Lokalizacija u konvolucijskim neuronskim mrežama

Pavičić, Josip

Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni preddiplomski studij matematike i računarstva

Josip Pavičić

Lokalizacija u konvolucijskim neuronskim mrežama

Završni rad

Osijek, 2021.

Sveučilište J.J. Strossmayera u Osijeku
Odjel za matematiku
Sveučilišni preddiplomski studij matematike i računarstva

Josip Pavičić

Lokalizacija u konvolucijskim neuronskim mrežama

Završni rad

Mentor: izv. prof. dr. sc. Domagoj Matijević

Osijek, 2021.

Sažetak

U svakom konvolucijskom sloju neuronske mreže očuvaju se spacijalne informacije objekata sa slike pomoću kojih se može napraviti lokalizacija. To ćemo pokazati pomoću CAM (en. class activation maps) i GAP (en. global average pooling). CAM za svaku klasu određuje diskriminativna područja na slici, na temelju koje konvolucijska neuronska mreža određuje kojoj klasi pripada slika. Drugim riječima, konvolucijska neuronska mreža označi dio slike na temelju kojeg je klasificira za unaprijed zadanu ili predviđenu klasu.

Konvolucijska neuronska mreža trenirana je na 16380 slika likova iz popularne serije Simpsoni, dimenzija 255x255 podijeljenih u 6 kategorija.

Rad ima pripadni praktični projekt u kojem je implementirana ideja rada. Napravljeni su eksperimenti koji su sastavni dio ovog rada.

Ključne riječi

CAM (en. class activation maps), GAP (en. global average pooling), konvolucijske neuronske mreže, granični okvir (en. bounding box)

Localization in convolutional neural networks

Summary

Spatial information about objects from the pictures is stored in every convolutional neural network layer which can aid with localization. This is presented using CAM (class activation maps) and GAP (global average pooling). CAM determines discriminatory areas for every class on the picture, based on which the convolutional neural network determines how the picture is classified. In other words, convolutional neural network designates a part of the picture based on which it then classifies the picture under a preset class.

Convolutional neural network training is based on 16380 pictures of characters from the critically acclaimed series "The Simpsons", measuring 255x255 split into 6 categories.

This bachelor's thesis contains the accompanying practical project where the thesis idea is carried out. The experiments form an integral part of the thesis.

Key words

CAM (class activation maps), GAP (global average pooling), convolutional neural networks, bounding box

Sadržaj

Uvod

Računalni vid (en. Computer vision) popularno je područje dubokog učenja. Temelji se na presjeku više znanosti, računalna znanost, matematika, inženjerstvo, fizika, biologija i psihologija. Računalni vid je konstrukcija eksplicitnih, smislenih opisa fizičkih objekata sa slike (vidi [1]).

Zamislimo da nam je dana neka slika mora. Naš zadatak je pronaći brod i prstom pokazati gdje se nalazi na slici, naravno, ako ga uopće ima na slici. Dakle, prvo pogledamo sliku i pogledamo sadrži li slika brod. Zatim lokaliziramo brod na slici, odnosno prstom ga pokažemo. Ovaj problem je poprilično lagan i intuitivan za nas ljude, ali za računala to je malo drugačija stvar.

Ovaj rad će obraditi probleme klasifikacije slike i lokalizacije objekta na istoj slici gdje će pratiti gore opisanu ideju. Klasifikacija slika je poddomena računalnog vida u kojoj algoritam danoj slici dodjeljuje klasu kojoj pripada. Klase su unaprijed definirane na kojima je algoritam treniran. Navedeni algoritam možemo podijeliti na nadzirano i nenadzirano učenje. U nadziranom učenju, klasifikacijski algoritam je treniran na slikama koje su već klasificirane, odnosno svaka slika ima klasu kojoj pripada. Time algoritam tijekom treniranja izvlači važne informacije za svaku klasu iz danih slika. Algoritam koristi te informacije kako bi klasificirao neviđene podatke, odnosno slike koje nisu unaprijed klasificirane. Za razliku od nadziranog učenja, u nenadzirano učenju algoritam treniramo na podacima koji nisu klasificirani te pokušavamo podijeliti podatke po nekakvoj sličnosti u klastere. U ovom radu imat ćemo pristup nadziranog učenja. Nakon klasifikacije slike, javlja se problem lokalizacije. On treba za klasificiranu sliku pronaći objekt na slici po kojoj je slika klasificirana, odnosno uokviriti taj dio slike pomoću graničnog okvira. Na primjer, u ovom radu za sliku koju klasificiramo kao Homer Simpson trebamo uokviriti lik Homera na slici. (slika 1)

Za potrebe ovog rada napravljen je projekt, koji pomoću konvolucijske neuronske mreže trenirane da klasificira 6 različitih likova iz popularne serije Simpsoni s točnošću od 94.67%, zatim ih lokalizira računajući CAM (en. class activation maps) za željenog lika, odnosno klasu te oko njega nacrta granični okvir s imenom lika.

U prvom poglavlju rada će se obraditi problem klasifikacije slika. Kako se slika klasificira i na koji način. Za klasifikaciju koristi konvolucijske neuronske mreže. U drugom poglavlju se definira problem lokalizacije i lokalizacija pomoću CAM-a.

Slika 1: Lokalizacija Homera Simpsona

1 Klasifikacija slika

Problem klasifikacije slika glasi: Za dani skup slika koje su označene sa svojom klasom kojoj pripadaju, trebamo predvidjeti te klase na nekom neviđenom, novom skupu slika i izmjeriti preciznost predikcija. Preciznost je broj u obliku postotka kojeg računamo:

$$preciznost = \frac{broj\ tocno\ odredenih\ predikcija}{ukupan\ broj\ predikcija} \cdot 100$$

Postavlja se pitanje kako napisati algoritam koji klasificira slike. Znanstvenici računalnog vida (en. computer vision) su osmislili podatkovni pristup problemu. Umjesto da za svaku klasu pišemo nepromjenjivi kod specijaliziran na njena jedinstvena obilježja, algoritmu damo puno slika za pojedinu klasu. Algoritam se tijekom treniranja postepeno popravlja (mijenja vrijednosti parametara unutar mreže) s krajnjim ciljem da ima što veću točnost . Za model koristimo konvolucijsku neuronsku mrežu, kojem ćemo pripremiti skup označenih slika na kojima će se model sam postepeno popravljati dok ne dobijemo zadovoljavajući model.

