

SVEUČILIŠTE U ZAGREBU  
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

# **Ekstrakcija značajki slika u svrhu poboljšanja sustava preporuke**

Toni Vlaić, Viran Ribić

Zagreb, ožujak 2018.

*Ovaj rad izrađen je u Zavodu za elektroniku, mikroelektroniku, računalne i inteligentne sustave Fakulteta elektrotehnike i računarstva Sveučilišta u Zagrebu pod vodstvom doc. dr. sc. Marina Šilića i predan je na natječaj za dodjelu Rektorove nagrade u akademskoj godini 2017./2018.*

# SADRŽAJ

<b>1. Uvod</b>	<b>1</b>
1.1. Opći i specifični ciljevi rada . . . . .	1
1.2. Programska i hardverska podrška . . . . .	1
1.2.1. Programska podrška . . . . .	1
1.2.2. Hardverska podrška . . . . .	1
<b>2. Modeli za sažimanje</b>	<b>3</b>
2.1. Konvolucijske mreže . . . . .	3
2.1.1. Konvolucijski slojevi . . . . .	3
2.1.2. Slojevi sažimanja . . . . .	6
2.1.3. Potpuno povezani slojevi . . . . .	7
2.2. Analiza glavnih komponenti . . . . .	8
2.3. VGG . . . . .	10
2.4. Inception v4 . . . . .	10
<b>3. Rezultati i rasprava</b>	<b>13</b>
3.1. Metrike sličnosti . . . . .	13
3.2. Predtrenirana arhitektura VGG . . . . .	14
3.3. Predtrenirana arhitektura Inceptionv4 . . . . .	16
3.4. T-SNE prikaz smanjenog skupa podataka . . . . .	21
<b>4. Demo preporučitelj po sadržaju</b>	<b>22</b>
4.1. Neki tvoj section? . . . . .	22
<b>5. Zaključak</b>	<b>23</b>
<b>6. Sažetak</b>	<b>24</b>
<b>7. Summary</b>	<b>25</b>
<b>Literatura</b>	<b>26</b>

# POPIS TABLICA

3.1. Primjer uštede prostora . . . . .	20
--	----

# POPIS SLIKA

2.1.	Primjer ulazne slike 32x32x3 i rezultata izlaza nakon jednog konvolucijskog sloja dubine 5 [1] . . . . .	4
2.2.	Primjer računanja izlaza (Lijevo) iz ulaza koji sadrži jedan kanal (Desno) pomicanjem prozora po kanalu [2] . . . . .	5
2.3.	Utjecaj iznosa pomaka filtera na izlaze s prikazanim težinama na desnoj strani slike [1] . . . . .	5
2.4.	Povećanje receptivnog polja s dubinom mreže [3] . . . . .	6
2.5.	Sloj sažimanja s receptivnim poljem veličine 2x2 i pomakom 2 [1] . . . . .	7
2.6.	Primjer sažimanja maksimalnom vrijednošću filterom 2x2 i pomakom 2 [1]	7
2.7.	Potpuno povezana unaprijedna neuronska mreža . . . . .	8
2.8.	Određivanje glavnih komponenti podatkovnog skupa . . . . .	8
2.9.	Transformirani podatkovni skup . . . . .	9
2.10.	Podatkovni skup i crveno odabrane glavne komponente [4] . . . . .	9
2.11.	Konfiguracije VGG tima [5]. Parametri konvolucije su prikazani kao conv<dimenzije filtera>-<broj kanala> a nakon svake se nalazi ReLU aktivacija koja nije prikazana . . . . .	10
2.12.	Inception v4 [6] . . . . .	11
2.13.	Inception-A blok [6] . . . . .	12
2.14.	Reduction-A blok (za Inception v4 k=192, l=224, m=256, n=384) [6] . . .	12
3.1.	Primjer euklidske udaljenosti [7] . . . . .	13
3.2.	Primjer kosinusne udaljenosti [7] . . . . .	14
3.3.	Top 12 sličnih rezultata za sliku na vrhu . . . . .	15
3.4.	Primjer slike čije se slične traže (lijevo) i slika u top 5 koja nije ispravna (desno) . . . . .	16
3.5.	Top 12 sličnih rezultata za sliku na vrhu . . . . .	17
3.6.	Top 12 sličnih rezultata za sliku na vrhu . . . . .	18
3.7.	Top 12 sličnih rezultata za sliku na vrhu . . . . .	19
3.8.	Top 12 sličnih rezultata za sliku na vrhu . . . . .	20
3.9.	STL-10 dataset reprezentativni vektori prikazani u dvije dimenzije . . . . .	21

# **1. Uvod**

## **1.1. Opći i specifični ciljevi rada**

Hipoteza ovog istraživanja bila je da se korištenjem dubokog učenja može razviti sustav koji uspješno stvara nisko dimenzionalne reprezentacije slika proizvoljnih veličina kako bi se iste mogle efikasnije grupirati i pretraživati po njihovim semantičkim značajkama te iskoristiti u sustavima za preporučivanje temeljenim na sadržaju ili kao dodatak sustavima preporuke temeljenim na kolaborativnom filtriranju.

Cilj ovog istraživanja bila je ostvariti takav duboki model te ispitati različite načine ekstrakcije značajki iz dubokih modela te metrika kojima se ekstrahirane značajke mogu uspoređivati. Također za kraj potrebno je i razviti prototip jednostavnog sustava za preporučivanje temeljen na sadržaju koji koristi značajke ekstrahirane razvijenim modelom.

## **1.2. Programska i hardverska podrška**

### **1.2.1. Programska podrška**

Python

Tensorflow - u literaturi ti linkah rad pa mos nesto citirat cisto za reference dok ovo budes opisivao [8]

OpenCV - preprocessing i loadanje slikica (to mogu ja slozit) [9]

NMSLIB - brza pretraga skalabilnost etc (ono sta napisah u literaturi isto) [10]

TSNE - za vizualizaciju (to mogu ja opisat) [11]

Django?

### **1.2.2. Hardverska podrška**

Implementacija rješenja i ispitivanje napravljeno je na računalu s operativnim sustavom Ubuntu 16.04. Od važnijih komponenti koje utječu na brzinu i mogućnost reprodukcije

rezultata bitno je naglasiti da računalo ima grafičku karticu GTX Titan Pascal s 12GB memorije, intelov i7 procesor sa šest jezgri i 32GB radne memorije što je omogućilo obradu velikog podatkovnog skupa u razumnom vremenu

## 2. Modeli za sažimanje

### 2.1. Konvolucijske mreže

Konvolucijske neuronske mreže su postale popularne kao sredstvo za obradu slika nakon pobjede Alexa Krizhevsky-og na natjecanju 2012 ILSVRC (ImageNet Large-Scale Visual Recognition Challenge [12]) sa svojom konvolucijskom neuronskom mrežom danas poznatijom pod imenom AlexNet. Od tada je njihova popularnost kao sredstvo za obradu slika naglo porasla te su danas konvolucijske arhitekture najdominantnije na području obrade slika.

Računalo promatra sliku kao trodimenzionalnu matricu dimenzija  $X*Y*Z$  gdje se dimenzija  $Z$  može smatrati brojem kanala promatrane slike (uobičajeno crveni-zeleni-plavi kanal, poznatije kao RGB). Uzimajući u obzir da su ulazni podaci modela konvolucijske mreže uobičajeno slike, odnosno trodimenzionalne matrice, ona svoje neurone kojima obrađuje primljene podatke također uređuje u tri dimenzije širina, visina i dubina koja se može smatrati i brojem kanala kao u slikama, no bez limitacija na broj istih.

Jedna duboka konvolucijska mreža se uobičajeno gradi od konvolucijskih slojeva, slojeva sažimanja i potpuno povezanih slojeva unaprijedne neuronske mreže. Između svih slojeva u mreži nalazi se tzv. aktivacijska funkcija čiji je cilj uvesti nelinearnost u model.

