toc60936167)

[3.1 Ispitna baze 10](#_Toc60936168)

[3.1.1 UCSD skup podataka 10](#_Toc60936169)

[3.1.2 UMN skup podataka 11](#_Toc60936170)

[4. Opis programske implementacije rješenja 12](#_Toc60936171)

[5. Zaključak 13](#_Toc60936172)

[6. Literatura 14](#_Toc60936173)

# Projektni zadatak

## Opis projektnog zadatka

Otkrivanje nenormalnih događaja u scenama gdje postoje gužve, izuzetno je izazovno područje računalnog vida zbog ozbiljnih zakrivanja objekata scene, različitih intenziteta gustoće gužve i kompleksnih mehanika ljudskih gužvi, odnosno nepravilnim kretanjima ili aktivnostima gužve u okolini.

Široko korištenje nadzornih sustava u kolodvorima, zračnim lukama, na cestama ili u kupovnim centrima rezultira sve većoj količini video zapisa. Osim što postoji velika potreba za prepoznavanjem objekata i njihovog ponašanja isto tako potreba za detekcijom nenormalnih događaja postaje sve veća. Automatska detekcija nenormalnih događaja ili aktivnosti tijekom dugih tokova video zapisa je jedna od najvažnijih komponenata inteligentnog video nadzora.

Dodatno, postoje potencijalno opasne aktivnosti u okolini gdje postoji mnoštvo. Prilikom kakvih društvenih događaja koji se odvijaju na javim mjestima poput stadiona, trgova i sl., postoji mogućnost izbijanja stampeda ili bilo kakvog iznenadnog događaja kojim bi se potencijalno narušila sigurnost mnoštva. Za izbjegavanje takvih situacija potrebno ih je detektirati na vrijeme i reagirati adekvatno stoga ima smisla uložiti napor u razvitak sustava za detekciju nenormalnih događaja.

## Pregled i opis srodnih rješenja

Za detekciju nenormalnih događaja iz perspektive statične kamere postoji dosta literature. Primjerice u [2] je dan opsežan pregled postojećih metoda poput histograma, Bayesovih mreža, neuronskih mreža, stroja potpornih vektora, metoda grupiranja i dr., ali gotovo sve te metode su ograničene činjenicom da rade zadovoljavajuće raštrkanim scenama, odnosno scenama gdje su ljudi jasno vidljivi i među pojedinih ljudi nema preklapanja.

Predloženo je nekoliko modela inspiriranih fizikom gdje se primjerice u [3] za detekciju kretanja mnoštva koristi tehnika optičkog toka kako bi se lokaliziralo kretanje mnoštva uz model socijalne sile (engl. *social force model* - SFM) kako bi se pokušalo izdvojiti nenormalne događaje. [32] predlaže metodu korištenja optimizacije rojem čestica gdje se pokušava skrenuti čestice u regiju slike s najvećim kretanjem mnoštva.

Metode bazirane na kretanju obično izvlače informaciju poput optičkog toka i trajektorija kretanja iz prostorno vremenskog volumena (nekoliko sličica unatrag u vremenu) i tako modeliraju kretanje mnoštva.

Jedan od glavnih izazova kod detekcije nenormalnih događaja je detekcija u vremenskoj i prostornoj domeni [4]. To znači da je potrebno naći sličice u kojima se događaju anomalije i lokalizirati regije unutar sličice u kojima je nenormalan događaj.

Osim što je problem detektirati nenormalne događaje, drugi problem je što nema formalne definicije nenormalnog događaja. Ta definicije ovisi uvelike o problemu koji se rješava.

Postojeći pristupi detekciji nenormalnih događaja mogu se podijeliti u dvije kategorije: 1) metode gdje je objekt u centru razmatranja i 2) holističke metode gdje je u centru zbivanja pojedinac. U klasičnom modelu gdje je u centru zbivanja objekt, mnoštvo se tretira kao skup pojedinaca. Za razumijevanje zbivanja mnoštva, potrebno je segmentirati mnoštvo interesa u objekte. Metode gdje je objekt u centru razmatranja suočavaju se s problemima kod detekcije objekata, praćenja trajektorija i prepoznavanja ponašanja u krcatim scenama. Jednom kada kompleksnost scene naraste u smislu količine objekata, može se primijetiti značajna degradacija kvalitete detekcije objekata. Drugi vid modela, holistički model, cilj nije detekcija i praćenje svakog pojedinca, nego, mnoštvo se gleda kao cijeli entitet. Tipične značajke poput prostorno vremenskih gradijenata ili optičkog toka su korištene kod ove metode za lokalizaciju regija sa zanimljivim kretanjem.