1.1 Konvolucijska neuronska mreža

Neuronske mreže izgrađene su od neurona organiziranih u slojeve. Svaki neuron u jednom sloju povezan je sa svim neuronima sljedećeg sloja. Matematički, neuron je nekakva funkcija $f(x)$ definirana kao kompozicija nekih drugih funkcija $g_i(x)$ koje su također kompozicija nekih drugih funkcija. Taj odnos neurona može se prikazati kao mreža sa strelicama koje pokazuju odnos između funkcija. Najčešća kompozicija neurona je $f(x) = K(\sum_i w_i g_i(x))$ gdje je K aktivacijska funkcija, a w_i težine. Graf takve mreže je usmjeren necikličan.

U treniranju neuronske mreže pronalazimo odgovarajuće težine neuronskih veza koje možemo naći zahvaljujući gradijentnoj povratnoj propagaciji (en. Gradient Backward propagation). U suštini, gradijentna povratna propagacija je algoritam koji se koristi za brzo

izračunavanje derivacija koje nam daju do znanja kako promijeniti težine modela za veću točnost.

Konvolucijske neuronske mreže neuronske su mreže s ograničenjima u arhitekturi koja smanjuju računalnu složenost i osiguravaju translacijsku invarijantnost (mreža tumači obilježja određene klase bez obzira na položaj u slici; Primjer: banana je banana bez obzira gdje se nalazi na slici). Konvolucijske neuronske mreže imaju tri važna obilježja:

- **Lokalna povezanost (en. local connectivity)**
  Neuroni u jednom sloju povezani su samo s neuronima u sljedećem sloju koji su im prostorno blizu. Ovaj dizajn prekida veliku većinu veza između uzastopnih slojeva, ali zadržava one koji nose najkorisnije informacije. Ovdje se pretpostavlja da ulazni podaci imaju prostorno značenje, odnosno da je odnos između dva udaljena piksela vjerojatno manje značajan od odnosa dva bliska piksela.

- **Zajedničke težine (en. shared weights)**
  To je koncept koji konvolucijske neuronske mreže čini "konvolucijskim". Prisiljavanjem neurona jednog sloja da dijele težine, prosljeđivanje prema naprijed (unos podataka kroz mrežu) postaje ekvivalent konvoluciji filtra preko slike za stvaranje nove slike, odnosno karte obilježja. Treniranje konvolucijskih neuronskih mreža tada postaje zadatak učenja filtera (odlučivanje koje značajke trebate tražiti u podacima). Rezultat zajedničkih težina je manji broj parametara mreže te, zato što imamo jednu matricu dijeljenih težina treniranih za izlučivanje nekog obilježja sa slike preko cijele slike, nije nam bitno gdje se to obilježje na slici nalazi.

- **Udruživanja (en. pooling) i aktivacijske funkcije(PReLU, ReLU, sigmoid, tanh)**
  Konvolucijske neuronske mreže imaju dvije nelinearnosti: udruživanja i aktivacijske funkcije. Udruživanja razmatraju blok ulaznih podataka i prosljeđuju jednu vrijednost koja ovisi o udruživanju. Na primjer, maksimum udruživanje vraća najveću vrijednost razmatranog bloka, dok prosječno udruživanje vraća aritmetičku sredinu razmatranog bloka podataka. Time se smanjuje veličina izlaza i smanjuje broj parametara za učenje, pa se slojevi udruživanja često koriste za regulaciju veličine mreže. Aktivacijska funkcija, na primjer ReLU, uzima jedan ulaz, x i vraća maksimum od $\{0, x\}$. Nelinearne aktivacijske funkcije uvode nelinearnost u model kako bi povećale mogućnost klasifikacije kompleksnih podataka.

1.1.1 Konvolucijski sloj

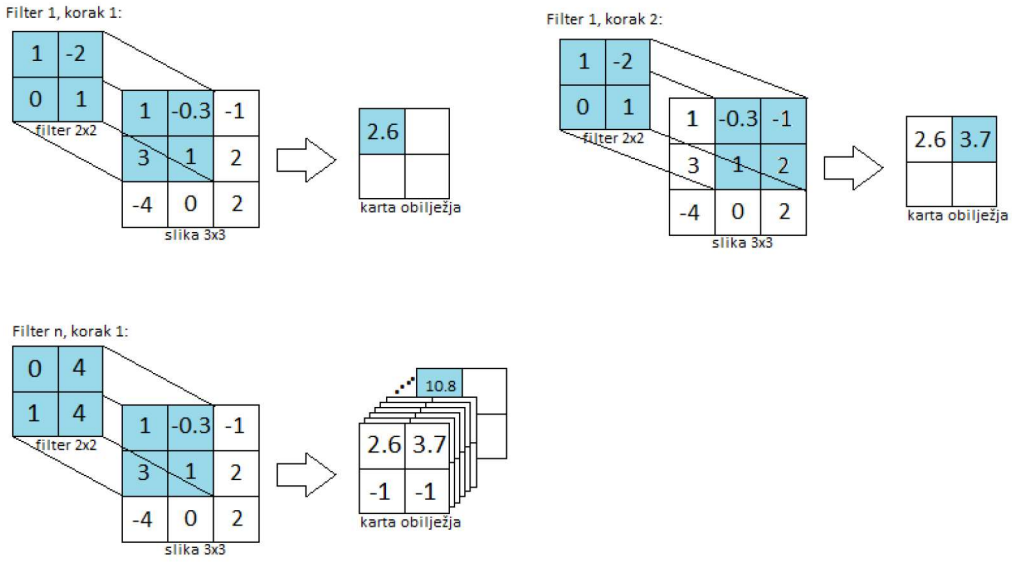
Najvažniji sloj u arhitekturi konvolucijske neuronske mreže. Sadrži skup konvolucijskih jezgri (en. kernel), također zvani filteri, koji se povezuju konvolucijama s ulaznom slikom za generiranje karte obilježja (en. feature map).

Filter je matrica koja sadrži diskretne vrijednosti, gdje svaku vrijednost nazivamo težinom jezgre. Na samom početku treniranja modela, sve težine su postavljenje na nasumično odabrane brojeve. Zatim treniranjem filter "nauči" izvući razna obilježja. Treba napomenuti da konvolucijska neuronska mreža koristi skup filtera u svakom konvolucijskom sloju tako da svaki filter izvlači drugačije obilježje slike. Što dublje ulazimo u kartu obilježja to su izvučena obilježja apstraktnija i ljudskom razumu ništa ne predstavljaju. Na slici 2 prikazan je filter dimenzija 2x2 i njegove težine.