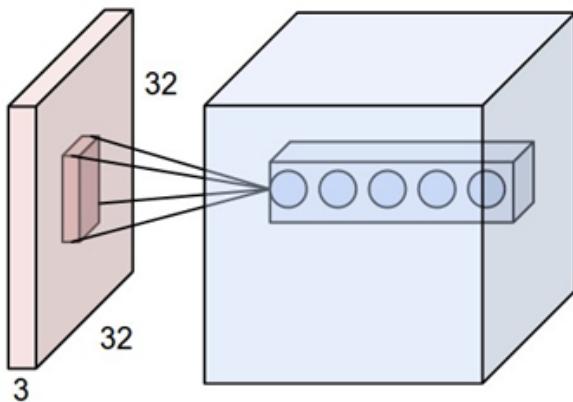
Aktivacijske funkcije su hiperparametri svake mreže i bitne su zbog unošenja nelinearnosti inače bi se mreža reducirala na afinu transformaciju koja preslikava ulazne vrijednosti u izlazne. Aktivacijska funkcija završnog sloja ovisi o tipu zadatka koji se mreža uči rješavati. Mreža trenirana za aproksimaciju funkcije neće raditi transformacije nad izlazom, već će aktivacijska funkcija izlaza biti funkcija identiteta. Drugi primjer bio bi zadatak klasifikacije za koji izlaz mora imati probabilističku interpretaciju, a ona se postiže sigmoidom u slučaju dvije klase ili softmax funkcijom za više klase.

#### 2.1.1. Konvolucijski slojevi

Konvolucijski sloj odradjuje većinu posla u konvolucijskoj mreži. Novost u konvolucijskoj mreži je povezivanje svakog neurona s malom regijom prethodnog sloja umjesto potpunog povezivanja svih neurona između slojeva kao u potpuno povezanoj unaprijednoj mreži. Svaki konvolucijski sloj ima tri dimenzije i neuroni svakog kanala dijele filtere s kojima

računaju izlaze iz prethodnog sloja.

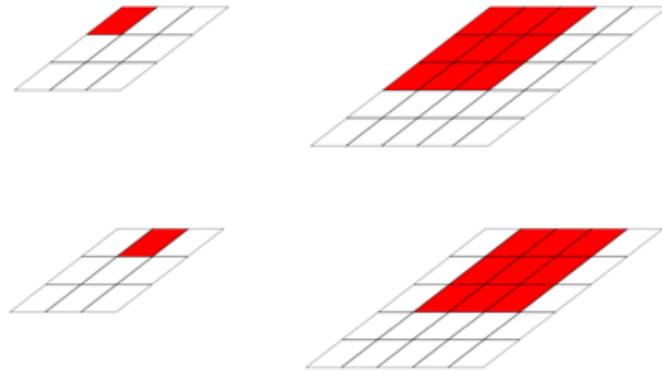
Uzmimo za primjer jedan konvolucijski sloj koji obrađuje CIFAR skup podataka s dimenzijama slike  $32 \times 32 \times 3$ . Ulaz takvog sloja su slike  $32 \times 32 \times 3$ , a filteri mogu biti zadani kao  $3 \times 3 \times 5$ . Dimenzije  $3 \times 3$  označava receptivno polje prozora koje pomiče po svakom kanalu ulaznog podatka, a broj pet označava novu dubinu nakon svih izračuna (Slika 2.1). Broj parametara takvog sloja iznosi broj izlaznih kanala ( $5 \times 3 \times 3 \times 3 = 135$ ) i dodatnih 5 vrijednosti za pristranost po svakom izlaznom kanalu.



**Slika 2.1:** Primjer ulazne slike  $32 \times 32 \times 3$  i rezultata izlaza nakon jednog konvolucijskog sloja dubine 5 [1]

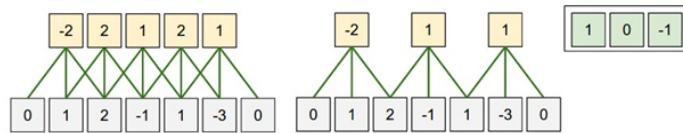
Intuitivno konvolucijski slojevi će naučiti filtere koji će reagirati na određeni tip vizualnih svojstava, pa ima smisla dijeliti parametre filtera unutar svakog kanala neovisno o trenutno promatranom području slike.

Svaki filter ima onoliko pomičnih prozora koliko prethodni sloj ima kanala, a svaki pomični prozor ima prethodno određenu površinu receptivnog polja, npr. 3 prozora površine  $3 \times 3$  kao u gornjem primjeru koji se pomiču po svom zaduženom kanalu. Moguće je svakom konvolucijskom filteru definirati korak (engl. stride) kojim se prozor pomiče po kanalima prethodnog sloja. Jedan položaj filtera daje jednu vrijednost u jednom izlaznom kanalu. Izlaz filtera je suma svih izlaza pojedinačnih prozora po kanalu (Slika 2.2). Svaki prozor sa slike množi određenom težinom element na odgovarajućem mjestu te sumira dobivene vrijednosti svih prozora dobivajući vrijednost krajnjeg izlaza. Nakon izračuna svih vrijednosti sloja one se šalju u aktivacijsku funkciju i koriste se kao ulaz u sljedeće slojeve.



**Slika 2.2:** Primjer računanja izlaza (Lijevo) iz ulaza koji sadrži jedan kanal (Desno) pomicanjem prozora po kanalu [2]

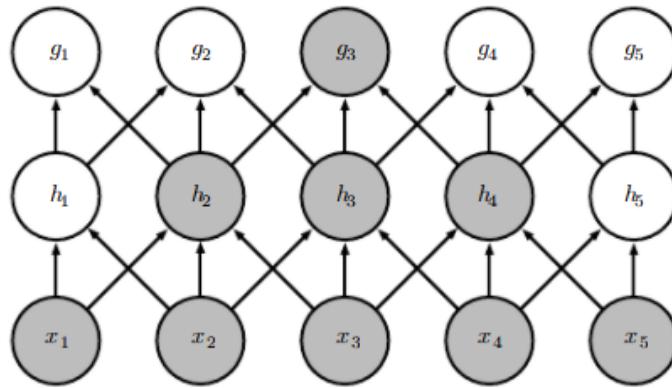
Korak pomaka (engl. stride) definira za koliko elemenata će se pomicati filter, odnosno svaki prozor pri računanju izlazne vrijednosti sljedećeg neurona. Veći korak utječe na dimenziju izlaznog sloja (Slika 2.3).



**Slika 2.3:** Utjecaj iznosa pomaka filtera na izlaze s prikazanim težinama na desnoj strani slike [1]

Osim koraka pomaka, na dimenziju izlaza utječe i površina receptivnog polja filtera. Veće receptivno polje uzrokovat će manji broj izlaza jer će se na ulaznu sliku moći posložiti manja količina prozora iz kojih se računaju izlazne vrijednosti. Uobičajena praksa je proširiti ulaznu sliku dodavanjem nula (engl. padding) na rubove kako bi se nakon konvolucije zadržala dimenzija izlaza jednaka ulazu. Količina nula koje je potrebno dodati ovisi o receptivnom polju i koraku konvolucije.

Dodavanjem slojeva receptivno polje novih neurona se povećava ovisno o receptivnom polju prošlih neurona. Primjerice nakon dvije konvolucije s receptivnim poljem  $3 \times 3$  i korakom jedan neuroni u drugom sloju vide veću površinu originalne slike od neurona u sloju prije njih (Slika 2.4). Zbog toga dublji slojevi konvolucijske mreže uče prepoznavati složenije vizualne značajke poput kotača, dok niži slojevi uče prepoznavati jednostavne značajke poput rubova objekata.

