Usporedba modela temeljenih na konvulucijskim neuronskim mrežama i transformer arhitekturama za detekciju objekata u prometu

Vareško, Lucia

Master's thesis / Diplomski rad

2024

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Pula / Sveučilište Jurja Dobrile u Puli**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:137:463147

SVEUČILIŠTE JURJA DOBRILE U PULI
FAKULTET INFORMATIKE

Lucia Vareško

Usporedba modela temeljenih na konvolucijskim neuronskim mrežama i transformer arhitekturama za detekciju objekata u prometu

DIPLOMSKI RAD

Pula, rujan, 2024. godine

SVEUČILIŠTE JURJA DOBRILE U PULI
FAKULTET INFORMATIKE

Lucia Vareško

Usporedba modela temeljenih na konvolucijskim neuronskim mrežama i transformer arhitekturama za detekciju objekata u prometu

DIPLOMSKI RAD

JMBAG: 0303082397, redoviti student
Studijski smjer: Informatika

Kolegij: Neuronske mreže i duboko učenje
Znanstveno područje : Društvene znanosti
Znanstveno polje : Informacijske i komunikacijske znanosti
Znanstvena grana : Informacijski sustavi i informatologija

Mentor: izv.prof.dr.sc. Goran Oreški

Pula, rujan, 2024. godine

Usporedba modela temeljenih na konvolucijskim neuronskim mrežama i transformer arhitekturama za detekciju objekata u prometu

Sažetak: Ovaj diplomski rad inspiriran je radom na ai.Shuttle projektu koji se provodi u okviru laboratorija Fakulteta informatike u Puli. Cilj projekta je kreirati autonomno vozilo koje bi prevozilo studente do sveučilišnih objekata, a kako bi to bilo moguće morali smo prvo napraviti detaljnu usporedbu modela za detekciju objekata. Rad istražuje dvije osnovne arhitekture dubokog učenja, konvolucijske neuronske mreže i transformere te analizira njihove najpoznatije modele poput AlexNeta, ResNeta, ViT-a i GPT-a. Osim teorijske analize, provelo se i istraživanje koje je za cilj imalo usporediti performanse različitih modela na stvarnim prometnim podacima. Rezultati ukazuju na specifične prednosti svake arhitekture, pružajući smjernice za buduća istraživanja i razvoj sustava za autonomnu vožnju.

Ključne riječi : usporedba performansi, detekcija objekata, prometni podaci, autonomna vožnja, konvolucijske neuronske mreže, transformeri

Comparison of Models Based on Convolutional Neural Networks and Transformer Architectures for Object Detection in Traffic

Abstract: This thesis is inspired by the work on the ai.Shuttle project, which is carried out within the laboratory of the Faculty of Informatics in Pula. The goal of the project is to create an autonomous vehicle that would transport students to university facilities, and in order for this to be possible, we first had to make a detailed comparison of object detection models. The paper investigates two basic architectures of deep learning, convolutional neural networks and transformers, and analyzes their most famous models such as AlexNet, ResNet, ViT and GPT. In addition to the theoretical analysis, research was also conducted with the aim of comparing the performance of different models on real traffic data. The results point to the specific strengths of each architecture, providing guidance for future research and development of autonomous driving systems.

Keywords : performance comparison, object detection, traffic data, self driving, convolutional neural networks, transformers

Sadržaj

1 Uvod

Razvoj autonomnih vozila predstavlja jednu od najvažnijih tehnoloških inovacija našeg doba koja bi mogla u potpunosti promijeniti sustav prometa kakav danas poznajemo. Percepcija autonomnog vozila ključna je za njegovo sigurno kretanje te interpretaciju elemenata u njegovom okruženju, uključujući pješake, druga vozila, prometne znakove i prepreke. Kako bi se vozilo moglo sigurno kretati urbanim područjima, potrebno je razviti sustav koji može analizirati složene prometne situacije u stvarnom vremenu i donositi ispravne odluke na temelju tih analiza. Tehnike detekcije objekata temeljene na dubokim neuronskim mrežama i umjetnoj inteligenciji omogućile su znatna poboljšanja u ovom području, zbog svoje sposobnosti učenja iz velikih količina podataka i postizanja visoke točnosti u raznim uvjetima.

Inspiracija za ovaj diplomski rad proizašla je iz rada na projektu ai.Shuttle koji se provodi u okviru laboratorija Fakulteta informatike u Puli, poznatog kao FIPULab. Cilj projekta je razvoj potpuno autonomnog mini autobusa koji će prevoziti studente do sveučilišnih objekata, koristeći napredne tehnike računalnog vida za detekciju objekata i razumijevanje okoline u kojoj se kreće. Ovaj projekt je financiran od strane Googlea i temelji se na suradnji između profesora, studenata i vanjskih stručnjaka pod vodstvom izv. doc. dr. sc. Gorana Oreškog. S obzirom na veliku dostupnost modela za detekciju objekata, jedan od ključnih izazova ovog projekta bio je upravo odabir modela prikladnog za detektiranje različitih objekata u prometu. Kako bi si olakšali tu odluku, odlučili smo napraviti usporedbu najpoznatijih arhitektura i modela za detekciju objekata te usporediti njihove performanse na stvarnim podatcima iz prometa.

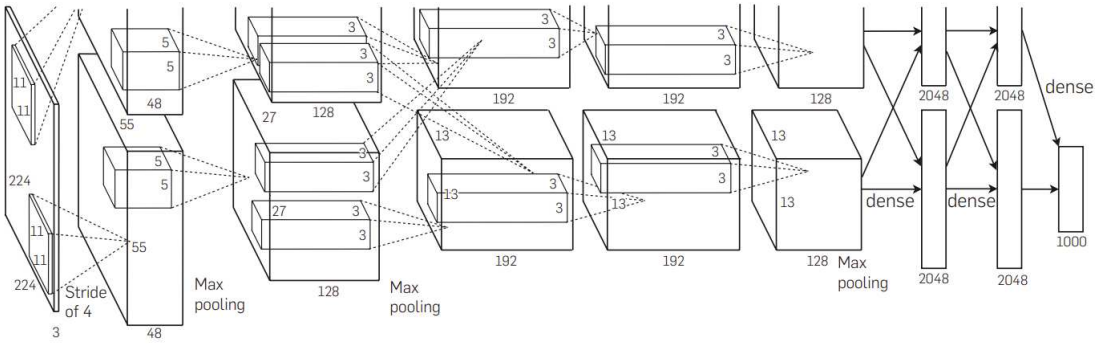
U prvom poglavlju diplomskog rada detaljno su opisane konvolucijske neuronske mreže s posebnim naglaskom na njezine ključne komponente, uključujući konvolucijski sloj, sloj sažimanja, potpuno povezani sloj i aktivacijske funkcije. Drugo poglavlje uključuje pregled nekih značajnih arhitektura temeljenih na konvolucijskim neuronskim mrežama: AlexNet, VVG i ResNet. Nakon temeljnog pregleda konvolucijskih neuronskih mreža, u poglavlju 4 prelazimo na analizu transformerskih arhitektura. Ovdje se fokusiramo na ključne pojmove transformer arhitekture te u sljedećem poglavlju prikazujemo nekoliko poznatih arhitektura: ViT, GPT-3 i GPT-4. Poglavlje 6 detaljno uspo-

ređuje ove dvije vrste arhitektura: konvolucijske neuronske mreže i transformere. Ova usporedba naglašava različite pristupe ovih arhitektura uz različite vizualizacije. Jednom kada su arhitektura teorijski objašnjene, u poglavlju 7 slijedi znanstveni rad koji se bavi usporedbom novih modela za detekciju objekata na podatcima iz prometa. Analizirat ćemo modele prema kriterijima točnosti, brzine izvođenja i sposobnosti detekcije objekata različitih veličina, s ciljem identificiranja onih koji postižu najbolje rezultate u specifičnim prometnim scenarijima, uključujući detekciju malih, srednje velikih i velikih objekata. Na kraju, rad završava detaljnim zaključkom u kojem se predstavljaju ključni nalazi istraživanja i daju preporuke za buduća istraživanja i primjene u ovom području.

2 Konvolucijske neuronske mreže

Arhitektura konvolucijskih neuronskih mreža temelji se na načelima vizualne percepcije gdje biološki neuroni odgovaraju umjetnim neuronima, a CNN jezgre funkcioniraju kao različiti receptori koji prepoznaju razne značajke. Aktivacijske funkcije simuliraju mehanizam u kojem na sljedeći neuron prelaze samo električni signali neurona koji prelaze određeni prag. Dodatno, kako bi konvolucijske neuronske mreže učile prema našim očekivanjima, razvijene su funkcije gubitka i optimizatori [17].

Konvolucijske neuronske mreže su se pokazale izuzetno uspješnima u zadacima kao što su klasifikacija slika i detekcija objekata, prvenstveno zahvaljujući njihovoj sposobnosti da automatski prepoznaju značajke unutar slika bez potrebe za ručnom ekstrakcijom [5].



Slika 1: Primjer arhitekture AlexNet [5]

Konvolucijske neuronske mreže se sastoje od nekoliko ključnih komponenti: konvolucijskih slojeva, slojeva sažimanja (*eng. pooling layers*) i potpuno povezanih slojeva (*eng. fully connected layers*). Prvi sloj konvolucijske neuronske mreže je obično konvolucijski sloj. Konvolucijski slojevi funkcioniraju kao receptori koji su osjetljivi na određene obrasce unutar slike, poput rubova i tekstura. Oni primjenjuju filtere na ulazne slike kako bi se izdvojili te značajke, dok aktivacijske funkcije kontroliraju koji se signali prosljeđuju na sljedeći sloj, ovisno o njihovoj važnosti. Kako podaci prolaze kroz slojeve mreže, model postupno uči prepoznavati složenije obrasce i oblike, sve do završnog sloja koji prepoznaje specifične objekte u Slici 1.

Slojevi sažimanja, reduciraju dimenzije podataka sažimanjem informacija, čime se smanjuje broj parametara i računska složenost modela, ali uz zadržavanje najvažnijih značajki. Nakon nekoliko iteracija kroz ove slojeve, konačni sloj, poznat kao potpuno povezani sloj, ”spljoštava” podatke i povezuje sve neurone iz prethodnog sloja sa svakim neuronom u završnom sloju, omogućujući modelu donošenje konačnih klasifikacijskih odluka [5].