Detekcija nenormalnih događaja može biti definirana kao problem binarne klasifikacije u razrede „normalan događaj“ i „nenormalan događaj“.

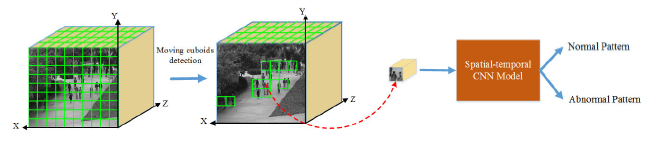
U skoro vrijeme puno radova demonstriralo je korisnost konvolucijskih neuronskih mreža (engl. CNN) [6] u različitim zadatcima poput klasifikacije i detekcije, prepoznavanje teksta, detekcije rubova, detekcija lica i dr. Korištenje konvolucijskih neuronskih mreža kod video zapisa pokazano je u [7] gdje se koristi 3D CNN za prepoznavanje ljudskih akcija gdje 3D označava niz slika u vremenu.

## Konceptualno rješenje zadatka

U ovom radu se fokusira na izazov detekcije nenormalnih događaja u prostornoj i vremenskoj domeni video zapisa (pojedinih sličica) gdje se nalazi mnoštvo promjenjive gustoće.

Za detekciju lokalnih anomalija koje se nalaze u lokalnim regijama tipična metoda bi bila da se po cijela slika podijeli u više manjih nepreklapajućih dijelova pa se onda svaki od tih manjih dijelova obrađuje bez obzira nosi li taj dio kakvu informaciju ili ne. Takva metoda je uvelike računalno zahtjevna i lako daje pogrešne detekcije anomalija. Bolja metoda koja bi smanjila računalnu složenost i poboljšali detekciju anomalija ima smisla obrađivati samo dijelove slike koji pružaju neku korisnu informaciju poput kretanja ljudi. S druge strane svi dijelovi slike koji ne pružaju nikakvu informaciju se ne obrađuju.

Postupak kojim se izvlače korisni dijelovi slike naziva se optički tok. Konkretnije, optički tok se izračunava Farnebackovom metodom [8] između pojedinih sličica video zapisa. Koliko sličica sudjeluje u izračunavanju optičkog toka može se odrediti proizvoljno, a ovdje se koristi 7 sličica u vremenskoj domeni. Jednom kada se izračuna optički tok, koriste se dijelovi slike na kojima je optički tok najizraženiji, a ta se izraženost može iskazati kao postotak piksela koji se kreću unutar pojedinih kvadrata (ovdje 65%). Ta područja su kvadrati dimenzija koje se također mogu odrediti proizvoljno, ovisno o skupu podataka koji se koristi. Ti kvadrati se protežu u vremenskoj domeni kroz broj sličica koji je određen prije i na taj način tvore volumen u prostorno vremenskoj domeni (engl. *spatial-temporal volume of interest* – SVOI). Svaki SVOI daje se na ulaz CNN mreže na analizu je li u tom SVOI-ju nenormalan događaj ili nije. Opisani postupak moguće je vidjeti na slici 1 koja je preuzeta iz [1].