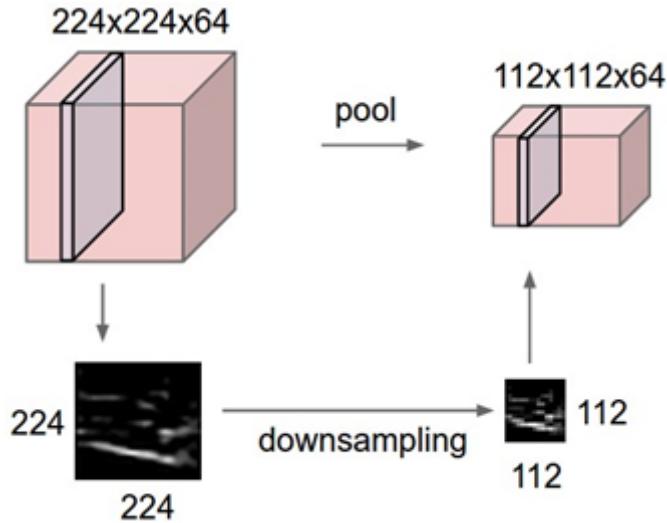


**Slika 2.4:** Povećanje receptivnog polja s dubinom mreže [3]

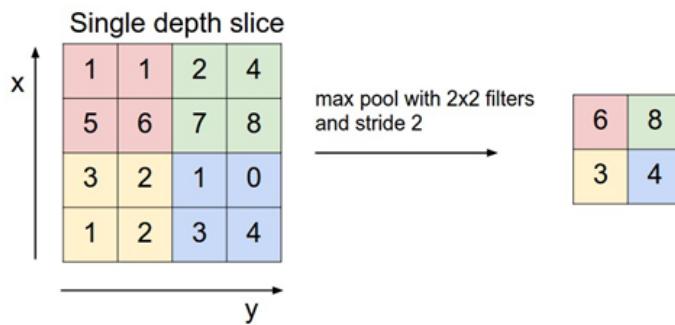
### 2.1.2. Slojevi sažimanja

Sloj sažimanja je vrlo jednostavan, ali vrlo bitan dio konvolucijske mreže. Sažimanje obično računa neku statističku vrijednost nad receptivnim poljem poput maksimalne vrijednosti ili srednje vrijednosti. Ideja sloja sažimanja je smanjiti dimenzionalnost (Slika 2.5) i broj potrebnih parametara mreže. Dodatni pozitivan efekt smanjenja broja parametara je i smanjenje vjerojatnosti da se takva mreža prenauči na ulazne podatke.

Osim reduciranja dimenzionalnosti slojevi sažimanja unose invarijantnost na pomake u slici. Do intuitivnog objašnjenja prethodne tvrdnje možemo doći ako pretpostavimo da filter reagira na neki oblik u prethodnom sloju što prepoznajemo po većoj pozitivnoj vrijednosti izlaza od ostalih susjednih neurona (referenca na drugu sliku ispod). Budući da za klasifikaciju nije bitno znati gdje se oblik nalazi na slici nego nalazi li se na slici može se iskoristiti sažimanje maksimalnom vrijednosti. Takvo sažimanje će u sljedeće slojeve propustiti samo maksimalni odziv u promatranom prozoru (Slika 2.6).



**Slika 2.5:** Sloj sažimanja s receptivnim poljem veličine  $2 \times 2$  i pomakom 2 [1]

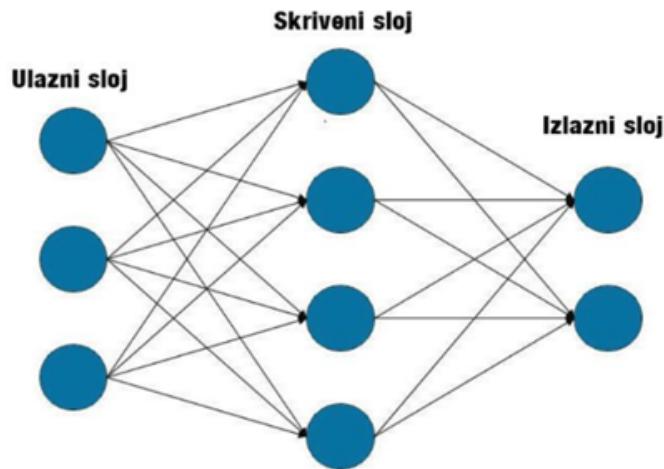


**Slika 2.6:** Primjer sažimanja maksimalnom vrijednošću filterom  $2 \times 2$  i pomakom 2 [1]

### 2.1.3. Potpuno povezani slojevi

Potpuno povezani slojevi dolaze iz najjednostavnijeg oblika neuronskih mreža, potpuno povezanih unaprijednih mreža, zbog čega će ovi slojevi biti objašnjeni direktno objašnjavanjem načina rada te vrste arhitekture mreža.

Potpuno povezana unaprijedna neuronska mreža je građena od slojeva koji se razlikuju jedino po aktivacijskoj funkciji na izlazu, ulazni sloj na koji stižu podaci čija je aktivacija funkcija identiteta, skrivenih slojeva na čijem se izlazu nalazi neka derivabilna nelinearna funkcija te izlaznog sloja s konačnim rezultatom izračuna s aktivacijom prikladnom za problem koji se nastoji riješiti (Slika 2.7). Svaki ulaz u neuron ima svoju težinu koja množi ulaz te se svi ulazi na kraju sumiraju i šalju u aktivacijsku funkciju. Rezultat aktivacijske funkcije zatim postaje ulaz u sljedeće slojeve ili predstavlja završni rezultat iz zadnjeg sloja.

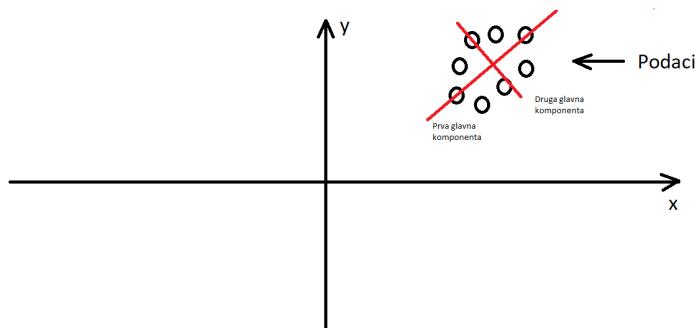


**Slika 2.7:** Potpuno povezana unaprijedna neuronska mreža

## 2.2. Analiza glavnih komponenti

Analiza glavnih komponenti (engl. Principal component analysis - PCA) standardni je alat u modernoj analizi podataka - u različitim područjima od neuroznanosti do računalne grafike zbog toga što je jednostavna, neparametarska metoda za ekstrahiranje relevantnih informacija iz podatkovnog skupa [4].

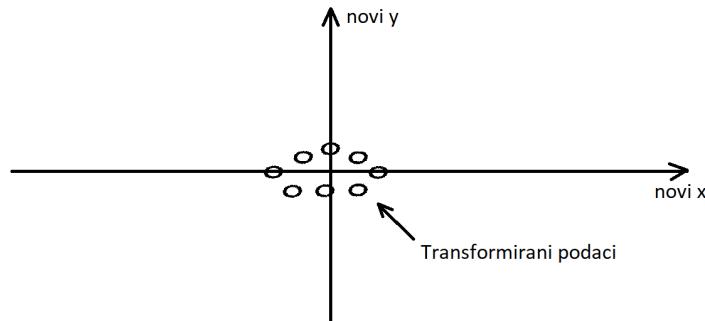
Bez uloženja u matematiku koja se može pronaći u radu [4] PCA se može objasniti laćim terminima kao tehnika koja pronalazi temeljne varijable koje najbolje diferenciraju podatkovni skup (poznatije kao glavne komponente - engl. principal components). Glavne komponente skupa su dimenzije po kojima je naš podatkovni skup najviše raspršen (Slika 2.8).