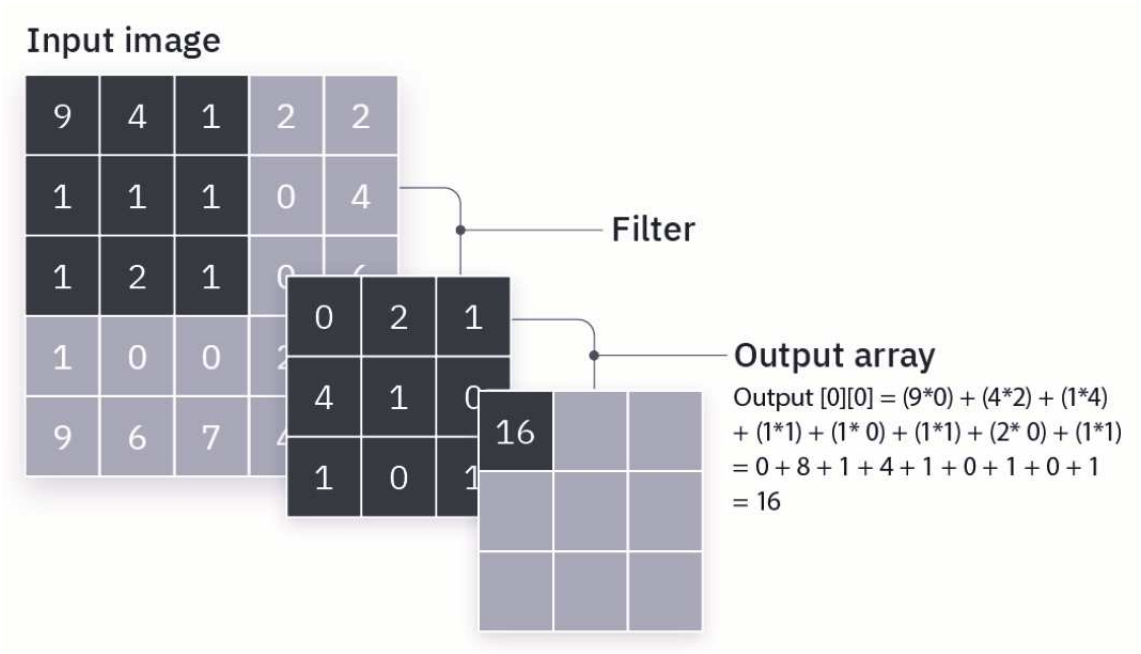
Potpuno povezani slojevi igraju ključnu ulogu u završnoj fazi klasifikacije, gdje se izdvojene značajke kombiniraju kako bi mreža mogla donijeti konačne predikcije za ulazne podatke. Funkcije gubitka i optimizatori koriste se za usmjeravanje učenja mreže, omogućujući konvolucijskim neuronskim mrežama da se prilagođavaju i poboljšavaju kroz iterativne procese treninga [17].

Ova višeslojna arhitektura omogućuje CNN-ovima da prepoznaju i klasificiraju složene obrasce unutar slikovnih podataka, što ih čini izuzetno učinkovitim za zadatke poput prepoznavanja objekata, klasifikacije slika i analize vizualnih podataka.

U nastavku ćemo obraditi konvolucijski sloj, sloj sažimanja, potpuni povezani sloj te na samom kraju aktivacijske funkcije koje se najčešće primjenjuju u ovakvim arhitekturama.

2.1 Konvolucijski sloj

Konvolucijski sloj (*eng. Convolutional Layer*) je ključna komponenta konvolucijskih neuronskih mreža u kojoj se odvija većina računskih operacija. Ovaj sloj primjenjuje operaciju konvolucije na ulazne podatke koristeći filtere (poznate i kao jezgre ili kerneli) kako bi detektirao različite značajke unutar slike, poput rubova, tekstura, oblika pa čak i složenijih uzoraka. Konvolucijski sloj može prepoznati slike čak i kada su one pomaknute, smanjene ili rotirane, sve dok su te slike prepoznatljive ljudskom oku.



Slika 2: Primjer operacije konvolucije [5]

Konvolucijski sloj koristi više filtera kako bi istovremeno generirao različite značajke. Na primjer, jedan filter može biti specijaliziran za prepoznavanje horizontalnih rubova, dok drugi može prepoznavati vertikalne rubove ili specifične teksture. Filteri su obično veličine 3×3, a njihove težine inicijalizirane su nasumično i ažuriraju se tijekom treninga. Više filtera omogućava mreži da istovremeno prepoznaje razne aspekte slike, čime se poboljšava sposobnost mreže da razumije kompleksnije značajke.

Svaki filter provodi lokalnu prostornu konvoluciju preko ulaznih podataka, a rezultat konvolucije je skalarni produkt koji prikazuje raspodjelu detektiranih značajki na slici. Ovaj skalarni produkt nazivamo mapa značajki (*eng. feature map*) ili aktivacijska mapa.

Filter se pomiče kroz sliku koristeći parametar korak (*eng. stride*). Korak definira za koliko se piksela filter pomiče nakon svake operacije, a promjena ove vrijednosti može utjecati na gustoću i veličinu izlazne mape značajki. Veći korak smanjuje veličinu izlaza i dovodi do veće kompresije slike. Kako bi se smanjio gubitak informacija, osobito na rubovima slike, koristi se popunjavanje (*eng. padding*). Popunjavanje dodaje piksele s vrijednošću 0 oko rubova slike kako bi se očuvale informacije koje bi inače bile izgubljene.

Za izračunavanje veličine izlaza mape značajki uz popunjavanje, koristi se sljedeća formula:

$$W_{out} = \frac{W - F + 2P}{S} + 1$$

gdje je W dimenzija ulaza, F veličina filtera, P veličina popunjavanja i S veličina koraka.

Konvolucijski slojevi omogućuju mreži da prepoznaje sve složenije značajke kroz slojeve, čime se postiže dublje razumijevanje slike. Svaki dodatni sloj u mreži omogućava apstrakciju početnih značajki u složenije oblike, čime se mreža približava ljudskom razumijevanju vizualnih informacija.

2.2 Sloj sažimanja

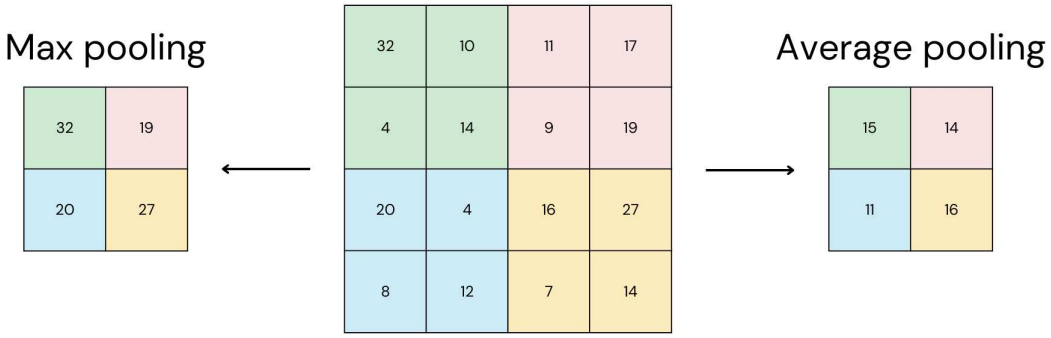
Sloj sažimanja (*eng. Pooling Layer)* je sloj koji se često koristi kako bi se smanjile dimenzije aktivacijskih mapa iz prethodnog sloja konvolucijske mreže. Ovaj sloj koristi funkciju sažimanja koja na na temelju statističkih podataka u lokalnoj okolini mijenja izlazni rezultat mreže. Dvije najčešće metode sažimanja su sažimanje maksimalnih vrijednosti (*eng. max pooling*) i prosječno sažimanje (*eng. average pooling*). Sažimanje maksimalnih vrijednosti uzima najveću vrijednost unutar regije, dok prosječno sažimanje računa prosjek svih vrijednosti. Obje metode smanjuju veličinu značajki koje mreža mora obraditi, čime pojednostavljuju model, poboljšavaju njegovu sposobnost generalizacije i smanjuju rizik *overfittinga*.

Veličina izlaza može se izračunati prema formuli:

$$W_{out} = \frac{W - F}{S} + 1$$

gdje je W dimenzija ulaza, F veličina filtera i S veličina koraka.

Na primjer, uzmimo matricu 4×4 koja predstavlja značajke slike. Ako primijenimo sažimanje maksimalnih vrijednosti s filterom veličine 2×2 i korakom 2, filter će se pomicati preko matrice, za svaku poziciju odabrati najveću vrijednost iz područja koje pokriva i zapisati je u rezultantnoj matrici. Time se aktivacijska mapa smanjuje za faktor 2. Ovaj postupak se ponavlja za sva područja matrice, što rezultira manjom matricom koja još uvijek sadrži ključne informacije iz izvorne slike. Proces je sličan za prosječno sažimanje, pri čemu se umjesto najveće vrijednosti uzima prosječna vrijednost. Jednostavan primjer sažimanja prikazan je na Slici 3.



Slika 3: Primjer sažimanja maksimalnih vrijednosti i prosječnog sažimanja

2.3 Potpuno povezani sloj

Za razliku od konvolucijskih slojeva, gdje su neuroni povezani samo u lokalnoj regiji ulaznih podataka, neuroni u potpuno povezanom sloju (*eng. Fully Connected Layer, FC*) imaju potpunu povezanost sa svim aktivacijama iz prethodnog sloja. Ovaj sloj obično se nalazi pri kraju mreže i odgovoran je za generiranje konačnih izlaznih predikcija.

Aktivacije u ovom sloju računaju se matričnim množenjem ulaznih vrijednosti s težinama sloja, uz dodatak pristranosti (*eng. bias*). Na kraju se obično primjenjuje aktivacijska funkcija koja pomaže u uklanjanju negativnih vrijednosti i uvodi nelinearnost u model, omogućujući složenije modeliranje i preciznije predikcije. Više o aktivacijskim funkcijama u sljedećoj cjelini.

Matematički, aktivacija u potpuno povezanom sloju može se izraziti kao:

$$Z = f(X \cdot W + b)$$

gdje je X ulazni vektor, W matrica težina, b pristranost, a f aktivacijska funkcija. Nakon što su aktivacije izračunate, model koristi te vrijednosti za generiranje konačnih predikcija, kao i za izračunavanje gubitka i ažuriranje parametara tijekom procesa učenja. Općenito, potpuno povezani sloj uči globalne obrasce i zahtijeva veći broj parametara, čime doprinosi složenosti modela.

2.4 Aktivacijske funkcije

Aktivacijske funkcije su ključne komponente u neuralnim mrežama jer uvode nelinearnost u model, što omogućava mrežama da uče složene obrasce i odnose u podacima. Bez nelinearnosti, mreža bi se mogla ponašati kao linearni model, što značajno ograničava njezinu sposobnost učenja kompleksnih funkcija. Funkcije aktivacije sigmoid, tanh i ReLU često se koriste u praksi, a objašnjenja ovih funkcija ovdje su temeljena na izvoru [17].