Slika 6 Prikaz izvlačenja SVOI-ja iz niza sličica iz video zapisa

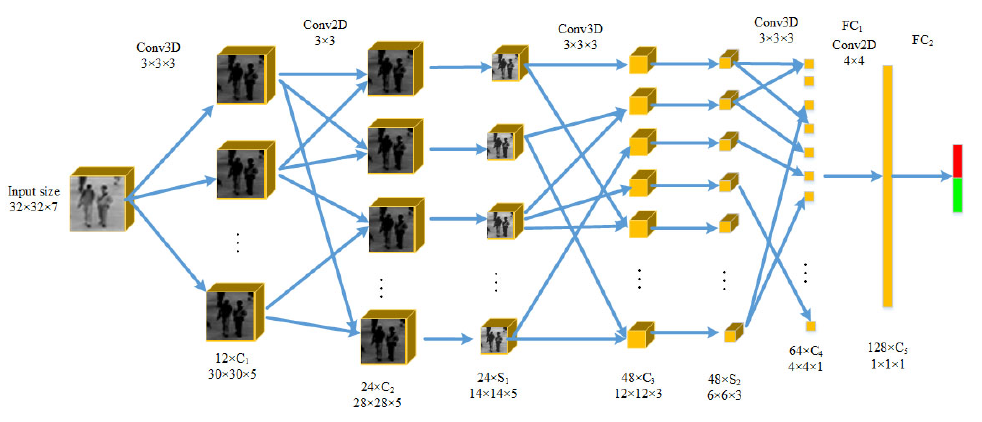
CNN je lokalno povezan model neuronske mreže koji izvlači lokalne značajke ograničavajući ulaze koje dobivaju skriveni slojevi. Tipična CNN se sastoji od konvolucijskih slojeva nakon kojih slijede potpuno povezani slojevi. Konvolucijski sloj koristi skup filtara kako bi obradio male lokalne regije ulazne slike. Korak poduzorkovanja (engl. *subsampling*) generira filter manje rezolucije na način da uzima najveću vrijednost unutar nekog definiranog prozora koji se kreće po ulaznom filteru.

U CNN-u konvolucija se računa u konvolucijskim slojevima. Formalno, *j-*ta aktivacijska mapa *aij* u *i­*-tom sloju je definirana kao gdje označava nelinearnu funkciju definiranu kao . predstavlja jezgru filtera, indeks skupa aktivacijskih mapa povezanih s trenutnom aktivacijskom mapom u -tom sloju, \* označava operaciju konvolucije, je dodatan skalar trenutnog sloja. Postupak učenja mijenja vrijednosti i za bolje izvlačenje značajki iz slike.

Za bolje izvlačenje informacija primjenjuju se prostorno vremenske konvolucije, a to se postiže korištenjem 3D jezgre koja zatim prolazi prostorno vremenskim volumenom, odnosno SVOI-jem. Takvim konvolucijama su aktivacijske mape povezane s više sličica u prethodnim slojevima čime se hvata koncept kretanja. Formalno, prostorno vremenska konvolucija između 3D jezgre i prostorno vremenskog volumena može se definirati kao:

Gdje predstavljaju visinu, širinu i vremensku domenu 3D jezgre, odgovara veličini prostorno vremenskog volumena .

### Arhitektura prostorno vremenske konvolucijske neuronske mreže



Slika 7 Arhitektura prostorno vremenske CNN

Arhitektura prostorno vremenske konvolucijske neuronske mreže prikazana je na slici 2 koja je preuzeta iz [1]. Mreža se sastoji od 8 slojeva, a na ulaz dolazi SVOI koji je dimenzija , gdje 32 označava veličinu kvadrata, a 7 vremensku dubinu SVOI-ja. Iako je veličina kvadrata SVOI-ja različite veličine za različite skupove podataka, svaka se veličina skalira na 32 kako je i prikazano na slici. Nakon ulaza u mrežu, primjenjuje se prva 3D konvolucija s veličinom jezgre pri čemu se koristi 12 filtera kojima se želi učiti različite značajke i zatim se generira 12 različitih aktivacijskih mapa kako je prikazano u sloju . Veličina svake mape je jer dolazi do gubitka rubne informacije sa svake strane volumena pošto nije korišteno nikakvo ispunjavanje slike kako bi se očuvala veličina nakon konvolucije. Nakon sloja primjenjuje se 2D konvolucija s veličinom jezgre koja prolazi svakim kvadratom SVOI-ja čime se dimenzija smanjuje na , ali ne prolazi vremenskom domenom SVOI-ja i time ta dimenzija ostaje očuvana. Nakon 2D konvolucije u sloju dolazi na red prvo poduzorkovanje veličine čime se želi dalje prenijeti važna informacija, ali u manjem formatu i nakon ove operacije svaka aktivacijska mapa je veličine (sloj ). Sljedeća je 3D konvolucija s jezgrom veličine čime se dimenzija svake mape smanjuje na . Nakon toga slijedi još jedno poduzorkovanje i 3D konvolucije gdje se u sloju dobivaju 64 aktivacijske mape veličine . Zadnja 3D konvolucija generira 128 aktivacijskih mapa koje su veličine u sloju . Svim konvolucijama i poduzorkovanjima se na kraju dobio jedan vektor koji ima 128 vrijednosti i koji se zatim linearnim slojem preslikava u dvije vrijednosti koje su na izlazu iz mreže. Dva broja su iz razloga jer želimo razlikovati normalne i abnormalne SVOI-je. Nakon dobivenih brojeva, primjenjuje se logistička regresija kako bi se normalizirale vjerojatnosti pojedinih razreda. Vjerojatnosti pojedinih razreda su dane s:

gdje označavaju prvi odnosno drugi broj koji su dobiveni na izlazu mreže.