1.3	0
1	-0.5

Slika 2: Filter dimenzija 2x2

Primjer 1. *Pokažimo kako to izgleda u praksi. Radi jednostavnosti računa uzet ćemo 3x3 crno-sivu sliku preko koje ćemo pomicati filter dimenzija 2x2 s korakom jedan.*



Slika 3: Računanje konvolucijskog sloja

U primjeru 1,odgovarajući kvadrat filtera konvoluiramo s odgovarajućim brojem na slici (Filter 1, korak 1: $1 \cdot 1 + (-2) \cdot (-0.3) + 0 \cdot 3 + 1 \cdot 1 = 2.6$). Koristimo konvolucijske operacije bez podebljanja rubova (en. padding). Podebljani rubovi su dodani pikseli na rubove slike. S njima bolje izvlačimo obilježja iz rubova te, ako odaberemo odgovarajuću debljinu podebljanja, dimenzije karte obilježja se ne moraju nužno smanjiti. Također treba napomenuti da korak za pomicanje(en. stride) u lijevo i dolje je 1.

Proizvoljno odabiremo dubinu karte obilježja, odnosno to će biti broj jednak broju filtera odabranih u konvoluciji. Visinu h' i širinu w' mape obilježja moramo izračunati. Trebaju nam veličine podebljanja rubova p, korak s, dimenzija filtera f te visina h i širina w prethodnog sloja. Analizom rada iz primjera 1 lagano dođemo do jednadžbi:

$$h' = \left\lfloor \frac{h - f + 2 \cdot p}{s} + 1 \right\rfloor \tag{1}$$

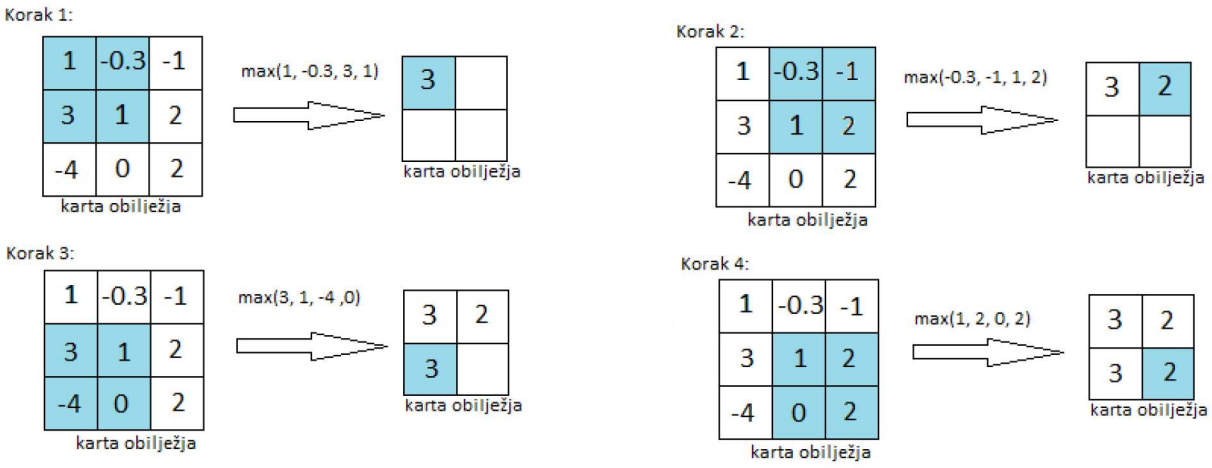
$$w' = \left\lfloor \frac{w - f + 2 \cdot p}{s} + 1 \right\rfloor \tag{2}$$

1.1.2 Udruživanja (en. pooling)

Slojeve udruživanja koristimo kako bi napravili poduzorke od karte obilježja. Uzimamo velike karte obilježja te joj smanjujemo visinu i širinu koristeći udruživanje. U sloju udruživanja nema težina, odnosno nema učenja. Za ovaj rad potrebna su nam dva udruživanja: globalno prosječno udruživanje (en. global average pooling) i maksimum udruživanje (en. max pooling).

Maksimum udruživanje ima visinu, širinu i korak. Pomičemo ga na isti način kao filter u konvolucijskom sloju, ali računanje je drugačije. Maksimum udruživanje u svakom koraku sačuva najdominantnije obilježje karte obilježja.

Primjer 2. *Pokažimo kako to izgleda u praksi. Radi jednostavnosti računa uzet ćemo 3x3 kartu obilježja s dubinom 1. Maksimum udruživanje će biti dimenzija 2x2 s korakom 1. U praksi ćemo imati kartu obilježja velikih dubina, za koju, na svakoj dubini ponavljamo prikazani postupak.*



Slika 4: Računanje maksimum udruživanja

Globalno prosječno udruživanje, za razliku od običnog koji je proizvoljne dimenzije, ima visinu i širinu karte obilježja. Dakle, za svaki sloj (dubinu n) karte obilježja dobit ćemo jedan broj. Iako se računa na drugačiji način, postupak je vrlo sličan maksimum udruženju. Prednost globalnog prosječnog udruživanja je ta što ima utjecaj regularizacije, odnosno pomaže pri sprječavanju pretreniranosti konvolucijske neuronske mreže.

Globalno prosječno udruživanje u popratnom projektu ovog rada implementirano je na kraju konvolucijskih slojeva, umjesto potpuno povezanog sloja.

Analizirajući postupak računanja iz primjera 2 možemo izvesti formule (3) i (4) za računanje visine *h'* i širine *w'* karte obilježja nakon primijenjenog udruživanja, uz oznake f za dimenziju udruživanja (kvadratna stoga imamo samo jednu vrijednost), s za korak, h za visinu i w širinu sloja na kojem računamo udruživanje .

$$h' = \left\lfloor \frac{h - f}{s} + 1 \right\rfloor \tag{3}$$

$$w' = \left\lfloor \frac{w - f}{s} + 1 \right\rfloor \tag{4}$$

Nedostatak udruživanja je što može smanjiti performansu konvolucijske neuronske mreže. Razlog tomu je zato što udruživanje pomaže mreži pronaći obilježja slike, ali ne obraća pažnju gdje se to obilježje nalazi.