Sigmoidna funkcija $\sigma(x)$ preslikava realne brojeve u interval između 0 i 1, a definirana je kao:

$$\sigma(x) = \frac{1}{1 + e^{-x}}$$

Ova se funkcija često koristi u binarnim klasifikacijama i kao aktivacijska funkcija u slojevima mreže gdje je potrebna vjerojatnost izlaza. Međutim, zbog asimptotskih granica pri ekstremnim vrijednostima x (blizu 0 ili 1), sigmoid može izazvati problem nestajanja gradijenata gdje mali gradijenti usporavaju ili zaustavljaju učenje tijekom propagacije unatrag. Također, izlazi nisu centrirani oko nule, što je vidljivo na grafu sigmoidne funkcije na Slici 4.

Tanh funkcija $\tanh(x)$ preslikava realne brojeve u interval između -1 i 1, a definirana je kao:

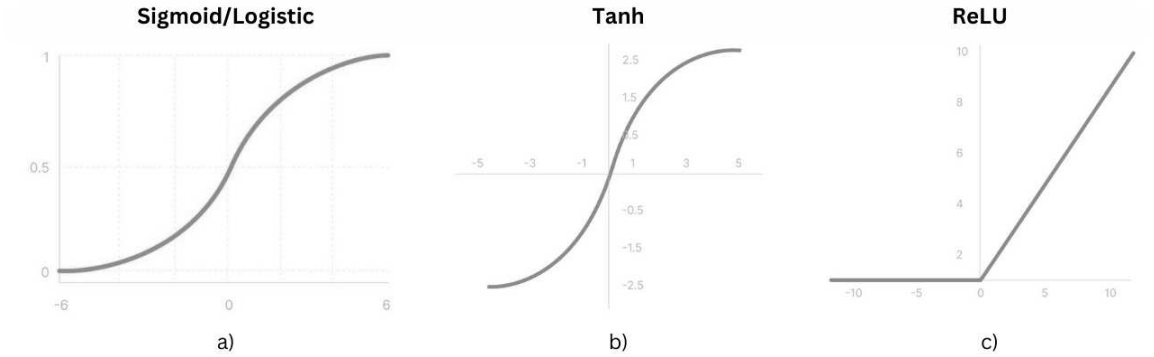
$$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

Za razliku od sigmoid funkcije, izlaz funkcije tanh centriran je oko 0. Ova centriranost može značajno poboljšati brzinu učenja i konvergenciju modela jer smanjuje problem nestajanja gradijenta i pomaže u održavanju većih gradijenata tijekom procesa učenja. Funkcija tanh se često koristi u skrivenim slojevima neuronskih mreža, sekvencijskim modelima, autoenkoderima i generativnim modelima kako bi pružila centrirane aktivacije koje pomažu bržem konvergiranju i boljem učenju. Graf tanh funkcije nalazi se na Slici 4.

ReLU $f(x)$ je jedna od najčešće korištenih aktivacijskih funkcija u dubokim neuronskim mrežama zbog svoje jednostavnosti i efikasnosti, a definirana je kao:

$$f(x) = \max(0, x)$$

Ova funkcija omogućava brže treniranje i smanjuje problem nestajanja gradijenata jer za pozitivne ulaze vraća izravno ulaznu vrijednost, dok negativne ulaze postavlja na 0. Međutim, ReLU može izazvati problem poznat kao ”umiranje ReLU” gdje neuroni prestaju učiti ako su njihovi ulazi konstantno negativni. Graf ReLU funkcije nalazi se na Slici 4.



Slika 4: Grafovi aktivacijskih funkcija [16]

Svaka od ovih funkcija ima svoje prednosti i nedostatke te se njihova primjena odabire ovisno o specifičnostima zadatka i arhitekturi mreže. Pravilno odabrane aktivacijske funkcije mogu značajno poboljšati performanse modela i ubrzati proces učenja.

3 Pregled arhitektura temeljenih na konvolucijskim neuronskim mrežama

U ovom poglavlju razmatramo neke od ključnih arhitektura temeljenih na konvolucijskim neuronskim mrežama. Fokusirat ćemo se na tri istaknute arhitekture koje su oblikovale smjer razvoja dubokog učenja: AlexNet, VGG i ResNet. Prikazat ćemo ih kronološkim redoslijedom kako bismo istaknuli njihov razvoj te međusobne razlike.

3.1 AlexNet

AlexNet, arhitektura koju su 2012. godine razvili Krizhevsky, Sutskever i Hinton [15], označila je prekretnicu u području dubokog učenja i računalnog vida. Ova je mreža bila jedna od prvih dubokih konvolucijskih neuronskih mreža koja je ostvarila značajan uspjeh na natjecanju ImageNet Large Scale Visual Recognition Challenge (ILSVRC-2012) sa stopom pogreške od 15.3%, dok je drugoplasirana mreža ostvarila 26.2%. Time je AlexNet pokazao veliki potencijal primjene konvolucijskih neuronskih mreža na zadatcima prepoznavanja slika koji podrazumijevaju velike skupove podataka.

AlexNet se sastoji od 5 konvolucijskih slojeva, praćenih s 3 potpuno povezana sloja, čime mreža omogućuje prepoznavanje složenih obrazaca u slikama. Svaki konvolucijski sloj primjenjuje ReLU aktivacijsku funkciju, koja je ubrzala proces učenja u odnosu na tradicionalne aktivacijske funkcije poput sigmoidne ili tanh. U svrhu smanjenja dimenzionalnosti podataka i sprječavanja prenaučenosti, AlexNet koristi slojeve maksimalnog sažimanja, dok *dropout* tehnika u prva dva povezana sloja smanjuje rizik od prevelike prilagodbe modela treniranim podacima. *Droput* tehnika podrazumijeva nasumično isključivanje neurona s vjerojatnošću 0.5 što doprinosi boljoj generalizaciji.

U odnosu na kasnije arhitekture, AlexNet koristi veće filtere u konvolucijskim slojevima. Prvi sloj primjenjuje filtere veličine 11×11, dok kasniji slojevi koriste filtere manjih dimenzija 5×5 i 3×3. Ova kombinacija omogućuje modelu da prvo prepozna globalne obrasce, a zatim se fokusira na finije detalje.

AlexNet ima preko 60 milijuna parametara, što predstavlja značajno povećanje u usporedbi s prijašnjim arhitekturama, te zahtijeva značajne računalne resurse za treniranje. Za razliku od prijašnjih arhitektura koje su za treniranje koristile CPU (*eng. Central processing Unit*), za treniranje AlexNet mreže korištena su dva GPU-a (NVIDIA GTX 580) što je omogućilo paralelno procesiranje ta tako značajno ubrzalo trening. Tijekom treninga su se koristile i različite tehnike augmentacije poput translacije i horizontalne refleksije za dodatno povećanje skupa podataka.

Ova je arhitektura postavila temelj za daljnji razvoj dubokih konvolucijskih mreža, a njeni inovativni pristupi poput ReLU aktivacije i *dropouta* danas se široko koriste u modernim arhitekturama. Krizhevsky, Sutskever i Hinton zaključili su da je dubina mreže presudna za uspješno prepoznavanje kompleksnih obrazaca u slikama, što je promijenilo smjer istraživanja u području dubokog učenja i računalnog vida.

3.2 VGGNet

Kako bi poboljšali sposobnost modela da prepoznaju kompleksne veze među podatcima, znanstvenici su počeli eksperimentirati s dubinom mreže. Tako su Simonyan i Zisserman 2014. godine predstavili novu arhitekturu koja se temelji na VGG blokovima (*eng. Visual Geometry Group*) [29], fokusirajući se na povećanje dubine mreže uz korištenje jednostavnih i uniformnih konvolucijskih slojeva. Ova arhitektura značajno je poboljšala rezultate prethodnih modela, poput AlexNeta, pokazujući da dublje mreže mogu bolje prepoznati složenije značajke u slikama. Na ILSVRC-2014 natjecanju, VGG arhitektura osvojila je drugo mjesto u klasifikaciji i prvo mjesto u lokalizaciji objekata, što je dodatno potvrdilo njen značajan napredak u odnosu na prethodne modele.

U svom radu, Simonyan i Zisserman, predstavili su više varijacija VGG arhitekture: VGG11, VGG13, VGG16 i VGG19, gdje brojevi predstavljaju broj slojeva u arhitekturi. Na primjer, VGG16 ima 13 konvolucijskih slojeva te 3 potpuno povezana sloja, dok VGG19 ima 16 konvolucijskih slojeva te također 3 potpuno povezana sloja. Budući da su VGG16 i VGG19 dvije najpoznatije varijacije, u nastavku ćemo se fokusirati na njih.

VGG arhitekture se prvenstveno sastoje od konvolucijskih slojeva ispresijecanih slojevima za maksimalno sažimanje, nakon čega slijede potpuno povezani slojevi. Konvolucijski slojevi ovih arhitektura koriste vrlo male filtere dimenzije 3×3, čijom se primjenom kroz više konvolucijskih slojeva simulira receptivno ponašanje polja filtera većih veličina. Na primjer, primjenom konvolucijskog sloja s filterom veličine 7×7 i korakom 1 dobit ćemo izlaznu mapu veličine 1×1. Isti rezultat možemo dobiti s tri konvolucijska sloja koji koriste filter veličine 3×3. Prostorne dimenzije volumena smanjuju se prolaskom kroz dublje slojeve mreže, dok se dubina volumena povećava zbog većeg broja filtera, što omogućuje mreži da uhvati sve složenije značajke i apstraktne obrasce iz podataka.

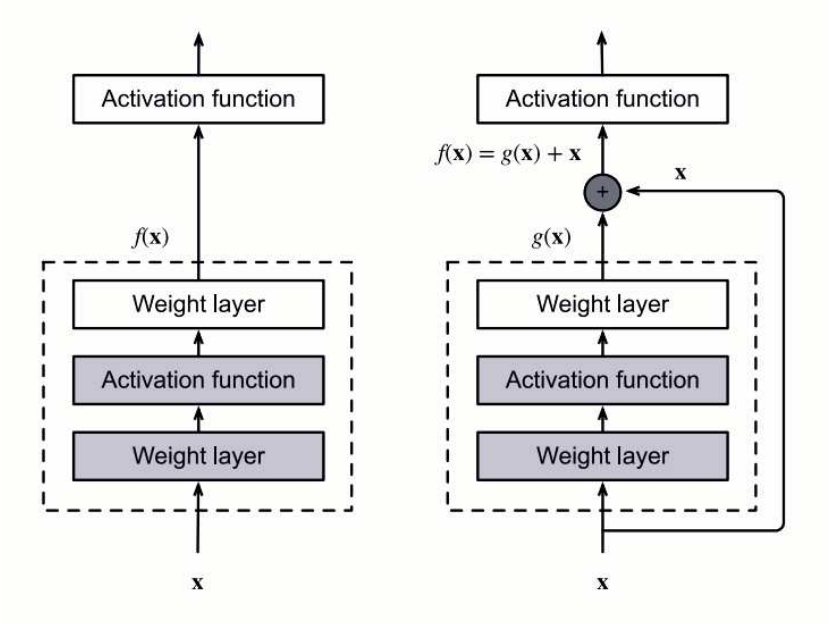
Što se tiče broja parametara, VGG16 ima oko 138 milijuna parametara, dok VGG19 ima približno 144 milijuna parametara. Ovaj velik broj parametara znači da VGG arhitekture zahtijevaju značajne računalne resurse i upravo to predstavlja njihov najveći nedostatak. Unatoč tome, VGG modeli su postali široko korišteni u zadatcima poput klasifikacije i lokalizacije objekata u slikama.