**Slika 2.8:** Određivanje glavnih komponenti podatkovnog skupa

Nakon što je određen traženi broj glavnih komponenti (u slučaju sa slike njih dva) podatkovni skup se transformira u novi koordinatni sustav (Slika 2.9).

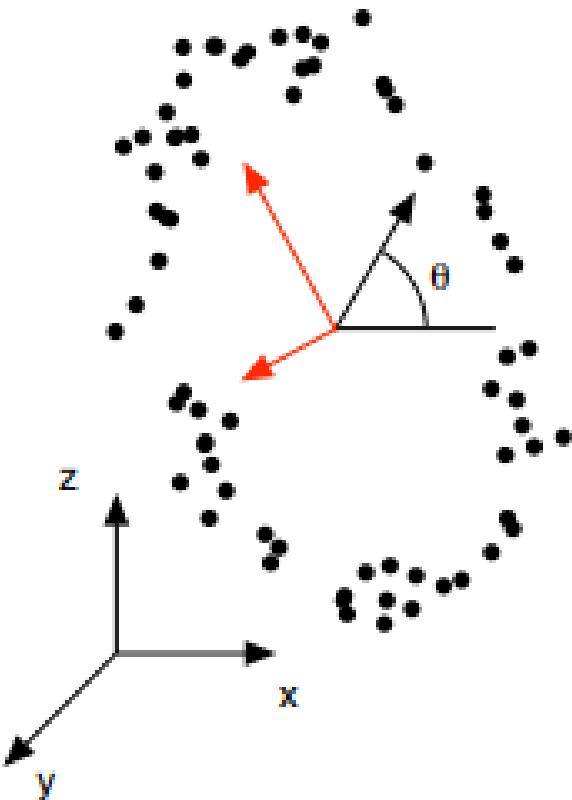
Iako se u gore navedenom primjeru dimenzionalnost podataka ne reducira (2D podaci ostaju u 2D prostoru) glavno područje primjene PCA tehnike je upravo redukcija dimen-



**Slika 2.9:** Transformirani podatkovni skup

zionalnosti koja funkcioniра na skoro identičan način. Razlikuje se po tome što se nakon određenih smjerova raspršenja oni poredaju silazno po svojstvenim vrijednostima te se odabere njih n (željena dimenzija na koju se reducira) nakon čega se podaci transformiraju u taj koordinatni prostor čime se gube ostale dimenzije s manjim svojstvenim vrijednostima.

Važno je naglasiti da je glavna motivacija iza PCA tehnike uklanjanje korelacije u podatkovnom skupu odnosno uklanjanje zavisnosti drugog reda između podataka [3]. Način na koji PCA nastoji ostvariti taj cilj je pretpostavkom su glavne komponente međusobno ortogonalne zbog čega neće biti primjerena za podatkovne skupove poput onog na slici 2.10.



**Slika 2.10:** Podatkovni skup i crveno odabrane glavne komponente [4]

## 2.3. VGG

Arhitektura prvog isprobog modela datira iz 2014., a dolazi sa sveučilišta Oxford čiji ju je tim koristio u ImageNet Large Scale Visual Recognition Challenge natjecanju. VGG tim, po kojem je mreža dobila i ime, ostvario je prvo mjesto iz zadatka lokalizacije objekata i drugo mjesto iz zadatka klasifikacije.

Njihov glavni doprinos je temeljita evaluacija arhitektura dubokih mreža koje koriste konvolucijske filtere malih dimenzija (3x3), čime su pokazali da se značajno poboljšanje u odnosu na tadašnje najbolje arhitekture može postići povećanjem dubine mreže na 16-19 slojeva dubine [5].

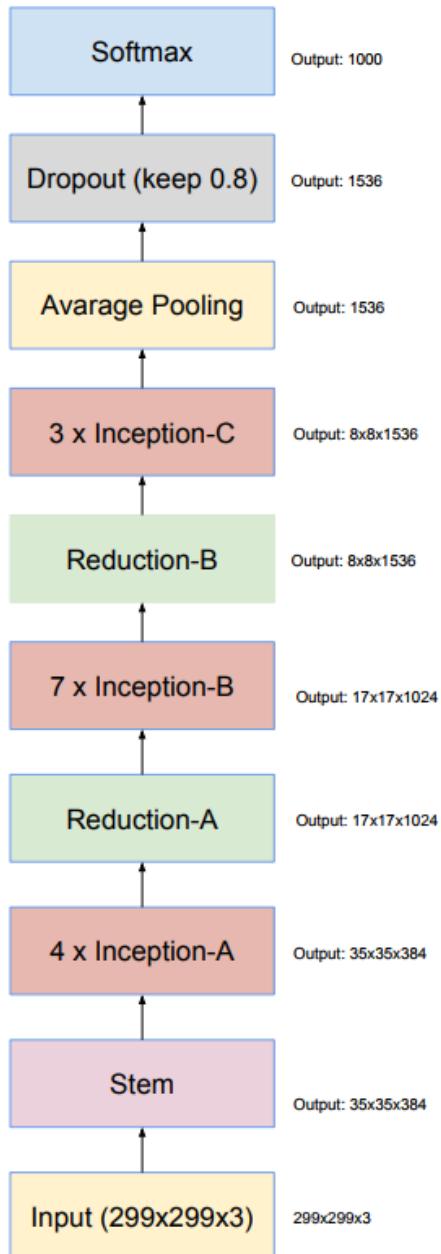
Nakon evaluiranja više konfiguracija mreža VGG tim je pokazao su da dublja konvolucijska mreža nadmašuje rezultate mreža s manje slojeva, zbog čega smo u našim ispitivanjima koristili njihovu najdublju mrežu u stupcu E sa slike 2.11.

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 LRN	conv3-64 <b>conv3-64</b>	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 <b>conv3-128</b>	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 <b>conv1-256</b>	conv3-256 conv3-256 <b>conv3-256</b>	conv3-256 conv3-256 conv3-256 <b>conv3-256</b>
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 <b>conv1-512</b>	conv3-512 conv3-512 <b>conv3-512</b>	conv3-512 conv3-512 conv3-512 <b>conv3-512</b>
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 <b>conv1-512</b>	conv3-512 conv3-512 <b>conv3-512</b>	conv3-512 conv3-512 conv3-512 <b>conv3-512</b>
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

**Slika 2.11:** Konfiguracije VGG tima [5]. Parametri konvolucije su prikazani kao conv<dimenzije filtera>-<broj kanala> a nakon svake se nalazi ReLU aktivacija koja nije prikazana

## 2.4. Inception v4

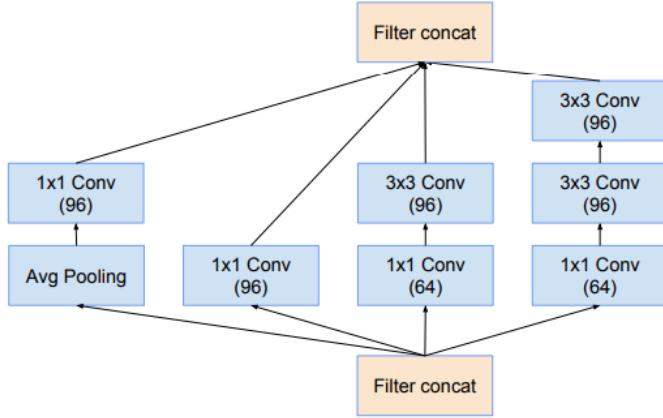
Inception v4 model je Googleova četvrta iteracija Inception arhitekture koja se od VGG arhitekture najviše razlikuje po svojoj “širini”. Arhitektura mreže u skraćenom prikazu Inception blokova se može vidjeti na slici (Slika 2.12).