# Postupak rješavanja zadatka

## Priprema skupova podataka

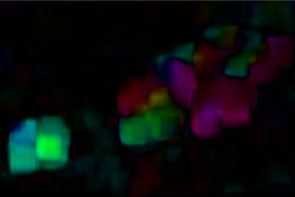
Skupovi podataka UCSD i UMN dolaze s označenim nenormalnim sličicama. UCSD dolazi s jednom matlab datotekom za svaki podskup (ped1 i ped2) gdje je zapisano koje su sličice abnormalni događaji. Radi pogodnosti se takav zapis prebacio u jednu tekstualnu datoteku kako bi se lakše učitao i koristio u programu. UMN skup podataka je jedan video zapis koji traje malo više od 4 minute gdje su pojedinih podskupovi spojeni zajedno i jedan počinje kada drugi završi. Svaki od tih podskupova u jednom trenutku počinje imate nenormalne događaje, a oni su označeni crvenim tekstom u gornjem lijevom uglu. Kako nije pogodno koristiti videozapis jer nema oznake po svakoj sličici, prvo je taj videozapis podijeljen u tri dijela. Svaki dio je jedan podskup (indoor, lawn i plaza).

Sljedeći korak je podijeliti svaki taj videozapis u pojedine sličice. Jednom kada je to napravljeno, u tekstualnu datoteku za svaki podskup je zapisano koje sve sličice sadrže abnormalne događaje.

Kod UCSD skupa podataka, sve su slike već u razinama sive boje dok je kod UMN skupa podataka, videozapis u boji. Prije konstrukcije pojedinih SVOI-ja, slike se prvo pretvore u slike sivih razina pa se tek onda nastavi postupak.

## Izvlačenje SVOI-ja iz skupa podataka

Kako bi dobili SVOI potrebno je izračunati optički tok Farnebackovom metodom na nekom (unaprijed određenom) broju sličica (). Za računanje optičkog toka između dvije slike koristi se implementacija Farnebackove metode iz knjižnice OpenCV [9]. Izlaz te metode je slika istih dimenzija kao i slike za koje se računa optički tok, a sadrži vrijednosti koje su izraženije na mjestima gdje je tok veći, odnosno veće su tamo gdje ima nekakvih promjena između dviju slika.



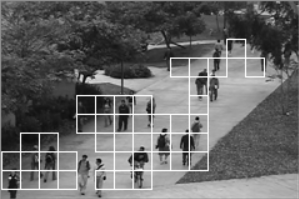
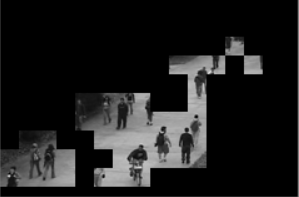
Slika 8 Prikaz dviju slika i optičkog toka među njima

Kada je izračunat tok (slika 8), filtriraju se vrijednosti ispod nekog praga za koje je smatrano da nisu dovoljno značajne (slika 9). Pozadina filtrirane slike je crna dok su pikseli koji zadovoljavaju uvjet, bijeli. Nakon izračuna optičkog toka nad slika dobiveno je novih slika koje imaju na određenim mjestima vrijednosti toka. Kako bi se dobili SVOI-ji od interesa prolazi se slikom slijeva na desno, odozgo prema dolje, gdje se trenutno gledaju vrijednosti u kvadratu dimenzija () preko svih slika. Ako u svakom tom kvadratu ima barem *p* posto vrijednosti koje nisu 0, onda to znači da je taj SVOI od interesa pošto se kroz svih slika događala nekakva promjena između slika na tom području slike.