3.3 ResNet

Vrlo brzo nakon uspjeha VGG mreže, pojavila se nova ResNet arhitektura koja je napravila značajan korak u razvoju dubokih neuronskih mreža. ResNet arhitekturu razvili su He et al. s ciljem rješavanja problema ”nestajanja gradijenata” koji se javlja kod treniranja dubokih mreža. Ova je arhitektura ostvarila veliki uspjeh na ILSVRC-2015 natjecanju, gdje je pobijedila u kategoriji klasifikacije objekata sa stopom pogreške od samo 3.57% [11].

Ključ uspjeha ResNet arhitekture leži u novom konceptu rezidualnih blokova (*eng. residual blocks*). Rezidualni blokovi koriste preskočne veze (*eng. skip connections*) koje omogućuju gradijentima da preskoče jedan ili više slojeva i tako se izravno propagiraju kroz mrežu. Na Slici 5 možemo vidjeti klasičan i rezidualni blok, kod kojeg se ulaz razdvaja na dvije grane: prva grana prolazi kroz nekoliko konvolucijskih slojeva, dok druga grana predstavlja preskočnu vezu koja zaobilazi te slojeve i koristi funkciju identiteta. Na kraju se izlazi obje grane zbrajaju i na njih se primjenjuje aktivacijska

funkcija kako bi se postigla nelinearnost. Primjenom preskočnih veza se osigurava prijenos važnih značajki do završnih slojeva.



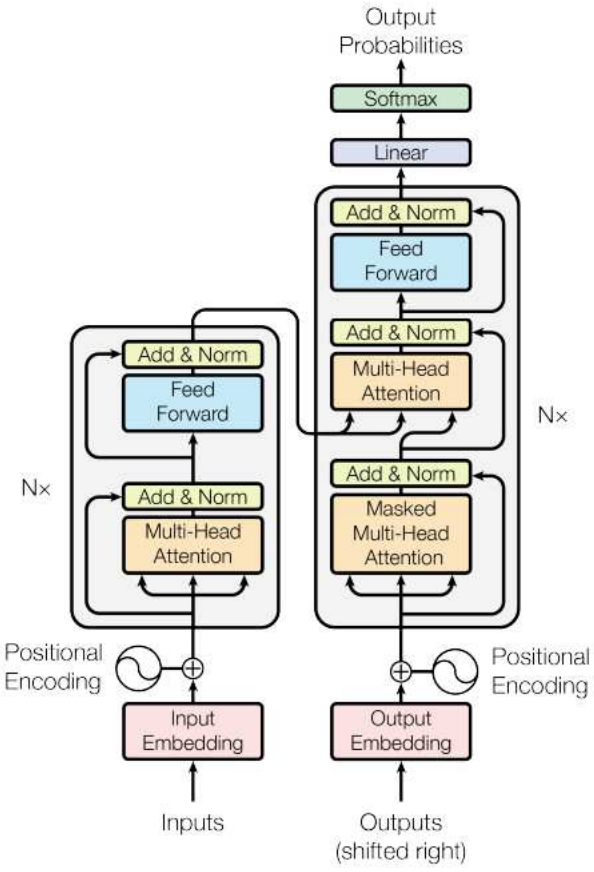
Slika 5: Usporedba običnog bloka (desno) i rezidualnog bloka (lijevo) [42]

ResNet arhitektura se sastoji od više rezidualnih blokova gdje svaki blok ima konvolucijski sloj 3×3. Također, na samom početku arhitekture nalazi se dodatni konvolucijski sloj *stem* koji obrađuje ulazne podatke prije nego što prođu kroz glavno tijelo mreže. Na kraju mreže nema uobičajenih potpuno povezanih slojeva, već se koristi samo jedan potpuno povezani sloj za klasifikaciju u 1000 klasa. Resnet arhitektura također dolazi u raznim varijacijama, a najpoznatije su: ResNet50, ResNet101 i ResNet152, gdje brojevi predstavljaju broj slojeva u arhitekturi. ResNet50 ima preko 25 milijuna parametara, ResNet101 ima preko 44 milijuna parametara, dok ResNet152 ima preko 50 milijuna parametara.

Uvođenjem rezidualnih blokova, ResNet arhitekture pokazale su kako se duboke mreže mogu uspješno trenirati bez gubitka performansi. Ovaj pristup ne samo da je omogućio veće dubine mreža, već je otvorio put razvoju još naprednijih arhitektura koje se i danas koriste u raznim područjima primjene dubokog učenja.

4 Transformeri

Transformerske arhitekture su prvi put uvedene u području obrade prirodnog jezika 2017. godine, ali njihov se utjecaj vrlo brzo proširio i na područje računalnog vida. Već 2020. godine, Dosovitskiy et al. odlučili su primijeniti standardnu arhitekturu transformera izravno na slike, uz manje prilagodbe arhitekture [8]. Umjesto klasičnog pristupa konvolucijskim operacijama, sliku su podijelili na manje dijelove ili *patcheve*, te sekvencu linearnih ugrađivanja tih *patcheva* proslijedili kao ulaz u transformeru. Na taj se način slikovni podatci mogu tretirati jednako kao tokeni, tj. riječi kod prirodne obrade jezika.



Slika 6: Arhitektura transformera [35]

Za razliku od konvolucijskih neuronskih mreža, transformeri predstavljaju arhitekturu koja se oslanja na mehanizam pozornosti (*eng. attention mechanism*) kako bi prepoznala složene odnose i dugoročne ovisnosti unutar slike. Ovisno o arhitekturi, transformeri mogu koristiti enkoder, dekoder ili oboje. Enkoder obrađuje ulaznu sekvencu i pretvara ju u skup kontekstualnih vektora, koji se zatim koriste za generiranje izlazne sekvence u dekoderu. Ovaj pristup omogućava modelima da učinkovitije prepoznaju globalne značajke unutar slike, za razliku od lokalnih značajki koje detektiraju konvolucijske neuronske mreže.

Kao što je već spomenuto, transformeri tretiraju slike kao niz manjih *patcheva*, koji se ugrađuju u vektore kroz modul za promjenu reprezentacije slike. Ovi vektori zatim čine ulaznu sekvencu za slojeve transformera. Svaki sloj transformera sastoji se od mehanizma višestruke pažnje (*eng. Multi-Head Self-Attention*), koji omogućava modelu simultano analiziranje različitih aspekata slike. Nakon što mehanizam pažnje obradi podatke, oni prolaze kroz naprijed-usmjereni neuronski sloj (*eng. feed-forward neural network*), koji dodatno rafinira informacije. Konačno, rezidualne veze s normalizacijom slojeva osiguravaju stabilnost treninga i omogućuju modelu učenje složenih reprezentacija.

Primjenom ovih principa, transformeri su postigli *state-of-the-art* rezultate u prepoznavanju slika, čime su pokazali svoju superiornost u odnosu na tradicionalne pristupe temelje na konvolucijskim neuronskim mrežama.

U nastavku ćemo se detaljnije fokusirati na dva glavna inovativna dijela transformera: modul promjenu reprezentacije slike i mehanizam pažnje, koji omogućuju modelu da učinkovitije obrađuje slikovne podatke.

4.1 Modul za promjenu reprezentacije slike

Transformeri zahtijevaju da ulazni podaci budu u sekvencijalnom formatu, što predstavlja izazov kada se radi o slikama koje su obično predstavljene kao dvodimenzionalne mreže piksela kroz tri kanala boja (RGB). Modul za promjenu reprezentacije slike (*eng. Patch Embedding Modul*) rješava ovaj izazov transformirajući sliku u sekvencu *patcheva*, pri čemu se svaki *patch* tretira kao token u sekvenci. Ovaj proces je ključan jer omogućuje transformeru da obrađuje slikovne podatke (*patcheve*) na sličan način kao što obrađuje sekvence riječi, tj. tokene u NLP zadacima [8].

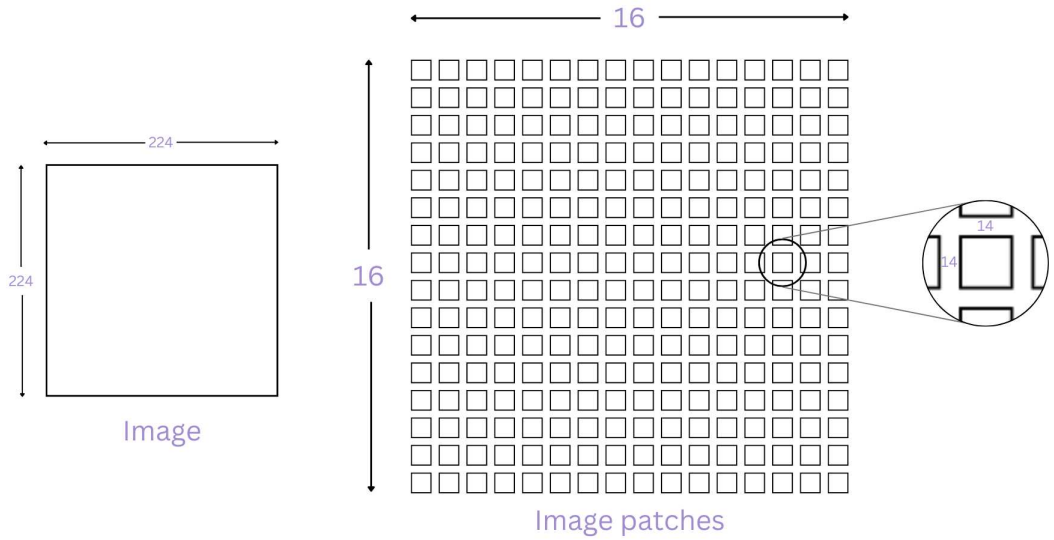
Štoviše, kad god je moguće uzeti signal i razdijeliti ga na dijelove te svaki od tih dijelova projicirati u vektor fiksne dimenzije, tada se podaci mogu tokenizirati. Kada su podaci tokenizirani, oni predstavljaju sekvencu vektora.

Osim transformacije slike u sekvencijalni format, modul za promjenu reprezentacije slike ima još dvije uloge: pozicijsko kodiranje i uključivanje CLS tokena (*eng. Classifier token)*.