**Slika 2.12:** Inception v4 [6]

Inception blokovi su postali popularni nakon što je Googleova tadašnja arhitektura GoogLeNet izgrađena od Inception blokova ostvarila dobre performanse na natjecanju ImageNet. Intuitivno njihova glavna prednost je rješavanje problem odabira dimenzija filtera pri izgradnji mreže jer se sastoje od više grana konvolucija koje ulazni podatak obrađuju s različitim dimenzijama filtera. Zbog stabilne razgranatosti Inception blok iz ulaza može prepoznati komplikiranije uzorce jer svaka grana ulazni podatak promatra s različitim receptivnim poljem (npr. Slika 2.13).

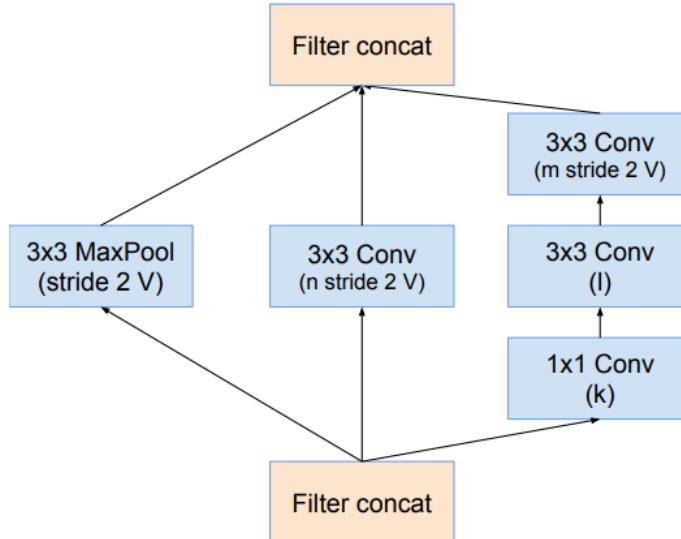
Inception moduli od kojih je građena mreža (A-C) obrađuju ulaze na sličan način. Za-



**Slika 2.13:** Inception-A blok [6]

jednička karakteristika im je da izlazne dimenzije podatka ostanu jednake (npr. Inception-A na ulazu prima  $35 \times 35 \times 384$  a na izlazu daje  $35 \times 35 \times 384$ ), a razlikuju se po načinu obrade ulaznog podatka, točnije broja konvolucijskih slojeva u granama i njihovih parametara.

Osim Inception modula mreža se sastoji i od Reduction blokova koji su, uz ekstrakciju značajki, zaslužni za smanjivanje dimenzionalnosti ulaznog podatka. Kao i Inception blokovi razlika između Reduction bloka A i Reduction bloka B je samo u parametrima grana i dizajnu grana, a oba bloka smanjuju širinu i visinu ulaznog podatka za faktor 2 kombinacijom globalnog sažimanja ili konvolucija s korakom 2 (Slika 2.14).



**Slika 2.14:** Reduction-A blok (za Inception v4  $k=192$ ,  $l=224$ ,  $m=256$ ,  $n=384$ ) [6]

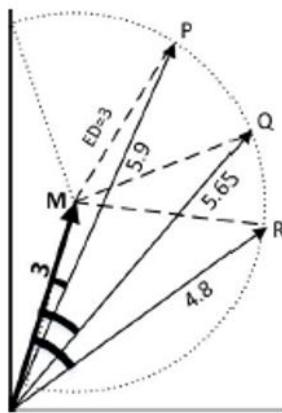
# 3. Rezultati i rasprava

## 3.1. Metrike sličnosti

Prilikom uspoređivanja sličnosti vektora isprobali smo dvije metrike euklidsku i kosinusnu udaljenost.

Euklidska udaljenost definirana je kao L2 udaljenost između dva vektora (3.1), a raspon vrijednosti iznosi od nula za točke koje se nalaze na istom mjestu u prostoru do beskonačnosti. Potencijalno negativna strana euklidske udaljenosti je zanemarivanje informacija o kutu između dvije točke. Primjer te situacije se može vidjeti na slici 3.1 gdje su točka P, Q i R jednako udaljene od točke M, no kada bi uzimali informaciju o kutu u obzir najsličnija bi nam bila točka P.

$$EuclideanDistance(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (3.1)$$

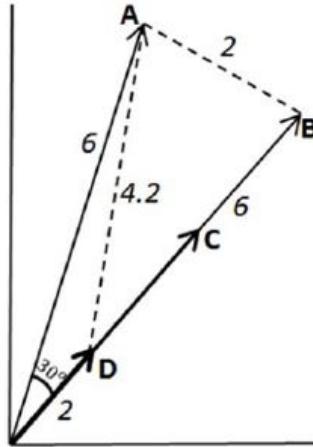


Slika 3.1: Primjer euklidske udaljenosti [7]

Kosinusna udaljenost je definirana kao jedan - skalarni produkt dva vektora podijeljen s umnoškom njihovih normi (3.2), a raspon vrijednosti je od 2 za vektore koji gledaju u suprotnim smjerovima do 0 za kolinearne vektore. Potencijalno negativna strana kosinusne udaljenosti je zanemarivanje duljine vektora. Primjer se može vidjeti na 3.2 gdje su točke B,

C i D jednako udaljene od točke A jer zatvaraju jednak kut, no kada bi se promatrala njihova euklidska udaljenost dobili bi različite rezultate.

$$\text{CosineDistance}(x, y) = 1 - \cos(\mathbf{x}, \mathbf{y}) = 1 - \frac{\mathbf{x} \cdot \mathbf{y}}{\|\mathbf{x}\| \cdot \|\mathbf{y}\|} \quad (3.2)$$



Slika 3.2: Primjer kosinusne udaljenosti [7]

Obje metrike su davale skoro identične rezultate uz male promjene poretku između vraćenih slika. Jedan od razloga zašto smo dobili takve rezultate je velika dimenzionalnost prostora koji vektori razapinju. Posljedica toga je da dvije točke koje se nalaze blizu u prostoru ujedno zatvaraju i maleni kut zbog čega se smatraju sličima u obje metrike.