1.1.3 Normalizacija

Normalizacija pomaže modelu da svaki njegov sloj bude nezavisniji od ostalih. Rezultate karte obilježja skalira u određene vrijednosti s manjom međusobnom razlikom. Normalizacija također ima ulogu regularizatora, odnosno sprječava pretreniranost modela. U popratnom projektu rada koristimo batch normalizaciju.

1.1.4 Aktivacijske funkcije

Aktivacijske funkcije nalazi se u svakom neuronu. Važna karakteristika aktivacijske funkcije je ta što omogućuje glatki prijelaz pri promjeni ulaznih vrijednosti, tj. mala promjena na ulazu proizvodi malu promjenu na izlazu. Nakon svakog konvolucijskog i potpuno povezanog sloja nalaze se nelinearni aktivacijski slojevi. Upravo radi te nelinearnosti naš model može "učiti". Da koristimo linearne aktivacijske funkcije, vrijedilo bi svojstvo:

$$A_1 * (A_2 * X) = (A_1 * A_2) * X = A * X$$

gdje su X podatak po kojem pomičemo filter, A_1, A_2 dva filtera takva da je A_2 sljedbenik od A_1 i A neki filter koji zadovoljava gornju jednakost.

Dakle, ako imamo linearne aktivacijske funkcije, filteri A_1 i A_2 su efektivni kao jedan filter A (također vrijedi za n uzastopnih filtera). Zato da bi svakim filterom izvukli različito obilježje, moramo imati nelinearne aktivacijske funkcije.

U popratnom radu kao najbolja od aktivacijskih funkcija (sigmoid, tanh, ReLU, Leaky ReLU, PReLU...) pokazala se najbolja PReLU.

Važna aktivacijska funkcija je softmax kojeg najčešće koristimo na izlazu neuronske mreže. Softmax je matematička funkcija koja pretvara vektor brojeva u vektor vjerojatnosti, gdje su vjerojatnosti svake vrijednosti proporcionalne veličini svake vrijednosti u vektoru. Svaka vrijednost u izlazu funkcije softmax tumači se kao vjerojatnost članstva za svaku klasu. Zbroj svih vjerojatnosti po klasama je 1. Softmax je dan izrazom:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{5}$$

gdje je σ softmax, $\vec{z}$ je vektor brojeva, K broj klasa u klasifikacijskom problemu.

1.1.5 Funkcije gubitka

Kako bismo odredili radi li algoritam dobar posao, moramo odrediti udaljenost između očekivanog izlaza i predikcije algoritma. Rezultat toga se koristi kao povratna informacija za prilagodbu rada algoritma, a to zovemo treniranje ili učenje algoritma.

Kako bismo odredili udaljenost koristimo funkcije gubitka. Najčešće, za klasifikacijski problem, koristi se funkcija Cross-entropy loss. Cross-entropy loss mjeri performanse klasifikacijskog modela čiji je izlaz vjerojatnost između 0 i 1. Cross-entropy loss raste kako se predviđena vjerojatnost razlikuje od stvarne oznake. Stoga bi predviđanje od 0.03 kad je stvarna oznaka promatranja 1 bilo krivo, što bi rezultiralo velikom gubitkom. Cross-entropy loss dan je formulom:

$$L = -\frac{1}{m} \sum_{i=1}^{m} y_i \cdot log(\hat{y}_i)$$

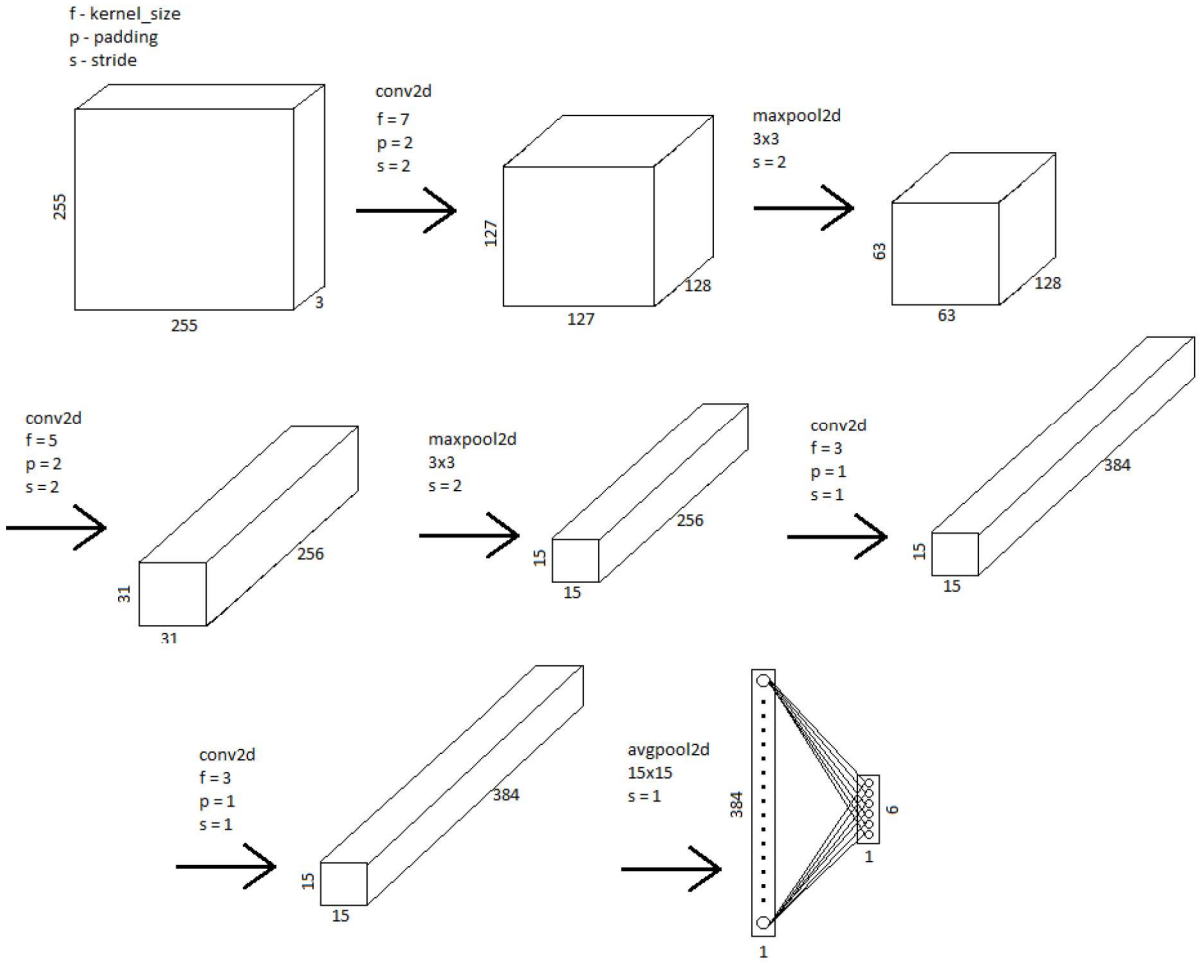
gdje je m broj podataka, y_i stvarne klase podataka, $\hat{y}_i$ predikcije modela podataka.