4.1.1 Dijeljenje slike na *patcheve*

Ako uzmemo u obzir da mehanizam pažnje koristi skalarni umnožak, kada bi umjesto tokena modelu proslijedili cijeli skup piksela slike, slika od 224×224 piksela zahtijevala bi 224^4 usporedbi, tj. 2,5 milijarde usporedbi za samo jedan sloj mehanizma pažnje. No, ako tu istu sliku podijelimo na *patcheve* veličine 14×14 piksela, imat ćemo 256 *patcheva*, te će tada jedan sloj mehanizma pažnje zahtijevati samo 9,8 milijuna usporedbi.

Ovaj primjer jasno pokazuje da tokenizacija slike, odnosno dijeljenje slike na *patcheve*, omogućuje značajno smanjenje složenosti i čini mehanizam pažnje primjenjivim na slike realnih dimenzija.



Slika 7: Grafički prikaz dijeljenja slike na *patcheve*

Jednom kada imamo *patcheve*, oni se preslikavaju u jednodimenzionalne vektore veličine $P^2 \times C$, gdje je P veličina *patcha*, a C broj kanala boje. Primjerice, *patch* veličine 14×14 piksela s tri kanala boja bit će predstavljen kao vektor veličine 588.

4.1.2 Pozicijsko kodiranje

Transformeri obrađuju cijelu ulaznu sekvencu istovremeno i tretiraju je kao neuređeni skup. Međutim, u kontekstu slika, prostorni raspored *patcheva* je ključan za pravilnu interpretaciju slike. Kako bi model znao redoslijed *patcheva* i kako bi mogao pravilno interpretirati sliku, koristi se pozicijsko kodiranje (*eng. Positional embedding*). Pozicijsko kodiranje je tehnika kojom se svakom elementu ulaznog niza dodjeljuje njegova reprezentacija pozicije u obliku vektora. Pozicijski vektori uče se tijekom treninga i prilagođavaju zajedno s ostalim parametrima, te se zbrajaju s pripadajućim *patch* ugrađivanjima, kako bi se dobila konačna sekvenca ugrađivanja. Na taj način, pozicijski vektori omogućuju modelu prepoznavanje relativne i apsolutne pozicije *patcheva* unutar slike, osiguravajući očuvanje lokalnosti i prostornog rasporeda informacija [35].

4.1.3 Uključivanje CLS tokena

Ideja CLS tokena (*eng. Classifier token)* dolazi iz jezičnog modela BERT gdje ovaj specijalni token predstavlja cijeli ulazni niz ili rečenicu i dodaje se na početak svake unesene rečenice [7].

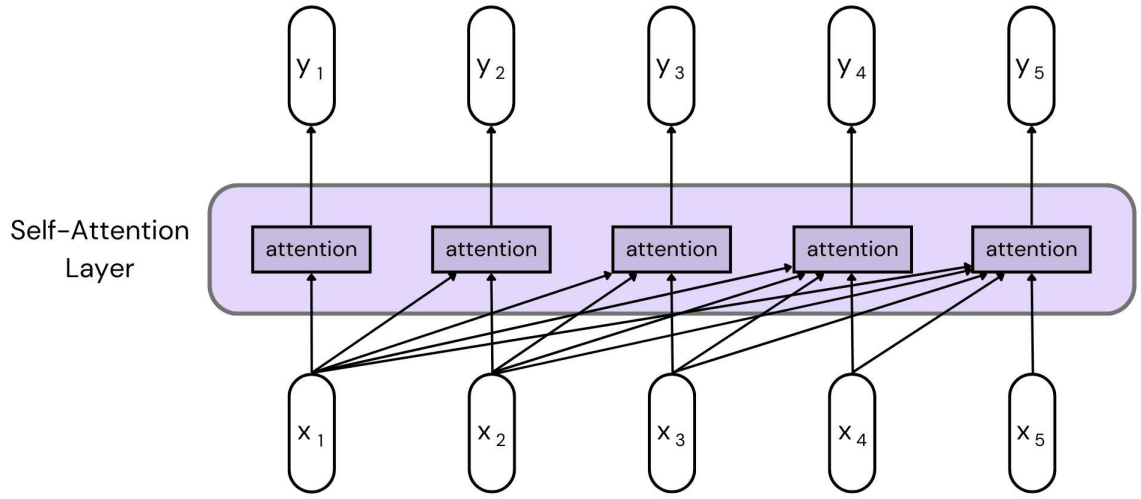
Isto tako, kod transformera, CLS token dodaje se na početak sekvence *patcheva* i koristi se za agregaciju informacija cijele slike. Ovaj token omogućuje modelu da stvori globalnu reprezentaciju slike integrirajući informacije iz svih *patcheva*, što je posebno korisno za zadatke poput klasifikacije.

CLS token, zajedno sa pozicijskim informacijama i outputom *Patch embedding* modula, čini ulaznu sekvencu koja se unosi u slojeve transformera.

4.2 Mehanizam pažnje

Mehanizam pažnje kao ulaz uzima listu vektora, a kao izlaz ponovno vraća listu vektora koji su dobiveni kao rezultat međusobnog utjecaja svakog vektora na ostale, što je jasno vidljivo na Slici 5. Drugim riječima, funkcija pažnje može se opisati kao proces mapiranja upita i skupa parova ključ-vrijednost na izlaz, pri čemu su upit, ključevi, vrijednosti i izlaz predstavljeni vektorima. Izlaz se računa kao ponderirani zbroj vrijednosti, gdje se težina dodijeljena svakoj vrijednosti određuje kompatibilnošću upita s odgovarajućim ključem [35].

Ovaj mehanizam pažnje omogućuje modelu da se fokusira na najrelevantnije dijelove ulazne sekvence, čime se ažurira reprezentacija svakog tokena. U kontekstu vizualnih transformera, to znači da model određuje relevantnost svakog *patcha* u odnosu na druge *patcheve* i na temelju toga ažurira njihovu reprezentaciju [4].



Slika 8: Grafički prikaz utjecaja *patcheva*

4.2.1 Matrice upit, ključ i vrijednost

Svaki *patch* slike predstavljen je vektorom dimenzije d_e koji se transformira u tri različita vektora: upit (*eng. query*) dimenzije d_q, ključ (*eng. key*) dimenzije d_k i vrijednost (*eng. value*) dimenzije d_v.

Svaki od ovih vektora ispunjava specifičnu ulogu unutar mehanizma pažnje:

- **Upit:** Vektor koji definira što trenutni *patch* "traži" u odnosu na druge *patcheve* i koje informacije su mu potrebne.
- **Ključ:** Vektor koji nosi informacije o značajkama *patcha* koje mogu biti relevantne za trenutni *patch*.
- **Vrijednost:** Vektor koji sadrži stvarne informacije svakog *patcha*, koje će se koristiti za stvaranje nove, ažurirane reprezentacije trenutnog *patcha* na temelju relevantnosti.

Generalno, koncepti upita, ključa i vrijednosti u transformerima mogu se usporediti s operacijama u bazi podataka. U bazi podataka, upit predstavlja pojam za pretragu, ključ predstavlja stupac ili polje koje se pretražuje, a vrijednost predstavlja sadržaj koji

se pretražuje. Sličnost između ova dva koncepta leži u tome da obje operacije uključuju pretragu specifičnih informacija na temelju određenih kriterija.

U praksi, funkcija pažnje se računa istovremeno na skupu upita, gdje se svi upiti grupiraju u matricu Q. Ključevi i vrijednosti također se grupiraju u matrice K i V, što omogućuje istovremenu obradu svih *patcheva* unutar slike [35].

Matricu Q, K i V dobivamo tako što ulazne vektore grupiramo u matricu X i množimo ju s tri različite naučene težinske matrice: W^Q za upit, W^K za ključ i W^V za vrijednost.

$$Q = XW^Q$$

$$K = XW^K$$

$$V = XW^V$$

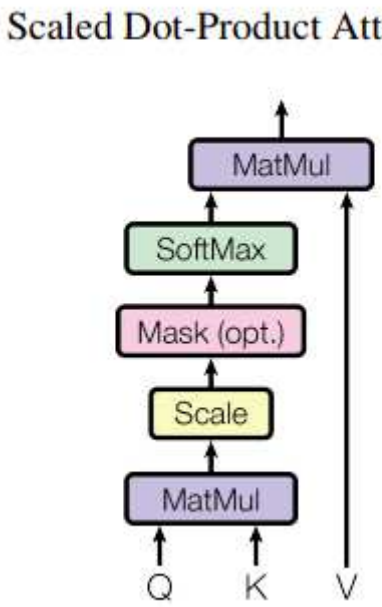
4.2.2 Skalirani unutarnji produkt

Relevantnost između trenutnog *patcha* i svih drugih *patcheva* izračunava se pomoću skalirane verzije unutarnjeg produkta (*eng. Scaled Dot-Product*) između matrica upita (Q) i ključeva (K) [35]:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

Unutarnji produkt između matrica upita (Q) i ključeva (K) normalizira se dijeljenjem s kvadratnim korijenom dimenzije ključeva $\sqrt{d_k}$, kako bi se spriječilo prekomjerno povećanje vrijednosti unutarnjeg produkta i stabilizirali gradijenti tijekom trening. Rezultirajuća matrica relevantnosti se zatim normalizira pomoću Softmax funkcije, koja generira vjerojatnosti koje prikazuju koliko je svaki *patch* relevantan za trenutni *patch* [35].

Konačna faza mehanizma pažnje koristi ove vjerojatnosti za ponderiranje matrice vrijednosti (V). Svaka vrijednost se skalira prema relevantnosti, a sve ponderirane vrijednosti se kombiniraju kako bi se dobila ažurirana reprezentacija trenutnog *patcha*. Ovaj proces omogućuje modelu da u svakom trenutku obrađuje sve relevantne informacije iz konteksta [14].



Slika 9: Skalirani unutarnji produkt [35]

4.2.3 Višestruka pažnja

Višestruka pažnja (*eng. Multi-Head Attention*) omogućuje modelu da uči različite aspekte relevantnosti i konteksta unutar podataka. Ključni korak u ovom procesu je podjela početnih matrica upita (Q), ključeva (K) i vrijednosti (V) na više manjih matrica, poznatih kao ”glave” (*eng. heads*. Svaka glava je odgovorna za učenje specifičnih značajki unutar podataka, omogućujući modelu da istovremeno obrađuje različite aspekte informacija.

Svaka glava prolazi kroz isti mehanizam pažnje, ali s manjim dimenzijama matrica, čime se omogućuje učinkovita obrada podataka. Nakon što se izračunaju vektori pažnje za svih h glava, oni se konkateniraju u jedinstvenu strukturu, te linearnom projekcijom realiziramo matricu W^O.