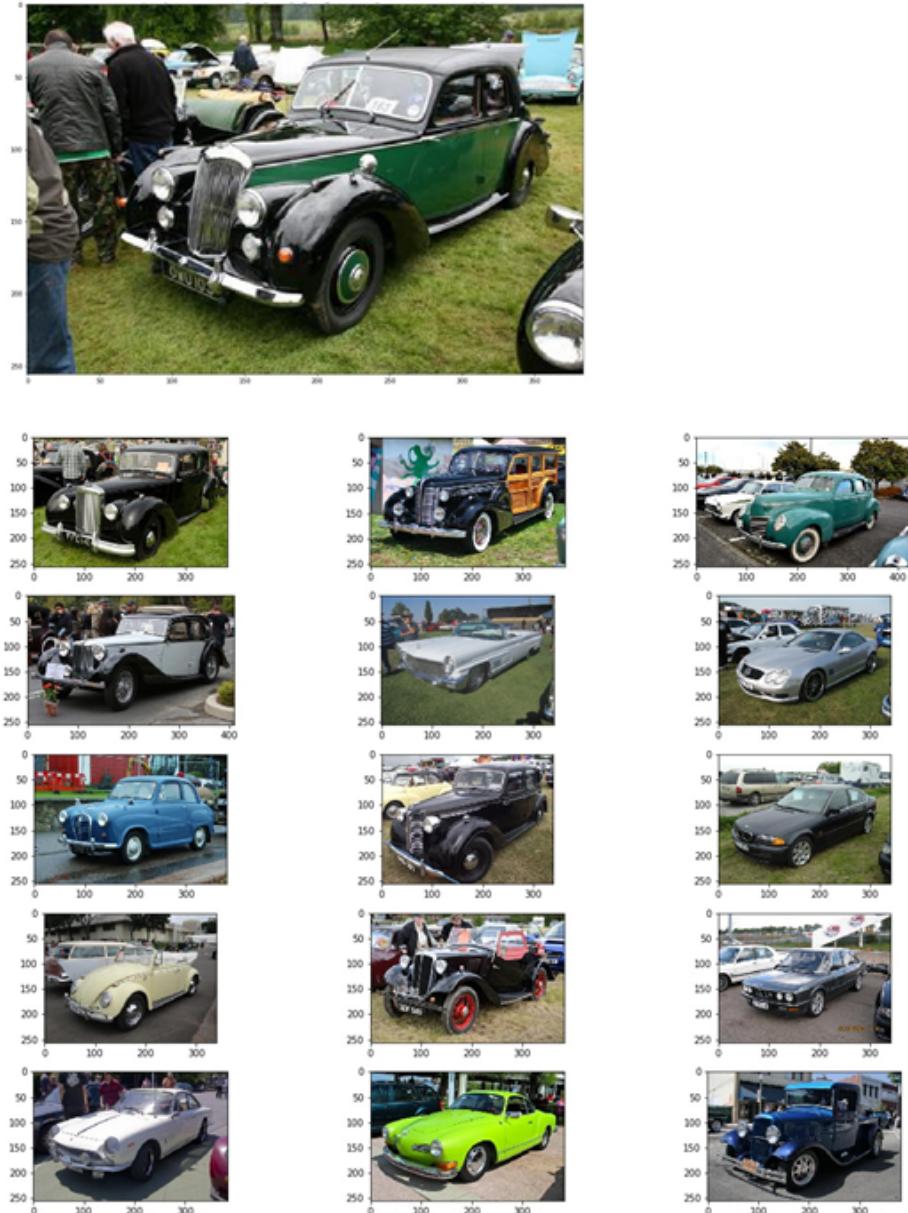
Nakon ispitivanja rezultata metrika odabrali smo kosinusnu udaljenost koja je davala zadovoljavajuće rezultate. Potencijalni problemi koji se mogu pojaviti kod te metrike u našem slučaju nisu pretjerano bitni budući da nam nije bitna duljina vektora već njegovo usmjerenje, odnosno relativan iznos aktivacija koje taj vektor opisuje. Primjerice ako našem sustavu predamo sliku mačke i tražimo ostale slične slike želimo vratiti sve vektore kojima su dominantni elementi koji detektiraju oči, njušku, uši mačke. Zbog toga nam je kosinusna udaljenost zadovoljavajuća jer vraća slike čiji reprezentativni vektori imaju jednake dominantne elemente unutar vektora.

## 3.2. Predtrenirana arhitektura VGG

VGG arhitektura je prva isprobana mreža za ekstrakciju značajki, točnije njen posljednji sloj konvolucijskog dijela mreže. Težinski koeficijenti mreže preuzeti su iz pobjedničke arhitekture ImageNet natjecanja čiji su filteri već naučeni prepoznavati komplikirane uzorke na ulaznim slikama.

Ulazna slika u mrežu je dimenzija 224x224x3, a finalni rezultat sloja kojeg koristimo je vektor od 25.000 elemenata. Budući da je krajnji cilj ostvariti reprezentaciju slike vektorom manjih dimenzija što bi omogućilo brže pretraživanje rezultat konvolucijske mreže se PCA modelom dalje reducira na 3000 dimenzija. Za treniranje PCA modela nasumično je izabrano 50.000 slika kako bi dobili što bolju procjenu glavnih komponenti.

Rezultati dobiveni na ovaj način za pojedine slike daju očekivane rezultate (Slika 3.3), dok za druge već za top 5 najsličnijih slika pojedini rezultati budu ne povezani (Slika 3.4).



**Slika 3.3:** Top 12 sličnih rezultata za sliku na vrhu



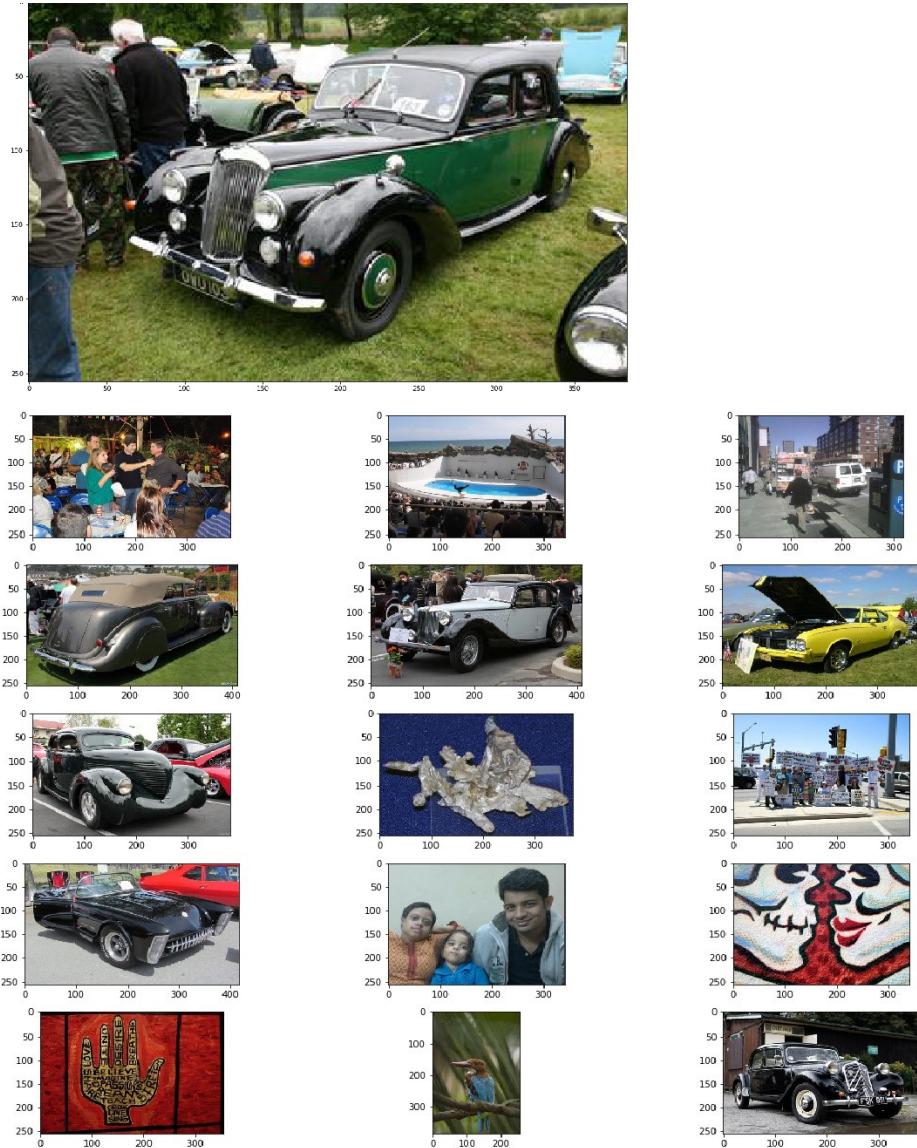
**Slika 3.4:** Primjer slike čije se slične traže (lijevo) i slika u top 5 koja nije ispravna (desno)

Dobri rezultati sažimanja, kao i lažni pozitivi dobiveni korištenjem jednostavnog VGG modela u dalnjim testiranjima koristili smo kao referencu koliko uspješno komplikiraniji modeli ekstrahiraju značajke iz ulazne slike.

### 3.3. Predtrenirana arhitektura Inceptionv4

Sljedeća arhitektura sa značajno većim kapacitetom je Inception v4. Kao i u slučaju arhitekture VGG i za Inception v4 smo preuzeli težinske koeficijente iz arhitekture naučene na skupu ImageNet.