Slika 9 Prikaz optičkog toka nakon filtriranja piksela koji nisu toliko važni





Slika 10 Prikaz korisnih SVOI-ja

Na slici 10 prikazano je kako izgleda kada se iz niza slika izvuku korisni SVOI-ji. Veličina SVOI-ja u vremenskoj domeni je 7 sličica. Lijeva strana pokazuje zadnju od 7 slika SVOI-ja i one kvadrate koji su dobiveni kao korisni. Desna slika pokazuje te iste kvadrate na crnoj pozadini kako bi se lakše vidjeli korisni kvadrati. Mijenjanjem različitih parametara poput veličine kvadrata, vremenske dubine, postotka piksela koji se moraju kretati i sl. može se dobiti slučaj da više kvadrata pokriva izvornu sliku ili slučaj da je manje kvadrata. Te je parametre potrebno mijenjati ovisno o skupu podataka jer ne postoji jedan skup parametara primjenjiv na svim skupovima podataka. SVOI se konstruira na slika kako je već rečeno i zatim se prelazi na novih slika koje služe za novu SVOI, nema nikakvog preklapanja među pojedinim slikama, a ako skup podataka nema broj slika koji je višekratnik broja , višak slika se ne koristi.

## Učenje prostorne vremenske konvolucijske neuronske mreže

Na slici 7 prikazana je arhitektura mreže. Mreža na ulaz dobiva SVOI dimenzija koji se dobiva na način opisan maloprije, a potrebno ga je skalirati pošto izvorni SVOI može biti različitih dimenzija. Dimenzije ulaznog SVOI-ja mijenjaju se kako je opisano u 1.3.1 i na kraju se dobiva vjerojatnost pripadnosti SVOI-ja razredu normalnih događaja ili nenormalnih.

Za učenje u neuronskoj mreži potrebno je moći računati grešku za svaki uzorak koji je predočen na ulazu i gradijente za propagaciju unatrag. Koristi se pogreška unakrsne entropije koja je definirana na sljedeći način (preuzeto iz [10]):

gdje predstavlja vektor dvaju brojeva koji se dobivaju na izlazu mreže, a predstavlja razred pripadnosti trenutnog SVOI-ja. Gubitak se računa za svaki SVOI zasebno jer količina SVOI-ja nije jednaka za svaki niz slika. Neke slike sadrže mnoštvo u većoj količini dok neke uopće ne sadrže mnoštvo stoga nema niti SVOI-ja koji su informativni.

Optimizacija parametara mreže radi se primjenom postupka gradijentnog spusta. Pokušava se pronaći minimum funkcije pogreške na način da se izračunavaju gradijenti funkcije pogreške po svakoj varijabli i onda se ažuriraju parametri koji se uče u smjeru negativnog gradijenta pošto se minimizira. Formula za ažuriranje težina gradijentnim spustom jer sljedeća:

gdje označava sve težine, odnosno parametre koji se uče, predstavlja stopu učenja (veličinu koraka prilikom svakog osvježavanja težina), označava funkciju pogreške. Formula prikazuje kako se težine za sljedeću iteraciju algoritma izračunavaju na temelju prošlih, a u prvoj iteraciji težine se nasumično inicijaliziraju, npr. u intervalu [0,1]. Potrebno je obratiti pažnju na stopu učenja i prilikom učenja isprobati različite vrijednosti jer ako je stopa učenja prevelika, algoritam može divergirati i suprotno, ako je stop premala, konvergencija algoritma optimizacije može trajati u beskonačnost. Stopu učenja je također moguće smanjivati kako iteracije algoritma napreduju čime se želi sve utjecati na težine jednom kada se algoritam približava optimumu.

Postupak učenja mreže terminira u slučaju da je dostignut maksimalan broj epoha ili je greška na skupu za učenje dovoljno mala. Odluku kreće li se u ponovno učenje s drugim parametrima donosi se nakon evaluacije modela mreže na testnom skupu podataka. Skupu koji mreža još nije vidjela čim se daje realnija procjena kvalitete modela.

# Ispitivanje rješenja

## Ispitna baze

Korišteni su skupovi podataka UCSD i UMN. UCSD dolazi s dva skupa podataka: „ped1“ i „ped2“ dok UMN dolazi s 3 podskupa: „indoor“, „lawn“ i „plaza“. Neki od primjera uzoraka slika iz svakog skupa podataka dani su na slikama koje slijede.