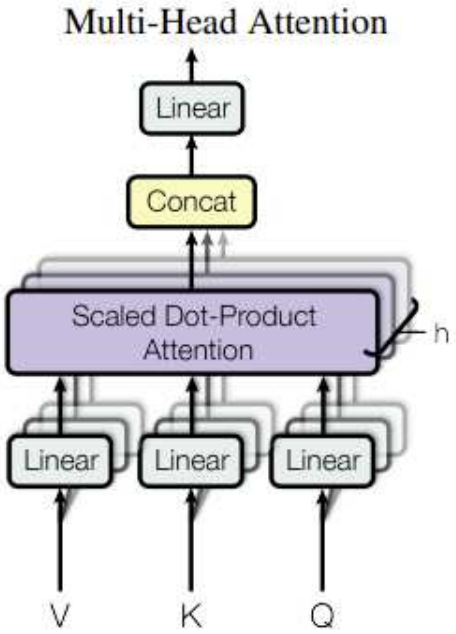
Matematički, višestruka pažnja se može opisati sljedećom formulom:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)W^O$$

gdje je svaka glava definirana kao:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$$

Nakon što su glave kombinirane, konačni rezultat se projicira natrag u izvornu dimenziju pomoću matrice W^O. Ovaj proces omogućuje modelu da istovremeno uči i koristi višestruke perspektive iz podataka, što značajno poboljšava njegovu sposobnost razumijevanja kompleksnih struktura i odnosa unutar podataka.



Slika 10: Višetruka pažnja [35]

5 Pregled transformer arhitektura

U ovom poglavlju dajemo prikaz različitih transformer arhitektura koje su se pokazale izrazito uspješnima kako u jezičnoj, tako i u vizuanoj domeni dubokog učenja. Fokusirat ćemo se na vizualne transformere ViT i DeiT, te popularne jezične modele GPT-2 i GPT-3.

5.1 ViT

Dosovitskiy et al. uspješno su iskoristili ideju transformera kako bi riješili problem klasifikacije slika te su 2020. godine predstavili Vision Transformer model (ViT) [8]. Ovaj model predstavlja prvi korak prema spajanju jezične i vizualne domene stojnog učenja, što se nekad činilo nemogućim.

ViT modeli korsite prethodno opisanu transformer arhitekturu te tako u svom fokusu imaju mehanizam pažnje. Trenirani su na velikim skupovima podataka poput skupa ImageNet i JFT-300M, ostvarujući rezultate koji su usporedivi ili čak nadmašuju performanse najboljih konvolucijskih neuronskih mreža. Budući da ViT nema ugrađeno ”predznanje” kao što to imaju knovolucijske neuronske mreže, jedan od ključnih izazova je upravo njegova potreba za velikim količinama podataka tijekom treninga. Dok konvolucijske neuronske mreže koriste unaprijed definirane filtre za prepoznavanje rubova, oblika i drugih vizualnih elemenata, ViT uči te značajke direktno iz podataka.

ViT modeli dolaze u nekoliko varijacija koje se razlikuju prema veličini modela i složenosti arhitekture, a tri glavne varijante zajedno sa njhihovim ključnim parametrima prikazane su u Tablici 1.

Tablica 1: Detalji varijacija ViT modela [8]

Model	Layers	Hidden size D	MLP size	Heads	Params
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

ViT je postavio temelje za daljnji razvoj modela temeljenih na transformer arhitekturi u domeni računalnog vida, pružajući inspiraciju za modele kao što su Swin Transformer i DeiT. Ovi modeli također koriste mehanizam pažnje za obradu vizualnih podataka, ali raznim optimizacijama smanjuju računalne zahtjeve i poboljšavaju učinkovitost modela na manjim skupovima podataka.

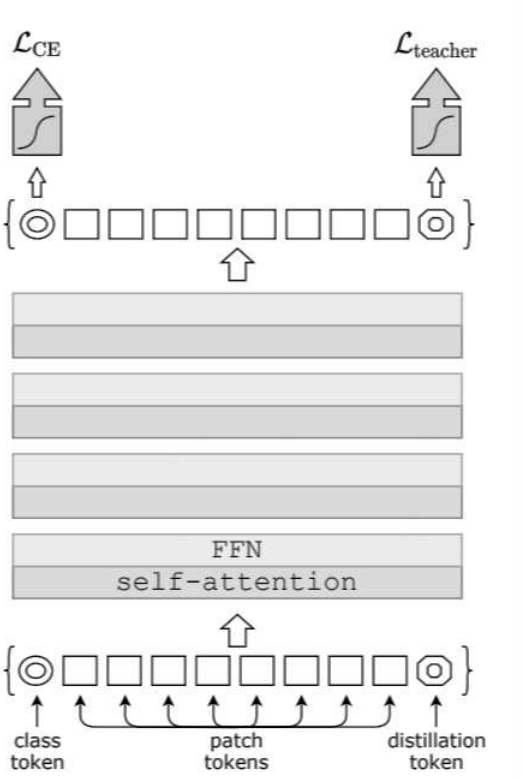
5.2 DeiT

Objavom DeiT modela 2021. godine, Touvron et al. dokazali su da se transformer arhitekture mogu uspješno trenirati na srednje velikim skupovima podataka poput ImageNet-a u relativno kraćim periodima u usporedbi s ViT modelom [32].

DeiT obradi slika pristupa vrlo slično kao i ViT, dijeleći sliku na 16×16 *patcheva* koji se vektoriziraju te prosljeđuju transformer blokovima. Ključna inovacija ovog modela je u destilaciji znanja (*eng. knowledge destillation*) koja označava trening paradigmu koja uključuje istovremeno treniranje dviju neuronskih mreža: jake, prethodno trenirane ”učiteljske” mreže te slabije, nasumično inicijalizirane ”učeničke” mreže. U ovom slučaju, kao učiteljska mreža koristi se konvolucijska mreža RegNetY-16GF.

Destilacijska procedura uključuje dodavanje posebnog destilacijskog tokena (*eng. destillation token*) u modulu za promjenu reprezentacije slike. Destilacijski token se inicijalizira nasumično, može se učiti te se nalazi na kraju ulazne sekvence (suprotno od CLS tokena koji je na početku sekvence). Cilj ovog tokena je naučiti korisne informacije iz predikcija učiteljske mreže. Na izlazu mreže koristi se kombinacija križne entropije i posebne destilacijske funkcije gubitka te na tako mreža istovremeno uči iz klasičnih tokena i destilacijskih tokena koji dolaze od učiteljske mreže.

Dodatna strategija koju koristi DeiT uključuje *fine-tuning* modela na slikama veće rezolucije. Tijekom treninga, DeiT koristi slike veličine 224×224 piksela, dok se tijekom *fine-tuninga* koriste slike veličine 384×384 piksela. Ovaj pristup učinkovito povećava količinu trening podataka jer veća rezolucija slika pruža detaljnije informacije koje model do sada nije vidio. Kako bi se slike veće rezolucije mogle primijeniti, koristi se posebna tehnika regularizacije.



Slika 11: Primjer destilacjiske procedure DeiT modela [32]

DeiT dolazi u više varijacija, od kojih je najpoznatija DeiT-B koja ima 86 milijuna parametara i postiže bolju točnost nego njezina učiteljska mreža.

Integriranjem ovih strategija, DeiT uspijeva iskoristiti snagu transformatora za zadatke klasifikacije slika čak i kada je količina trening podataka ograničena, čime poboljšava učinkovitost i performanse modela.

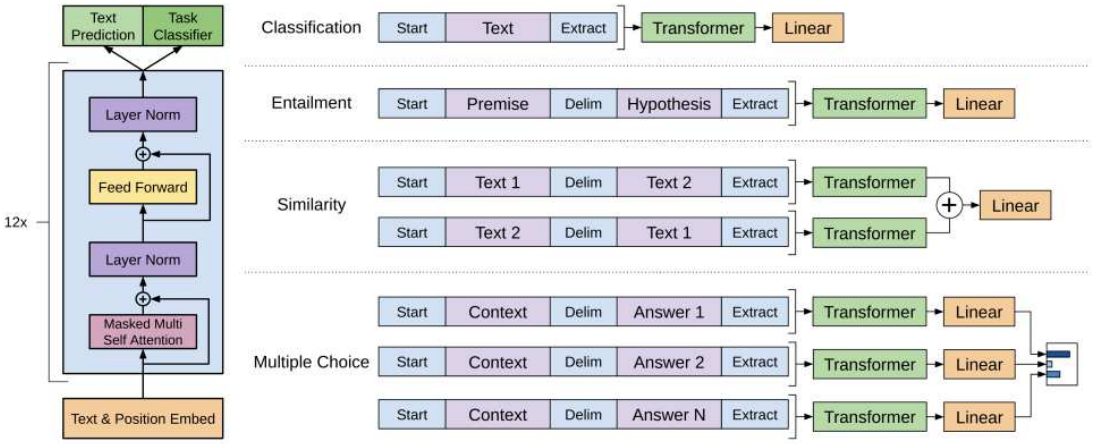
5.3 GPT-2

GPT-2 predstavljen je 2019. godine kao druga iteracija GPT (*eng. Generative Pre-Trained Transformer*) serije modela [24] . GPT arhitektura se kao i ViT temelji na arhitekturi transformera predstavljenoj u radu ”Attention Is All You Need”. Međutim, GPT arhitektura koristi samo dekoder komponentu jer je prvenstveno dizajnirana za generiranje teksta iz sekvencijalnog ulaza.

Iz samog naziva arhitekture mogu se izdvojiti njezine tri ključne stavke: generativnost, prethodno treniranje (*eng. pre-training*) i transformer arhitektura. Prethodno treniranje na velikim količinama podataka omogućuje mreži razumijevanje općih zna-

čajki i strukture jezika, što rezultira generiranjem teksta koji djeluje potpuno prirodno. Dodatno, transformer arhitektura omogućava modelu bolje rukovanje dugoročnim ovisnostima u tekstu zahvaljujući svojoj strukturiranoj memoriji.

Proces treniranja GPT modela započinje ne nadziranim treniranjem (*eng. Unsupervised pre-training*) na velikom skupu neoznačenih podataka čime se uče početni parametri mreže. Nakon toga slijedi *fine-tuning* tijekom kojeg se dodatnim treniranjem na podatcima specifičnim uz određenu domenu ili zadatak poboljšava se preciznost modela. Važno je naglasiti da, za različite zadatke, GPT ne zahtijeva promjene u arhitekturi, već samo u formatu ulazne sekvence. Transformacije koje se primjenjuju na ulaznu sekvencu vidljive su na Slici 12



Slika 12: Transformacije ulazne sekvence ovisno o tipu zadatka [23]

U usporedbi s prvom GPT-1 verzijom, GPT-2 treniran je na WebText skupu podataka koji sadrži različite izvore s interneta, uključujući članke, forume i blogove, izbjegavajući stranice s pretežno automatski generiranim ili korisnički generiranim sadržajem. GPT-2 dolazi u nekoliko verzija, a najveća ima približno 1.5 bilijuna parametara što joj omogućuje bolje generiranje dužih i koherentnijih tekstova.