Vođeni zaključcima koje smo dobili testirajući arhitekturu VGG za dohvaćanje sličnih slika prvi način ekstrakcije značajki, kao i u arhitekturi VGG, je bio uzimanjem izlaza zadnjeg sloja konvolucijske mreže. U ovom slučaju to je vektor od 1536 elemenata dobiven globalnim sažimanjem prosječnom vrijednošću. Dobivene vektore smo kasnije korištenjem PCA modela reducirali na 300 elemenata što je rezultiralo dosta lošim rezultatima unatoč velikom kapacitetu mreže (Slika 3.5).



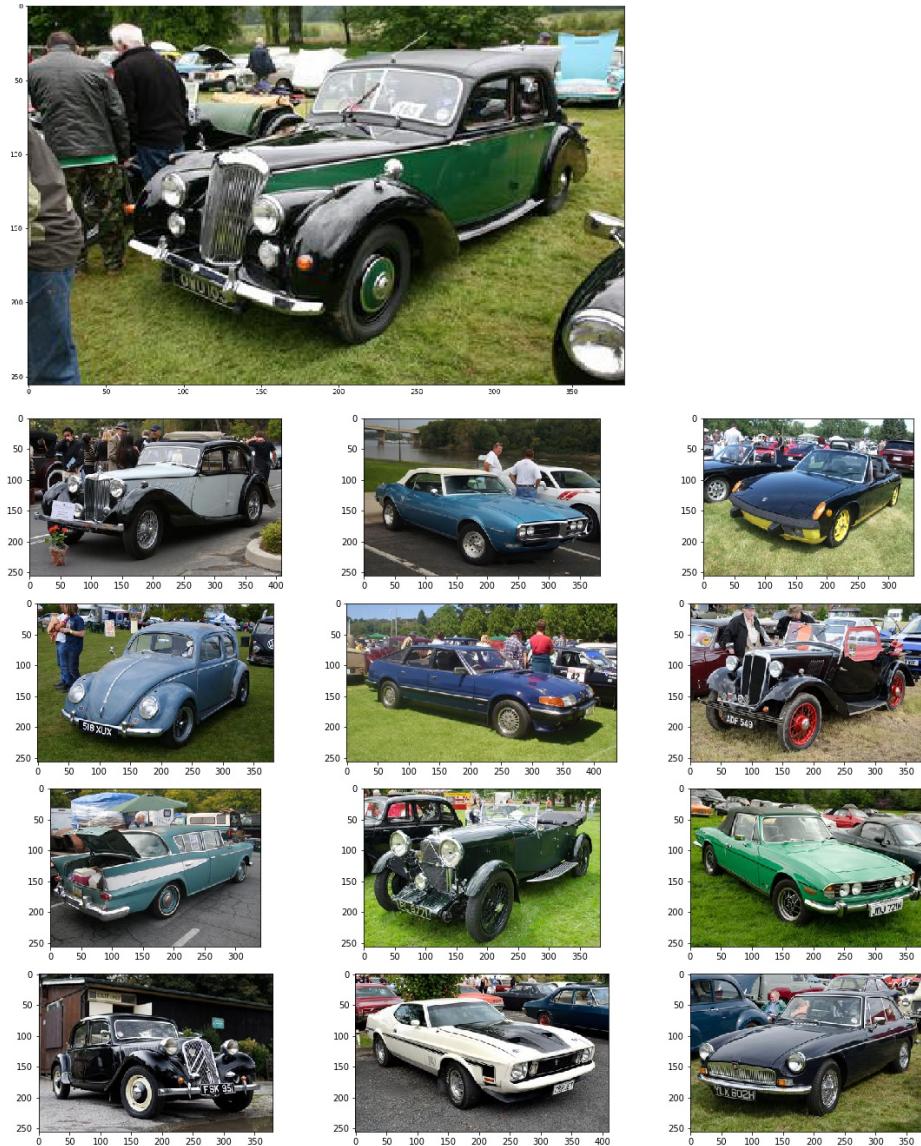
**Slika 3.5:** Top 12 sličnih rezultata za sliku na vrhu

Razlog zašto redukcija predzadnjeg sloja dovodi do ovoliko loših rezultata djelomično se može objasniti time što mreža taj sloj koristi za završnu klasifikaciju i time nema neku semantičku interpretaciju. Budući da je krajnji cilj dobiti vektor koji se sastoji od 300 elemenata gdje je intenzitet svakog elementa u vektoru označava pojavljivanje određenog uzorka na ulaznoj slici vektor odlučili smo značajke ekstrahirati iz svih konvolucijskih slojeva mreže.

Ekstrakcija značajki se vrši na sljedeći način, nad svakim konvolucijskim slojem u Inception mreži napravi se operacija globalnog sažimanja. Na primjer ako je izlaz konvolucijskog sloja  $12 \times 12 \times 512$  (visina, širina, broj kanala) globalnim sažimanjem svakog od kanala dobiti ćemo vektor  $1 \times 1 \times 512$ . Dobiveni vektori svakog konvolucijskog sloja nakon globalnog sažimanja se izravnaju i konkateniraju u jedan veliki vektor s  $\sim 16000$  elemenata. Tako

dobiveni elementi se zatim reduciraju na vektore od 300 elemenata korištenjem PCA modela. Kako bi nam rezultati bili što precizniji korišteni PCA model je prethodno istreniran korištenjem sto tisuća vektora od 16000 elemenata nasumično odabranih slika kako bi se što preciznije odredile glavne komponente podatkovnog skupa.

Primjer tako dobivenih rezultata može se vidjeti na sljedećim slikama 3.6, 3.7, 3.7, a korišteni kod za generiranje i dodatne rezultate moguće je vidjeti na [13].



**Slika 3.6:** Top 12 sličnih rezultata za sliku na vrhu



**Slika 3.7:** Top 12 sličnih rezultata za sliku na vrhu



**Slika 3.8:** Top 12 sličnih rezultata za sliku na vrhu

Kao što se vidi iz prikazanih slika unatoč reprezentiranja slike sa svega 300 elemenata što predstavlja višestruku uštedu prostora (3.1) vektor i dalje dobro ekstrahiru suštinu slike te slične slike grupira u bliske nakupine što je posebno vidljivo u sljedećem poglavljju.

**Tablica 3.1:** Primjer uštede prostora

Dimenzije slike	Ukupno elemenata	Faktor uštede
480p (12:9)	640x480	921 600
720p Widescreen	1280x720	2 764 800
1080p HD Widescreen	1920x1080	6 220 800