1.1.6 Arhitektura

Radi lokalizacije pomoću CAM-a koju želimo implementirati, potreban nam je posebna arhitektura konvolucijske neuronske mreže. Naime, nakon zadnjeg konvolucijskog sloja, slijedi točno jedan potpuno povezani sloj kojem je ulaz izlaz globalnog prosječnog udruživanja, a izlaz softmax veličine broja klasa klasifikacijskog problema. U prethodnim poglavljima objasnili smo kako se glavna obilježja slike očuvaju kroz slojeve. Ideja je posljednju kartu

obilježja direktno povezati s klasom za koju je ona značajna. Zbog tog razloga neuronska mreža mora na kraju konvolucijskog sloja imati globalno prosječno udruživanje čiji izlaz će softmax-om odmah povezati s klasom.

Prateći opis koji arhitektura konvolucijske neuronske mreže mora zadovoljavati kako bismo mogli izvući spacijalne informacije te napraviti lokalizaciju pomoću CAM-a, te računajući dimenzije aktivacija nakon svakog sloja pomoću jednadžbi (1), (2), (3) i (4) možemo skicirati arhitekturu mreže (slika 5).



Slika 5: Arhitektura konvolucijske neuronske mreže popratnog projekta

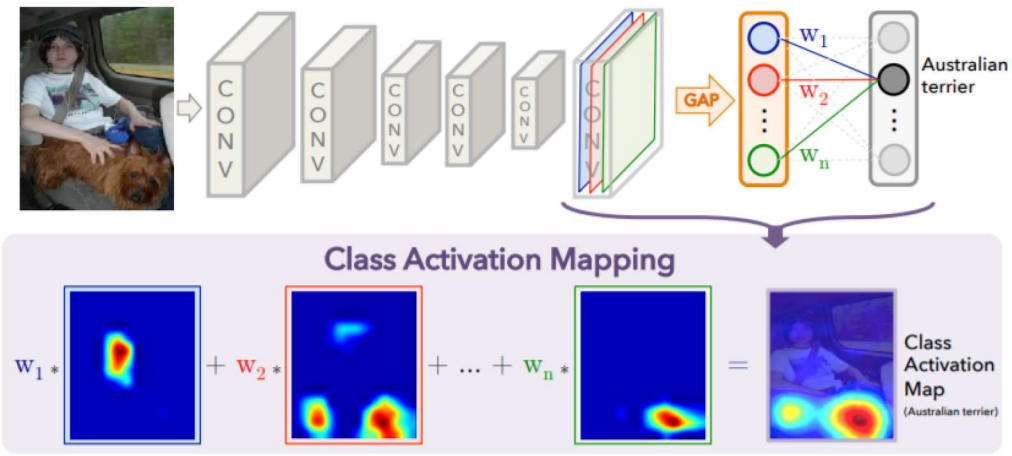
2 Lokalizacija

Prateći rad [2] pokušali smo implementirati lokalizaciju pomoću konvolucijske neuronske mreže koja klasificira slike u odgovarajuće klase. U radu se govori kako se konvolucijski jedinice u konvolucijskim neuronskim mrežama ponašaju kao detektori objekta iako nema nikakav nadzor učenja toga detektora. Iako konvolucijski slojevi imaju navedenu mogućnost pronalaženja objekta na slici, informacije o mjestu objekta se gubi potpuno povezanim slojevima.

Kako bismo dobili informaciju o lokaciji objekta na slici koristimo globalno prosječno udruživanje na posljednjoj karti obilježja iz konvolucijskog sloja. Zatim izlaz globalnog prosječnog udruživanja je ujedno i ulaz za potpuno povezani sloj u kojem djelujemo softmax funkcijom koja nam određuje klasu slike. Sad možemo pomoću trenirane težine povezati klasu s kartom obilježja.

2.1 CAM (en. class activation maps)

Kako model koji opisujemo kroz ovaj rad očuva i povezuje obilježja naučena u konvolucijskom sloju direktno s klasama te slike, možemo izračunati CAM pomoću kojeg ćemo lokalizirati objekt na slici.



Slika 6: Generiranje CAM-a

Izvor: [2], str. 3

Kako je prikazano na slici 6, globalno prosječno udruživanje (strelica GAP) računa prosječnu spacijalnu informaciju svakog zasebnog sloja iz posljednje karte obilježja. Zbroj svih umnožaka izlaza s odgovarajućom težinom nam generira završni sloj konvolucijske neuronske mreže. Kako imamo treniranu težinu za svaki sloj u posljednjoj karti obilježja, možemo za svaku vrijednost u karti obilježja koja se nalazi na koordinati (visina, širina, dubina) pomnožiti s odgovarajućom težinom željene klase. To nam daje, kad skaliramo dimenzije (visina, širina, dubina) na dimenzije ulaza, koliko je taj dio utjecao na klasifikacijsku odluku,

odnosno da se po tom dijelu slike klasificirala slika.