### UCSD skup podataka



Slika 1 Prikaz nekih slika iz ped1

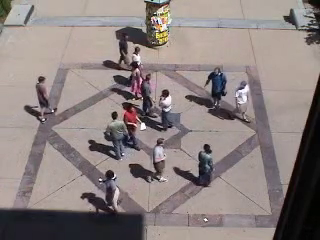
Slika 2 Prikaz nekih slika iz ped2

Moguće je vidjeti kako se na slikama iz USCD skupova nalazi mnoštvo promjenjive gustoće. Kamera se nalazi pod nekim kutom, a ujedno se događa puno prekrivanja pojedinih ljudi na slikama. Ovi skupovi dolaze s direktorijima „Train“ i „Test“, međutim direktorij „Train“ ne sadrži niti slike s nenormalnim događajima kako je i napisano od strane autora skupa podataka te se za učenje mreže koristi direktorij „Test“ koji se odgovarajuće podijeli na manje dijelove.

### UMN skup podataka



  
Slika 3 Prikaz nekih slika iz skupa "indoor"

  
Slika 4 Prikaz nekih slika iz skupa "lawn"

Slika 5 Prikaz nekih slika iz skupa "plaza"

Prikazom primjera iz skupa podataka UMN moguće također se vidi kako se pojavljuje mnoštvo različite gustoće, objekti se međusobno prekrivaju, a onda u jednom periodu nastupa nenormalan događaj koji je u ovom skupu označen u gornjem lijevom uglu crvenim slovima. Taj natpis ne utječe na kvalitetu klasifikacije mreže u smislu da bi ga se trebalo izrezati jer je natpis statičan, a statične stvari se ignoriraju prilikom izvlačenja SVOI-ja.

## Rezultati učenja i ispitivanja

## Analiza rezultata

# Opis programske implementacije rješenja

Programsko rješenje ostvareno je korištenjem Python 3.8 programskog jezika. Osim standardnih paketa bilo je potrebe za korištenjem funkcija iz OpenCV [9] i PyTorch [11] biblioteka. OpenCV je primarno korišten za manipulacije sa slikama poput učitavanje slike s diska računala, pretvorbu rgb slike u sliku sivih razina i skaliranje slike. PyTorch je biblioteka koja dolazi s gotovim implementacijama različitih algoritama iz strojnog učenja, a ovdje je korištena konvolucijska neuronska mreža s pripadnim algoritmom optimizacije gradijentnog spusta i funkcijom pogreške unakrsne entropije. Programska konstrukcija navedene mreže se vrlo lako ostvari jer je samo potrebno definirati sloj po sloj mreže. Osnovne stvari koje je potrebno definirati za konvolucijski sloj su: broj ulaznih i izlaznih aktivacijskim mapa i veličina jezgre. Sloj poduzorkovanja zahtijeva navođenje jezgre i veličine koraka. Potpuno povezani sloj zahtjeva broj ulaznih i izlaznih aktivacijskih mapa. Svaki sloj može imati neku nelinearnu aktivacijsku funkciju kroz koju prolaze rezultati operacija u samom sloju u sljedeći sloj koja je ovdje ReLu (engl. *rectified linear activation function*) čija je formula .

Prije bilo kakvog pokretanja programa jer potrebno instalirati potrebne biblioteke. Najlakši način, a da se ne dogode kolizije s drugim paketima je stvoriti novi python okoliš, aktivirati ga i onda npr. *pip* instaleru paketa predati datoteku koja se nalazi u direktoriju izvornog koda pod imenom „requirements.txt“. Ta datoteka sadrži definicije paketa potrebnih za ispravno funkcioniranje programa.

Jednom kada su paketi instalirani, program za treniranje je moguće pokrenuti. Taj program može primiti veći broj parametara od kojih nisu svi obavezni. Mogući parametri su:

|  |  |  |
| --- | --- | --- |
| Zastavica parametra | Zadana vrijednost parametra | Opis |
| -e | 20 | Broj epoha trajanja učenja algoritma. |
| -lr | 1 | Stopa učenja. |
| -tl | 7 | Vremenska dubina SVOI-ja. |
| -ss | 15 | Veličina kvadrata SVOI-ja. |
| -sigma | 10 | Prag iznad kojeg moraju biti razine slike nakon izračuna optičkog toka. |
| -ps | 0.5 | Postotak piksela koji se mora kretati unutar SVOI-ja kako bi se taj SVOI smatrao informativnim. |
| -resize\_images | False | Treba li skalirati slike tako da su dimenzije višekratnici broja stranice kvadrata SVOI-ja. |
| -ext | .tif | Ekstenzija slika koje se koriste za učenje. |
| -save\_model | False | Treba li spremiti model mreže nakon učenja. |
| -dataset | UCSD | Koji skup podataka se koristi. |
| -dataset\_name | PED1 | Koji skup podataka se koristi. |
| -h |  | Pomoć, prikaz mogućih parametara. |