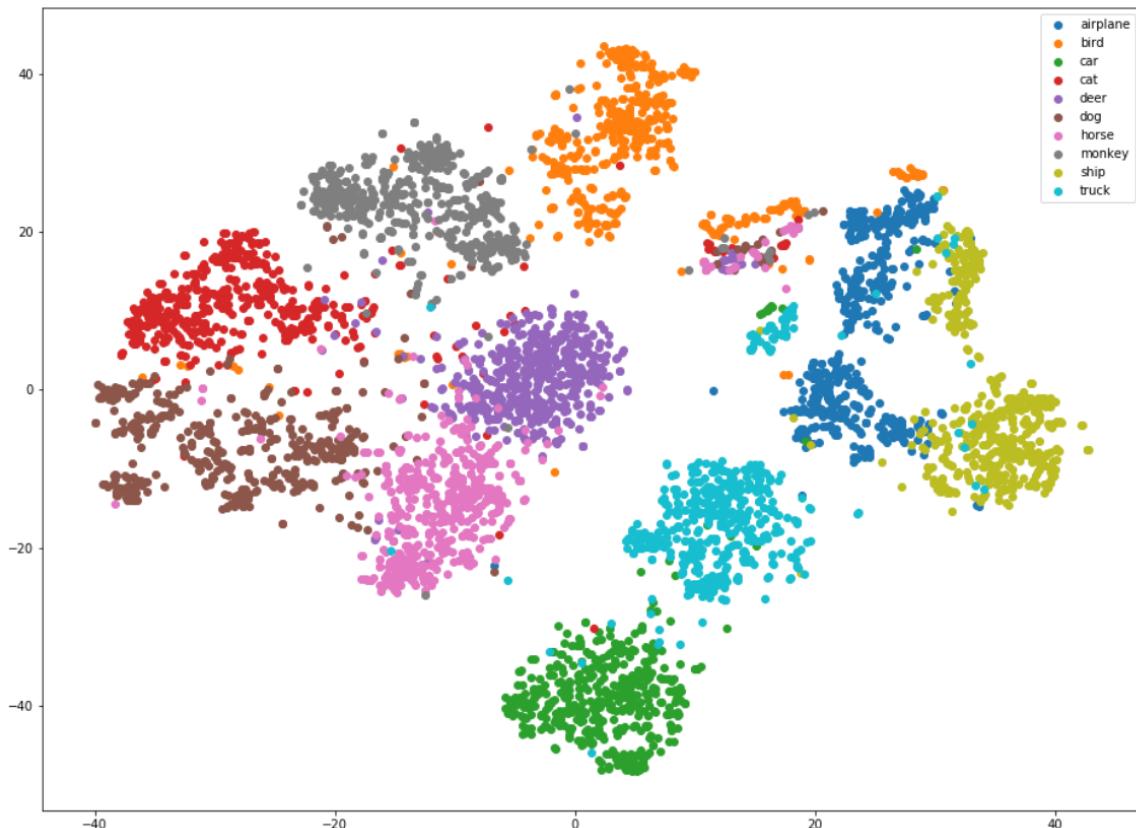
### 3.4. T-SNE prikaz smanjenog skupa podataka

Budući da korišteni podskup slika Googleov OpenImage [14] podatkovnog skupa velik (~1 milijun slika) i anotacije klase nisu nužno točne jer ih je većina generirana automatskom strojnom anotacijom korištenjem servisa poput Google Cloud Vision API za vizualizaciju smo odabrali manji skup podataka koji je anotiran ručno.

Odabrani skup podataka je STL-10 [15] čije slike imaju dimenziju 96x96x3, a nastao je po uzoru na podatkovni skup CIFAR-10 koji se obično koristi za vrednovanje modela.

Korištenjem metode t-SNE, značajke dobivene iz najboljeg modela Inception v4 s globalnim sažimanjem kanala su prikazane u 2D prostoru. Na slici 3.9 se vidi da su podaci grupirani u nakupine uz poneka miješanja grupa poput slika jelena i konja. Razlog tome može biti zbog vrlo slične pozadine slike i kuta snimanja gdje se značajke jelena (rogovi) ne vide jasno na slici zbog čega ih mreža lako može zamijeniti jer su obje životinje približno jednake boje.

Moguće je dodatno poboljšati performance mreže za određivanje sličnosti tako da se radi redukcija dimenzionalnosti modelom PCA na značajkama dobivenim iz klasa koje želimo preciznije razlikovati, u ovome slučaju klasa iz STL-10 podatkovnog skupa.



Slika 3.9: STL-10 dataset reprezentativni vektori prikazani u dvije dimenzije

## **4. Demo preporučitelj po sadržaju**

### **4.1. Neki tvoj section?**

## 5. Zaključak

Rezultati su pokazali da je korištenjem dubokih modela moguće značajno reducirati reprezentaciju slika na nisko dimenzionalan vektor koji čuva njenu semantiku. Takva redukcija ostvaruje veliku uštedu korištene radne memorije i zbog svoje male dimenzionalnosti omogućuje brzu usporedbu i dohvati sličnih slika preko njihovih reduciranih reprezentacija.

Budući da je u okviru ovog istraživanja korišten veliki i raznoliki skup podataka, kao i duboki modeli koji su trenirani na skupu podataka s drugačijom distribucijom od onoga na kojem je testiran to ostavlja prostor za dodatne eksperimente i naši rezultati određuju donju granicu performansi sustava. Razumno je pretpostaviti da bi se dodatna poboljšanja postigla kada bi se duboki model korišten za redukciju istrenirao na skupu podataka s istom distribucijom kao podaci koji će se koristiti u sustavu preporuke. Tako istrenirani model omogućio bi izbor manje dimenzije reprezentativnog vektora čime bi se ostvarile još bolje performanse sustava u vidu preciznosti i memorijskih/vremenskih zahtjeva pretrage.

Daljnja istraživanja, kao i napreci u području dubokog učenja ostavljaju mnogo prostora za detaljnije istraživanje dodatnih mogućnosti ekstrahiranja značajki.

## **6. Sažetak**

## **7. Summary**

# LITERATURA

- [1] Fei-Fei Li, Andrej Karpathy, and Justin Johnson. Cs231n: Convolutional neural networks for visual recognition 2016. URL <http://cs231n.stanford.edu/>.
- [2] Siniša Šegvić. Neslužbene stranice predmeta duboko učenje. URL <http://www.zemris.fer.hr/~ssegvic/du/>.
- [3] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. <http://www.deeplearningbook.org>.
- [4] Jonathon Shlens. A tutorial on principal component analysis. *CoRR*, abs/1404.1100, 2014. URL <http://arxiv.org/abs/1404.1100>.
- [5] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. URL <http://arxiv.org/abs/1409.1556>.
- [6] Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *CoRR*, abs/1602.07261, 2016. URL <http://arxiv.org/abs/1602.07261>.
- [7] A. Heidarian and M. J. Dinneen. A hybrid geometric approach for measuring similarity level among documents and document clustering. In *2016 IEEE Second International Conference on Big Data Computing Service and Applications (BigDataService)*, pages 142–151, March 2016. doi: 10.1109/BigDataService.2016.14.
- [8] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on

- heterogeneous systems, 2015. URL <https://www.tensorflow.org/>. Software available from tensorflow.org.
- [9] Itseez. Open source computer vision library. <https://github.com/itseez/opencv>, 2015.
  - [10] Leonid Boytsov and Bilegsaikhan Naidan. Engineering efficient and effective non-metric space library. In *Similarity Search and Applications - 6th International Conference, SISAP 2013, A Coruña, Spain, October 2-4, 2013, Proceedings*, pages 280–293, 2013. doi: 10.1007/978-3-642-41062-8\_28. URL [https://doi.org/10.1007/978-3-642-41062-8\\_28](https://doi.org/10.1007/978-3-642-41062-8_28).
  - [11] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. URL <http://www.jmlr.org/papers/v9/vandermaaten08a.html>.
  - [12] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.
  - [13] Toni Vlaić, Viran Ribić, and Lucija Šikić. Extracting deep features for image recommendation. <https://github.com/Mungosin/AVSP>, 2016.
  - [14] Ivan Krasin, Tom Duerig, Neil Alldrin, Vittorio Ferrari, Sami Abu-El-Haija, Alina Kuznetsova, Hassan Rom, Jasper Uijlings, Stefan Popov, Andreas Veit, Serge Belongie, Victor Gomes, Abhinav Gupta, Chen Sun, Gal Chechik, David Cai, Zheyun Feng, Dhyanesh Narayanan, and Kevin Murphy. OpenImages: A public dataset for large-scale multi-label and multi-class image classification. *Dataset available from https://github.com/openimages*, 2017.
  - [15] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In Geoffrey Gordon, David Dunson, and Miroslav Dudík, editors, *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15 of *Proceedings of Machine Learning Research*, pages 215–223, Fort Lauderdale, FL, USA, 11–13 Apr 2011. PMLR. URL <http://proceedings.mlr.press/v15/coates11a.html>.