Formalnije rečeno, za danu sliku neka $f_k(x, y)$ predstavlja vrijednost aktivacije na dubini k i položaju (x,y) u karti obilježja. Možemo zapisati globalno prosječno udruživanje izrazom:

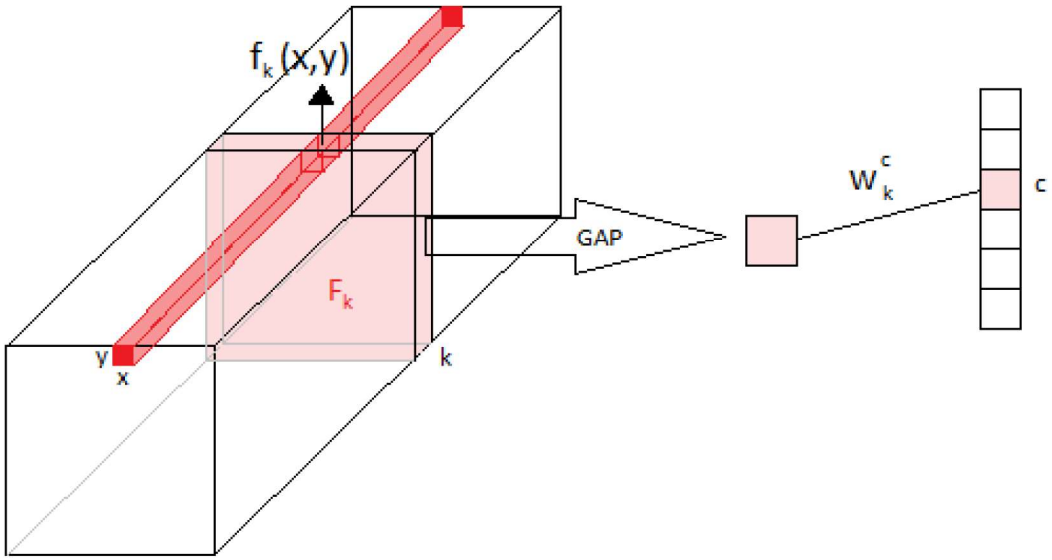
$$F^k = \sum_{x,y} f_k(x, y) \tag{6}$$

Ako imamo zadanu klasu, označimo ju s c, ulaz za softmax (S_c) je dan izrazom:

$$S_c = \sum_k w_k^c F_k \tag{7}$$

gdje je w_k^c je težina za dubinu k klase c.

Dakle težina w_k^c nam govori koliko je važan sloj F_k u karti obilježja za klasu c



Slika 8: Predočenje oznaka iz jednadžbi (6) i (7)

Uvrštavanjem (6) u (7) dobijemo:

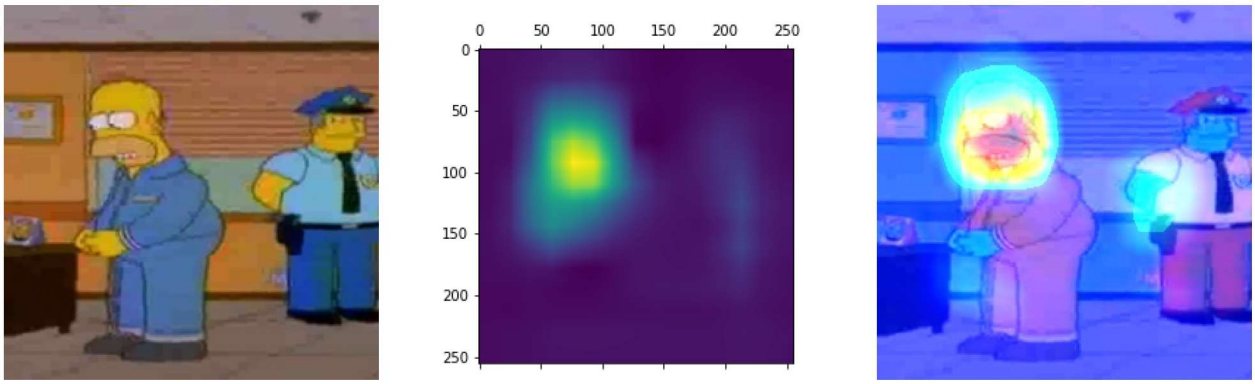
$$S_c = \sum_{x,y} \sum_k w_k^c f_k(x, y) \tag{8}$$

Definiramo M_c kao CAM za klasu c, gdje je svaki spacijalni element dan izrazom:

$$M_c(x, y) = \sum_k w_k^c f_k(x, y) \tag{9}$$

CAM je zapravo, za svaki (x, y), suma po dubini k spacijalnih informacija na (x, y) položaju pomnoženih s odgovarajućim težinama w_k^c. Time dobijemo CAM dimenzija manjih od ulaza, radi toga moramo promijeniti dimenzije CAM-a te ga možemo primijeniti na sliku.

Na slici 9, lijevu sliku smo klasificirali modelom implementiranim u popratnom projektu te izračunali njezin CAM (srednja slika) te smo primijenili CAM preko originalne slike (desna slika). Što je vrijednost CAM-a veća za piksel slike (x, y) to je piksel toplije boje. Sad možemo implementirati granični okvir.



Slika 9: CAM

2.2 Granični okvir (en. bounding box)

Granični okvir je, kao što samo ime kaže okvir koji će sadržavati toplije dijelove CAM-a. U popratnom projektu implementiran je tako što, nakon klasifikacije i izračuna CAM-a, traži najdesniji, najljeviji, najdonji i najgornji piksel koji na tom položaju u CAM-u ima vrijednost veću od 50% maksimalne vrijednost. Znajući te četiri vrijednosti s lakoćom napravimo okvir. Dodatno je još implementirano da napiše predikciju dane slike.

Slika 10: Granični okvir

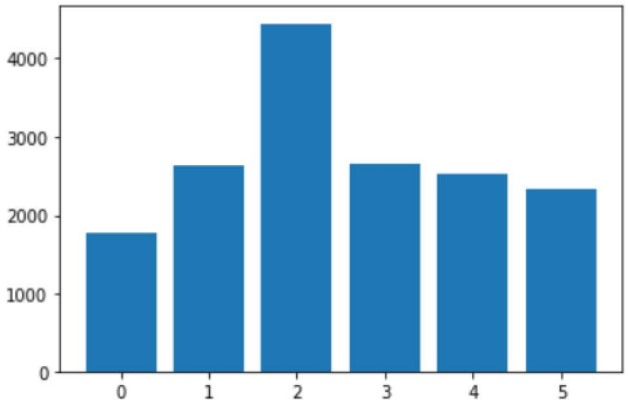
3 Evaluacija i rezultati

3.1 Podaci

Trenutno imamo neuronsku mrežu koju trebamo trenirati. Za to nam je potreban veliki skup podataka slika. Trening podaci bi trebali biti brojčano ravnomjerno zastupljeni za svaku kategoriju kako bismo dobili model koji nije treniran više za jednu kategoriju od

ostalih. Podaci, u našem slučaju slike u boji, trebamo promijeniti sve na istu veličinu, jer je ulaz konvolucijske neuronske mreže strogo definiran. Sa slike 7 vidimo da je potreban skup podataka dimenzija 255x255 u boji. Nadalje, poželjnije je imati što raznovrsnije slike. Kad učitavamo slike u konvolucijsku neuronsku mrežu, poželjno ih je promiješati kako se model ne bi prvo naučio na jednu klasu, zatim drugu i td., time ćemo potencijalno smanjiti evaluacijske performanse modela.