GPT-2 postiže dobre performanse kod zadataka sažimanja teksta, prijevoda, odgovaranja na pitanja i generiranja kreativnog sadržaja. Često generira tekst gramatički ispravan, ali on može sadržavati nelogične ili netočne tvrdnje.

5.4 GPT-3

GPT-3 predstavljen je 2020. godine [2] i temelji se na istoj arhitekturi kao GPT-2, ali ima značajno veći broj parametara i složeniju strukturu. Ovaj model je treniran na skupu podataka Common Crawl i ima čak 175 bilijuna parametara, što je velika razlika u odnosu na GPT-1 i GPT2. Ovu razliku možemo vidjeti u Tablici 2 koja sadrži detalje GPT modela.

Tablica 2: Usporedba GPT serija modela

	GPT-1	GPT-2	GPT-3
Parameters	117M	1.5B	175B
Decoder Layers	12	48	96
Context Token Size	512	1024	2048
Hidden Layer	768	1600	12288
Batch Size	64	512	3.2M

Uz povećanje broja parametara, GPT-3 donosi još neke ključne inovacije. Jedna od njih je izmjena guste i lokalno ograničene rijetke pažnje. Drugim riječima, neki slojevi transformera koriste tradicionalnu gustu pažnju kod koje svaki token može obraćati pažnju na sve druge tokene u sekvenci, dok drugi slojevi koriste lokalno ograničenu rijetku pažnju gdje svaki token obraća pažnju samo na podskup tokena unutar svog neposrednog konteksta. Ovaj pristup inspiriran Sparse Transformer modelom, smanjuje računalnu složenost modela uz zadržavanje dobrih performansi.

Također, GPT-3 postiže iznimno dobre performanse bez prethodnog treniranja na specifičnim zadatcima (*eng. zero-shot*), ili uz minimalan broj primjera (*eng. few-shot*). GPT-3 pokazuje izuzetne sposobnosti u kontekstualnom učenju zahvaljujući svom autoregresivnom pre-treniranju, što mu omogućuje izvrsno obavljanje zadataka bez dodatnog prilagođavanja na specifične zadatke.

GPT-3 postiže dobre rezultate kod mnogih zadataka, uključujući prijevod, odgovaranje na pitanja, rasklapanje riječi, generiranje programskog koda ili izvođenje 3-znamenkaste aritmetike. Međutim, unatoč velikom uspjehu, GPT-3 ima i neke nedostatke poput semantičkog ponavljanja, proturječenja, gubitka koherencije u dugim odlomcima i sl.

6 Usporedba transformera i konvolucijskih neuronskih mreža

Vizualni transformeri i konvolucijske neuronske mreže predstavljaju dva različita pristupa u dubokom učenju, posebno u području računalnog vida. Iako obje arhitekture imaju za cilj učinkovito procesiranje vizualnih podataka, njihovi unutarnji mehanizmi i načini na koje predstavljaju informacije značajno se razlikuju. U ovom poglavlju analizirat ćemo ključne razlike između ove dvije arhitekture, fokusirajući se na njihove pristupe ekstrakciji značajki, induktivnoj pristranosti (*eng. inductive bias*), očuvanju prostorne informacije te . Ova analiza će nam pomoći da razumijemo kako različite arhitekture utječu na performanse modela i zašto su određeni modeli pogodniji za specifične vrste zadataka.

6.1 Ekstrakcija značajki

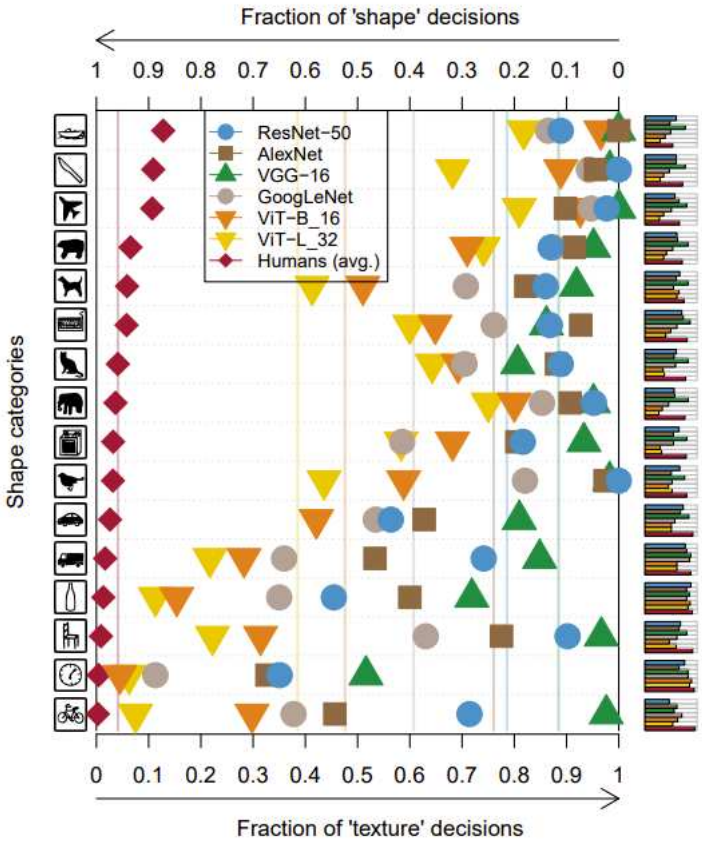
Konvolucijske neuronske mreže koriste konvolucijske slojeve koji primjenjuju filtere za prepoznavanje lokalnih značajki poput rubova i tekstura. Svaki filter specijaliziran je za prepoznavanje različitih značajki, a hijerarhijsko slaganje konvolucijskih slojeva omogućuje postupnu izgradnju složenijih reprezentacija. Ovakva arhitektura se pokazala iznimno uspješna u prepoznavanju objekata na slikama gdje su prisutni složeni lokalni uzorci, kao što su prepoznavanje lica ili tekstura.

S druge strane, transformeri slike obrađuju kao sekvence *patcheva* i koriste mehanizme pažnje za prepoznavanje kako globalnih tako i lokalnih ovisnosti u slici. Ovakav pristup transformerima omogućuje da od početka prepoznaju lokalne i globalne informacije, umjesto da ih postupno grade iz lokalnih informacija kao što to rade konvolucijske neuronske mreže.

6.2 Induktivna pristranost

Induktivna pristranost odnosi se na pretpostavke koje određeni model ima u procesu učenja, omogućujući mu da generalizira izvan skupa podataka na kojem je treniran. U modelima dubokog učenja induktivna pristranost značajno utječe na učinkovitost u različitim zadacima prepoznavanja slika i može biti osobito važna kada se uspoređuju konvolucijske neuronske mreže i vizualni transformeri.

Kako bi ilustrirali razliku u induktivnoj pristranosti između konvolucijskih neuronskih mreža i transformera, Tuli et al. analizirali su performanse ovih modela na skupu podataka SIN (*eng. Stylized ImageNet*) [33]. Pri kreiranju vizualizacije prikupili su sve izlaze klasifikatora koji odgovaraju teksturi ili obliku predmeta te zatim odredili udio koliko je predmeta ispravno klasificirano prema teksturi, a koliko prema obliku.

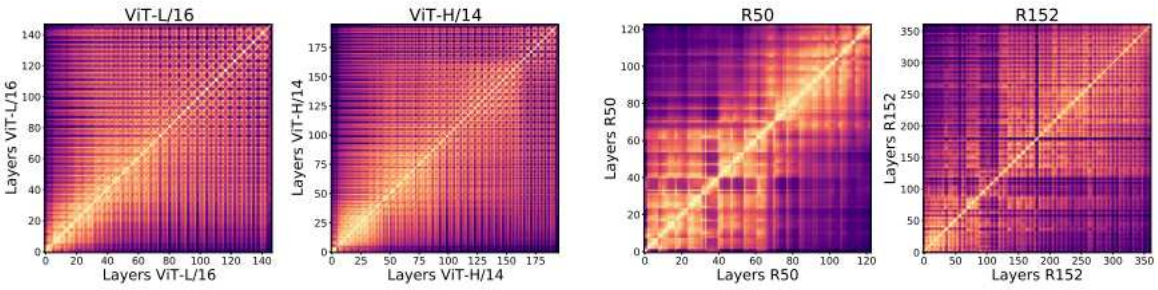


Slika 13: Pristranost oblika za različite modele na skupu podataka SIN [33]

Rezultati su prikazani na Slici 13, gdje je vidljivo da su vizualni transformeri pokazali veću sklonost prepoznavanju oblika u usporedbi s konvolucijskim neuronskim mrežama, koje su imale veću sklonost prepoznavanju teksture. Ove razlike mogu se objasniti induktivnim pristranostima svakog modela. Konvolucijske neuronske mreže, zbog svoje arhitekture koja se oslanja na lokalne filtere, imaju veću prirodnu sklonost prepoznavanju teksturalnih značajki prisutnih na manjem prostoru. Nasuprot tome, vizualni transformeri, koji koriste mehanizme pažnje za globalno učenje odnosa između značajki, mogu bolje prepoznati oblike jer su u stanju integrirati informacije iz cijele slike, čime smanjuju utjecaj lokalnih varijacija i bolje razumiju globalne obrasce. Ova sposobnost čini ih pogodnijima za zadatke koji zahtijevaju prepoznavanje složenih oblika u usporedbi s konvolucijskim neuronskim mrežama.

6.3 Reprezentacija

Kako bi usporedili i analizirali reprezentacije značajki između vizualnih transformera i konvolucijskih neuronskih mreža, Raghu et al. [25], uz pomoć različitih tehnika došli su do vrlo zanimljivih vizualizacija. Jedna od tih vizualizacija prikazuje kako se reprezentacije značajki u različitim slojevima transformera i konvolucijskih mreža međusobno odnose, a možemo ju vidjeti na Slici 14.



Slika 14: Reprezentacija značajki u različitim slojevima ViT i ResNet modela [25]

Na prvi pogled možemo primijetiti očite razlike između ViT i ResNet modela, gdje ViT modeli pokazuju impresivnu uniformnost u strukturi kroz slojeve. Ova uniformnost znači da su značajke prepoznate u različitim slojevima sličnije nego kod ResNet modela, gdje postoji jasnija razlika između značajki prepoznatih u plićim i dubljim slojevima, što je i očekivano, s obzirom na to da svaki sloj u konvolucijskim neuronskim

mrežama prepoznaje sve apstraktnije koncepte.