Najosnovniji oblik naredbe za učenje mreže je sljedeći (pretpostavljeno je da je korisnik u direktoriju izvornog koda):

python train.py

Pokretanjem ove naredbe podrazumijeva se skup podataka UCSD ped1, 20 epoha, vremenska duljina SVOI-ja 7, veličina kvadrata SVOI-ja 15 uz još neke dodatne parametre. Korištenje naredbe za učenje u ovakvom formatu služi primarno za provjeru rade li svi dijelovi programa kako treba.

Za normalno učenje potrebno je predati neki skup parametara, primjerice sljedeće:

python train.py -e 100 ^  
 -lr 0.1 ^  
 -dataset UCSD ^  
 -dataset-name PED2 ^  
 -ext .tif

Naredba iznad je formatirana na više redaka radi preglednost i simboli „^ “ služe samo da se naredba napiše u više redaka na Windows-u. Programu za učenje mreže je predano u obliku parametara da učenje traje 100 epoha, stopa učenja je 0.1, koristi se skup podataka UCSD ped2 i slike u direktoriju imaju nastavak „.tif“. Ekstenzija je potrebna zbog toga jer se u direktorijima sa slikama mogu naći i druge datoteke.

Osim navedenih parametara u naredbi iznad postoje i drugi, a njih je moguće vidjeti ako se koristi sljedeća naredba:

python train.py -h

gdje se prikazuje skup svih naredba s odgovarajućim nazivima pojedinih parametara i početnim vrijednostima tih parametara ukoliko se ne preda nikakva vrijednost parametra.

Pokretanjem naredbe za učenje u terminalu se pokazuju dvije napredujuće trake. Jedna prikazuje napredak epoha dok druga prikazuje napredak po direktorijima skupa podataka. Jednom kada je naučen neki model, sprema se u direktorij „models“ i moguće je iskoristiti taj model kako bi se klasificirali neki novi, još dosad neviđeni, primjer. Također, moguće je učitati već modele koji su naučeni i prije na sljedeći način:

python classify.py -m models/cnn.pt

-f dana/UCSD/ped1/Test/Test001

-ext .tif

gdje „models/cnn.pt“ označava stazu do modela CNN koji se želi koristiti za klasifikaciju, sa zastavicom „-f“ daje se putanja do direktorija s novim slikama koje se žele klasificirati, a „-ext“ označava kao i prije nastavak slika koje se klasificiraju. Nakon pokretanja navedene naredbe prikazuje se prozor sa zadnjom slikom u SVOI-ju od *temporal\_length* slika koja služi kao slika na kojoj se prikazuje klasifikacija sadrži li skup od *temporal\_length* slika nenormalan događaj ili ne. Pritiskom bilo koje tipke na tipkovnici, prelazi se na sljedećih *temporal\_length* slika koje se klasificiraju i tako sve do kada postoji neklasificiranih slika. Naredbi iznad moguće je i predati zastavicu „-a“ kojom se slike same izmjenjuju sličeći više nadzornom video zapisu koji se događa uživo.

Ukoliko se želi vidjeti kako i koliko SVOI-ja se na nekom skupu podataka izvuče, to je moguće pokretanjem sljedeće naredbe:

python svoi\_demo.py

nakon čega se pojavljuje prozor na kojem je prikazana slika i na toj slici bijelim kvadratima dobiveni SVOI-ji pomoću optičkog toka. Pritiskom bilo koje tipke na tipkovnici prelazi se na sljedeći niz slika itd. Ukoliko se naredba pokrene kako je navedeno iznad, podrazumijevaju se neki početni parametri. Za promjenu parametara može se prikazati pomoć pomoću zastavice „h“. Jednom kada se prođe kroz sve slike direktorija, program automatski terminira.

# Zaključak

Problematika koja je obrađena u ovom radu, detekcija nenormalnih događaja, ima veliko područje primjene. Brže rastući Internet i kapaciteti pohrane, IP sustavi video nadzora postaju sve popularniji. Kako se poboljšala mrežna video tehnologija, trošak instalacije nadzornog sustava je pao znatno stoga se korištenje sigurnosnih kamera povećalo. Tipični nadzorni sustavi imaju više različitih kamera spojenih preko mreže ili lokalno. Ljudsko nadgledanje kamera može biti skupo i neefektivno jer kamere pružaju veliku količinu informacije te čak uvježbano osoblje može izgubiti koncentraciju i propustiti važne detalje nakon samo deset minuta.