U popratnom projektu koristimo skup trening podataka koji broji 16380 slika Homera, Abrahama, Lise, Barta, Marge i Skinnera iz popularne serije Simpsoni raspoređenih u 6 klasa. (pogledaj sliku 10)



Slika 11: Granični okvir

Trening i validacijski skup podataka imaju svaki 50 slika za svaku kategoriju. Pomoću trening podataka gledamo je li model pretreniran ili podtreniran. Naime kad je model pretreniran ima odlične performanse na trening podacima, a lošije na test (neviđenim) podacima. To je zato što model ima previše parametara za svoju svrhu te nauči i šum slike, odnosno obilježja koja nemaju utjecaj na klasifikaciju klase. U terminima funkcije gubitka, funkcija gubitka je puno manja na trening podacima nego na test podacima. Podtrenirani model je suprotnost pretreniranom modelu. Model ima premalo parametara te ne može izvući sva bitna obilježja odgovarajuće klase. U terminima funkcije gubitka, funkcija gubitka je puno veća na trening podacima nego na test podacima.

3.2 Metrike modela

Za metrike modela koristimo podatke iz validacijskog skupa. Metrike ćemo računati pomoću matrice zabune.To je usporedba između stvarne klase podatka i klase koju model predvidi za taj podatak. Dobivamo matricu zabune ispunjenu s točno pozitivnim(TP), točno negativnim (TN), netočno pozitivnim(FP) i netočno negativnim(FN) klasificiranim slikama za svaku klasu. Pomoću te matrice možemo računati:

- točnost (en. accuracy) (svi točni/svi)
  - $accuracy = \frac{TP+TN}{TP+TN+FP+FN}$
  - generalna točnost modela. Nije korisna kad broj podataka među klasama nije približno jednak.
- netočnost (en. misclassification) (svi netočni/svi)
  - $misclassification = \frac{FP+FN}{TP+TN+FP+FN}$
  - koliko podataka nismo točno klasificirali
- preciznost (en. precision) (točno pozitivni/predviđeni pozitivni)
  - $precision = \frac{TP}{TP+FP}$
  - od svih predikcija n-te klase, koliko je njih uistinu bilo te klase
- osjetljivost (en. sensitivity aka recall) (točno pozitivni/svi pozitivni)
  - $recall = \frac{TP}{TP+FN}$
  - od svih pravih podataka n-te klase, koliko je njih točno klasificirano
  - maksimiziranjem preciznosti možemo smanjiti osjetljivost modela. Vrijedi i obratno.

```
Class:  abraham_grampa_simpson
[tn, fp, fn, tp]:  [249, 1, 6, 44]
Accuracy:  97.66666666666667
Misclassification:  2.3333333333333335
Precision:  97.77777777777777
Recall:  88.0
*****************************************
Class:  bart_simpson
[tn, fp, fn, tp]:  [247, 3, 1, 49]
Accuracy:  98.66666666666667
Misclassification:  1.3333333333333335
Precision:  94.23076923076923
Recall:  98.0
*****************************************
Class:  homer_simpson
[tn, fp, fn, tp]:  [248, 2, 0, 50]
Accuracy:  99.33333333333333
Misclassification:  0.6666666666666667
Precision:  96.15384615384616
Recall:  100.0
*****************************************
Class:  lisa_simpson
[tn, fp, fn, tp]:  [247, 3, 5, 45]
Accuracy:  97.33333333333334
Misclassification:  2.666666666666667
Precision:  93.75
Recall:  90.0
*****************************************
Class:  marge_simpson
[tn, fp, fn, tp]:  [245, 5, 0, 50]
Accuracy:  98.33333333333333
Misclassification:  1.6666666666666667
Precision:  90.9090909090909
Recall:  100.0
*****************************************
Class:  principal_skinner
[tn, fp, fn, tp]:  [248, 2, 4, 46]
Accuracy:  98.0
Misclassification:  2.0
Precision:  95.83333333333334
Recall:  92.0
*****************************************
```

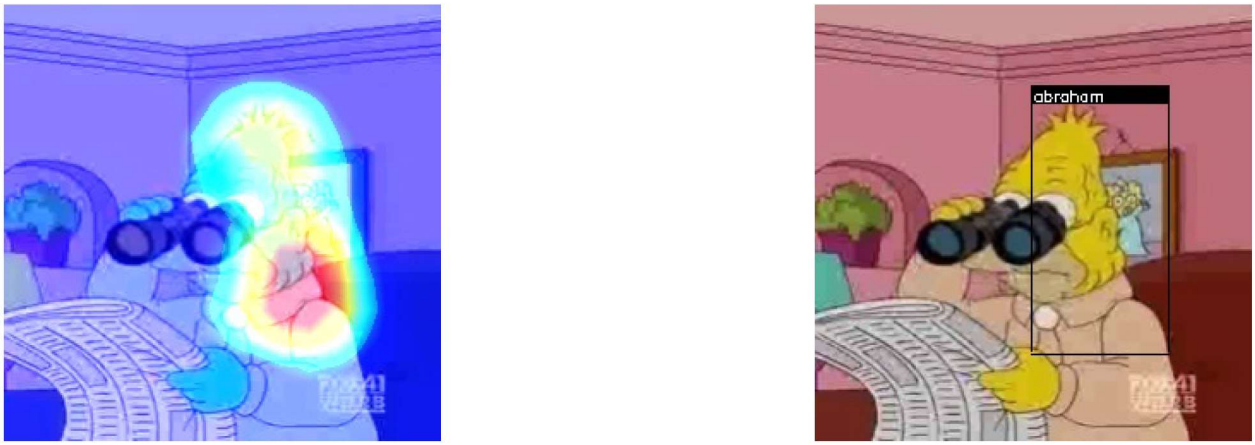
Slika 12: Metrike modela implementiranog prateći ovaj rad

Ako znamo da imamo 300 slika u validacijskom skupu te zbrojimo sve točno pozitivne za svaku klasu, možemo izračunati točnost modela koja iznosi $\frac{284}{300} = 0.9467$, odnosno 94.67%.

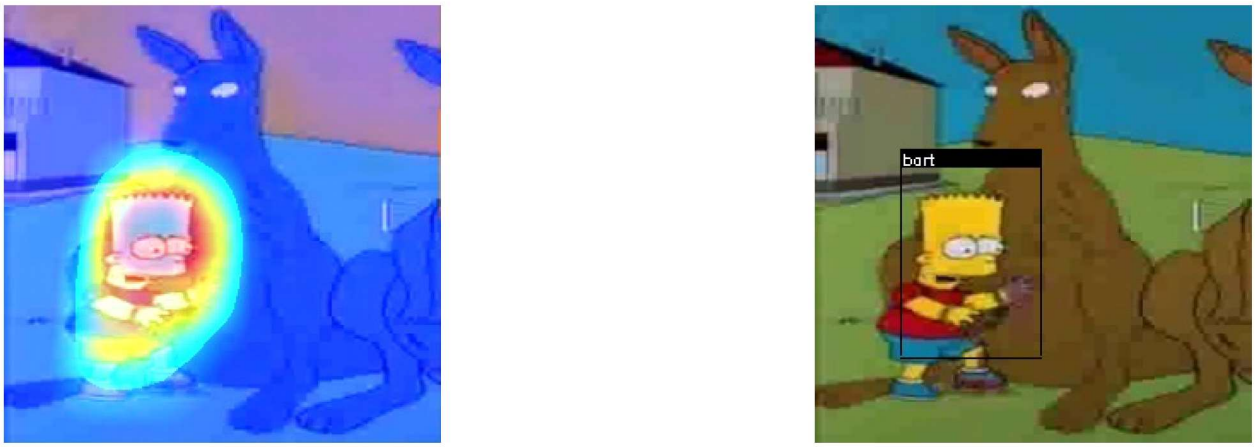
3.3 Rezultati lokalizacije

Rezultati lokalizacije također nam evaluiraju model na način što nam vizualno uokviri dio slike na temelju koje je klasificirana slika. Iz toga je vidljivo je li model naučio prepoznati prava obilježja slike ili je prepoznao na temelju nekakve nebitne pozadinske informacije. Na primjer imamo sliku auta na kojoj se vidi auto, ali i oblaci. Ako imamo trening podatke na kojima se mijenja položaj auta i sam auto, ali oblaci su konstantno na gornjem dijelu slike, moguće je da model naći da su oblaci auto te će u lokalizaciji uokviriti oblak, umjesto auta, što je krivo.

Računanjem lokalizacije kako je opisano u poglavlju dva uistinu vidimo da model, a i pristup lokalizaciji preko CAM-a daju zadovoljavajuće rezultate. Pogledajmo primjer lokalizacije za svaku klasu.

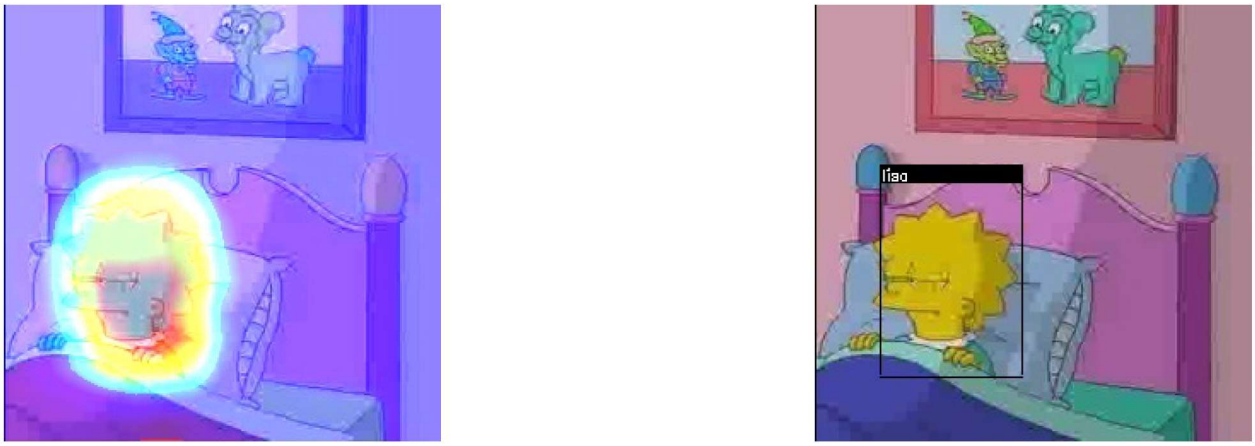


Slika 13: Abraham Simpson



Slika 14: Bart Simpson

Slika 15: Homer Simpson



Slika 16: Lisa Simpson

Slika 17: Marge Simpson

Slika 18: Principal Skinner

4 Zaključak

U ovom radu smo pokazali da koristeći odgovarajuću arhitekturu možemo model naučiti lokalizirati objekte iako ne nadziremo učenje istog. Takvu lokalizaciju nam omogućuje CAM. CAM nam vizualizira diskriminativne dijelove slike koje nam vizualno prikazuju na temelju kojeg djela slike i koliko je taj dio slike imao utjecaj pri klasifikaciji slike. Također ovim pristupom možemo direktno vidjeti koja karta obilježja prepoznaje koju klasu, odnosno koji filter izvlači obilježja za koju klasu i možemo vizualizirati koje je to obilježje. Dakle možemo filtrirati karte obilježja posljednjeg konvolucijskog sloja po klasama. Rezultat filtriranja je CAM kojim vidimo donosi li model dobre klasifikacijske odluke.

Literatura

[1] D.H. Ballard, C.M. Brown, *Computer vision, Prentice hall INC., New Jersey (1982), xiii*, dostupno na
https://homepages.inf.ed.ac.uk/rbf/BOOKS/BANDB/Ballard__D._and_Brown__C._M.__1982__Computer_Vision.pdf

[2] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, A. Torralba, *Learning Deep Features for Discriminative Localization (2015)*, dostupno na
https://arxiv.org/pdf/1512.04150.pdf