Kako je već objašnjeno na početku, detekcija nenormalnih događaja u krcatim scenama je vrlo zahtjevan zadatak. Svojstva tih nenormalnih događaja mogu ovisiti o mnogo faktora okoliša u kojem se scena odvija poput osvjetljenja, prostora i vremena dana. Problem je sastavljen od više komponenata: procesiranja slike niske razine, analize optičkog toka i prepoznavanja koje se bazira na binarnoj klasifikaciji pojedinih SVOI-ja u razrede normalnih i nenormalnih događaja.

Razvijen je sustav prostorno vremenskog modela konvolucijske neuronske mreže za detekciju i lokalizaciju nenormalnih događaja u različitim scenama koje su snimljene statičnim kamerama. Vođeno se činjenicom da prostorno vremenski volumeni (SVOI-ji) sadrže korisnu informaciju o kretanju objekata u sceni te su oni sami služili kao ulaz u mrežu koja je zatim pokušavala naučiti koji bi SVOI-ji predstavljali nenormalne događaje, a samim time bi se za slike koje taj SVOI obuhvaća smatralo da sadrže nenormalan događaj.

…

.

.

.

.

Prostora za napredak ima puno. Primjerice učenje ovakve mreže izuzetno je vremenski i računalno zahtjevno. Zbog neposjedovanja grafičkih kartica nije bilo moguće paralelizirati učenje mreže i na taj način uvelike ubrzati proces eksperimentiranja s različitim parametrima. Korištenje GPU-a bi također omogućavalo duže učenje (više epoha) koje bi potencijalno moglo dati puno bolje modele stoga bi smislu poboljšanja cjelokupnog sustava u budućnosti imalo smisla uložiti vrijeme u paralelizaciju trenutnog koda. Drugi način kako se mogu koristiti pogodnosti višestrukih grafičkih kartica su različiti pružatelji usluga *cloud computinga*, primjerice Amazon AWS, Google Compute Engine i sl. koji se naravno moraju plaćati ovisno o vrsti uređaja koji se koristi za računanje.

Buduće ulaganje vremena u povećanje skupa podataka, pogotovo dijelova s nenormalnim događajima, sigurno bi pomoglo i na kraju davalo bolju klasifikaciju. Postoje različiti postupci koji su više i manje složeni, odnosno daju bolje i lošije rezultate, a normalizacija takvih odnosa gdje jednog razreda ima u puno većem broju naspram nekog drugog spada u jedan od osnovnih problema loših performansa bilo kakvih sustava učenja.

# Literatura

[1] Zhou S., Shen W., Zeng D., et al.: Spatial-temporal Convolutional Neural Networks for Anomaly Detection and Localization in Crowded Scenes. Signal Processing: Image Communication 47, pp. 358-368 (2016)

[2] Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM computing surveys (CSUR) 41(3), 15 (2009)

[3] Mehran, R., Oyama, A., Shah, M.: Abnormal crowd behavior detection using social force model. In: Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on, pp. 935–942. IEEE (2009)

[4] Nam, Y., Hong, S. Real-time abnormal situation detection based on particle advection in crowded scenes. J Real-Time Image Proc 10, 771–784 (2015)

[5] Cong, Y., Yuan, J., Liu, J.: Abnormal event detection in crowded scenes using sparse representation. Pattern Recognition 46(7), 1851–1864 (2013)

[6] LeCun, Y., Bottou, L., Bengio, Y., Haffner, P.: Gradient-based learning applied to document recognition. Proceedings of the IEEE 86(11), 2278–2324 (1998)

[7] Ji, S., Xu, W., Yang, M., Yu, K.: 3d convolutional neural networks for human action recognition. Pattern Analysis and Machine Intelligence, IEEE Transactions on 35(1), 221–231 (2013)

[8] OpenCV, Optical flow, <https://docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.html>, posjećeno 6.12.2020.

[9] OpenCv, <https://opencv.org/>, posjećeno 7.12.2020.

[10] PyTorch, pogreška unakrsne entropije, <https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html>, posjećeno 20.12.2020.

[11] PyTorch, <https://pytorch.org/>, posjećeno 22.12.2020.