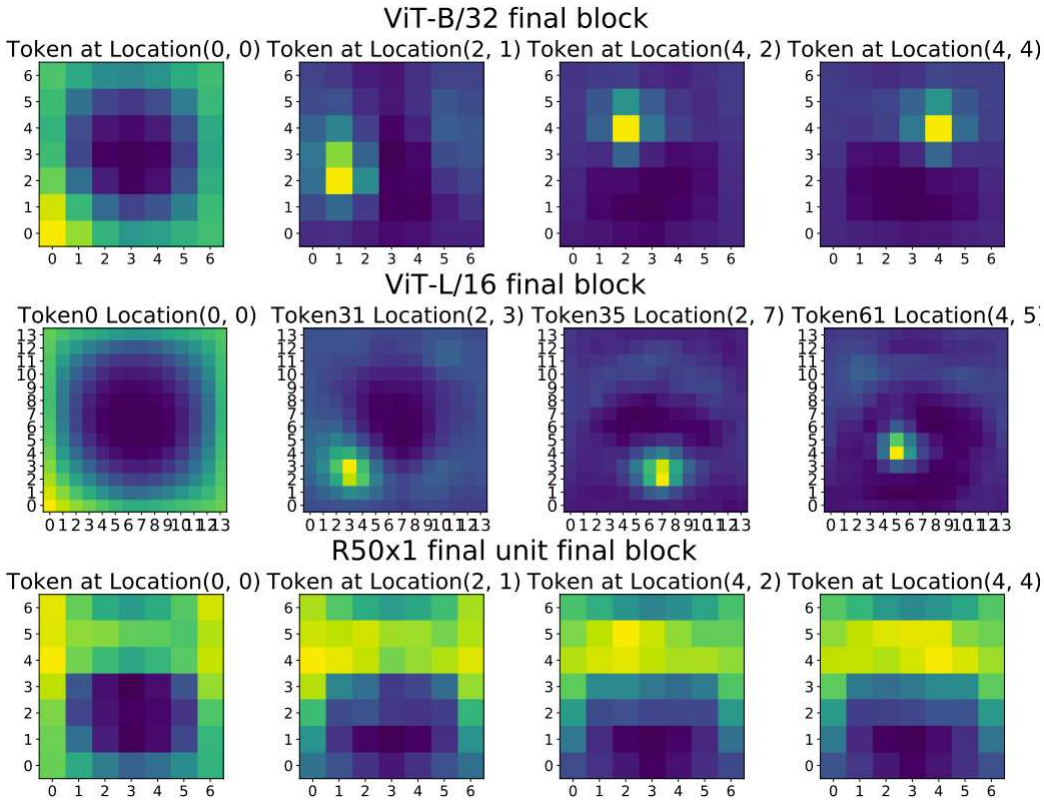
Budući da su reprezentacije ViT modela poprilično slične i uniformne, fokusirat ćemo se na donji dio reprezentacije ViT-L/16 modela. Ovaj dio je prilično uniforman, posebno u usporedbi s ResNet modelima, što znači da reprezentacije ostaju slične dok se propagiraju kroz ViT. Nasuprot tome, kod ResNet modela dolazi do kvalitativne promjene značajki kako se podaci kreću kroz mrežu. Konkretno, sličnost između značajki dubljih slojeva, od 80. do 120. sloja, i onih u plitkim slojevima znatno opada, što ukazuje na promjene u značajkama kroz mrežu.

Ako pogledamo gornji desni dio dijagrama modela R50, gdje se uspoređuje sličnost između značajki dubljih slojeva, primijetit ćemo visoku izlaznu sličnost, kao i u donjem lijevom kutu. Nasuprot tome, ViT modeli pokazuju mnogo ujednačeniju strukturu sličnosti. To implicira da ViT modeli zadržavaju sličnost značajki kroz sve slojeve, dok ResNet pokazuje značajnije promjene u značajkama kroz svoju arhitekturu.

6.4 Očuvanje prostorne informacije

Očuvanje prostorne informacije ključno je za mnoge zadatke računalnog vida, a pogotovo za detekciju objekata na slici. Kako bi analizirali očuvanje prostorne informacije, Raghu et al. [25] napravili su još jednu zanimljivu vizualizaciju koja ilustrira sličnost između reprezentacija tokena u višim slojevima ViT i ResNet modela te ulaznih dijelova slike. U ViT modelima, svaki token je povezan s određenim dijelom ulazne slike, dok se u ResNet modelima reprezentacija tokena odnosi na sve konvolucijske kanale na određenoj prostornoj lokaciji. Za računanje sličnosti korištena je metoda centriranog poravnanja jezgre (*eng. Centred Kernel Alignment, CKA*).

Na Slici 15 možemo vidjeti da tokeni u ViT modelima zadržavaju prostorne informacije vrlo precizno. Međutim, ako uzmemo token na poziciji (0,0) vidimo da on iz nekog razloga pokazuje sličnost i s drugim rubnim tokenima. Nasuprot tome, kod ResNet modela može se primijetiti slabije očuvanje prostorne informacije, pokazujući sličnost preko širokog raspona ulaznih prostornih lokacija. Jedan od faktora koji doprinosi ovoj razlici između arhitektura je način na koji su trenirani: ResNet koristi globalno prosječno sažimanje (*eng. Global Average Pooling, GAP*) za klasifikaciju, dok ViT koristi zase-



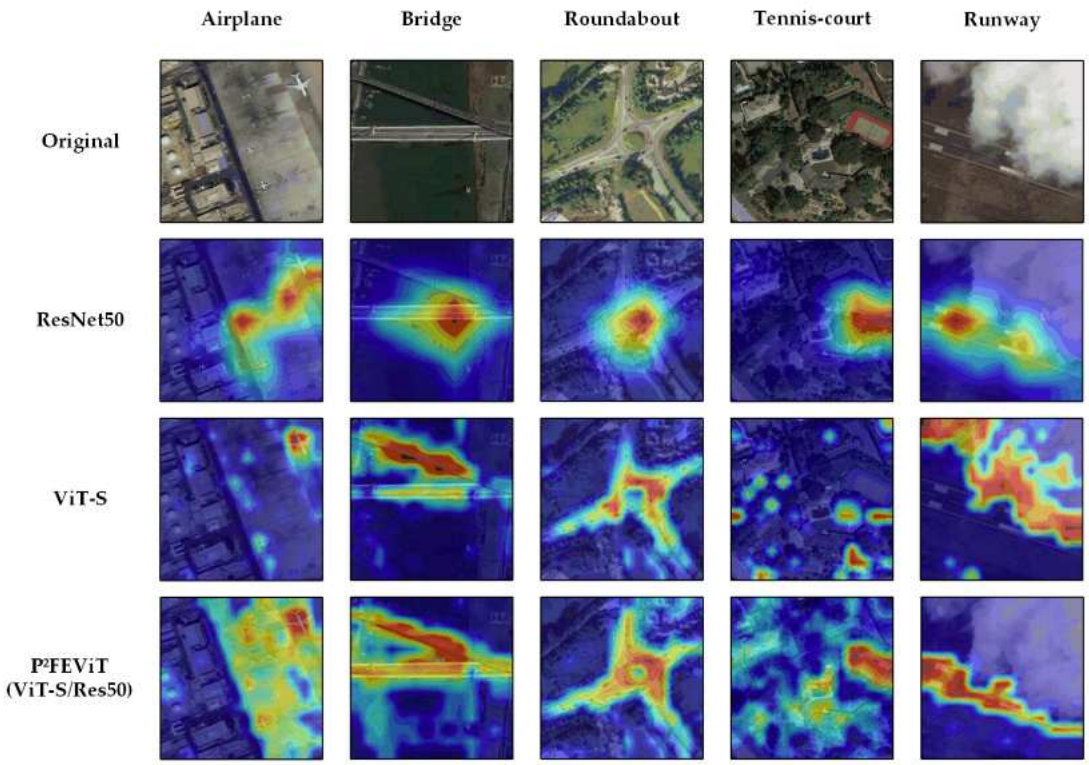
Slika 15: Vizualizacija sličnosti između reprezentacija tokena u višim slojevima ViT i ResNet modela te ulaznih dijelova slike [25]

ban CLS token. Globalno prosječno sažimanje uzima prosjek svake mape značajki i prosljeđuje rezultirajući vektor direktno softmax funkciji, umjesto da koristi potpuno povezane slojeve iznad mapa značajki. Raghu et al. također su pokazali kako bi ViT modeli izgubili prostornu svjesnost kada bi bili trenirani s globalnim prosječnim sažimanjem.

6.5 Fokus odlučivanja

Prošli smo kroz ključne razlike arhitekture vizualnih transformera i konvolucijskih neuronskih mreža, no kako se one reflektiraju na samo donošenje odluke analizirat ćemo kroz sljedeći primjer. Wang et al. predložili su hibridni model temeljen na ove dvije arhitekture te kako bi usporedili modele vizualizirali su mape značajki uz pomoć Grad-CAM metode (*eng. Gradient-weighted Class Activation Map*) [36]. Grad-CAM je tehnika koja jednostavno pruža vizualno objašnjenje za donošenje odluka specifičnog modela.

Ako pogledamo Sliku 16, možemo uočiti da se ResNet50 fokusira na lokalne značajke te manje uzima u obzir globalni kontekst slike. Ovo je posljedica mehanizma konvolucijskih neuronskih mreža, kojim se globalne informacije grade postupno iz lokalnih. S druge strane, klasičan ViT model u fokusu ima širi, tj. globalni kontekst slike zahvaljujući svom mehanizmu pažnje kod kojeg svaki token utječe na sve druge tokene u sekvenci.



Slika 16: Vizualizacija mapa značajki Grad-CAM metodom [36]

Kako bi se integrirale prednosti oba modela, istraživači su počeli raditi na tzv. hibridnim modelima koji bi uključili prednosti konvolucijskih neuronskih mreža u transformer arhitekturu te tako omogućili istovremeno opažanje lokalnih i globalnih značajki. Potaknuti ovom idejom, Wang et al. predložili su P^2FEViT model. Ako ponovno pogledamo Sliku 16 i fokusiramo se na hibridni model, možemo vidjeti kako on ima sposobnost fokusiranja na globalne i lokalne značajke istovremeno.

7 Usporedba performansi novih modela za detekciju objekata na prometnim podatcima[2]

Kako bismo identificirali razlike između različitih modela i arhitektura za detekciju objekata, odlučili smo usporediti njihove performanse na prometnim podatcima. Ovo poglavlje evaluira performanse 14 novih modela za detekciju objekata s ciljem pružana preporuke istraživačima koji ulaze u ovo područje. Performanse su procijenjene na temelju veličine objekata, vremena inferencije i broja parametara. Rezultati pokazuju da, iako nekoliko modela postiže slične rezultate na cijelom skupu podataka, njihova učinkovitost može značajno varirati za male i velike objekte. Također, razlike u vremenu inferencije su značajne i mogu utjecati na izbor prikladnog modela. Zaključak istraživanja naglašava najbolje modele na temelju njihovih specifičnih prednosti.

7.1 Uvod

Prepoznavanje i pronalaženje objekata od interesa u nekoliko sekundi smatra se trivijalnim zadatkom za ljude, ali u području računalnog vida (*eng. computer vision*) ta je aktivnost puno složenija. Računalni vid trenira strojeve da vide, promatraju i steknu visoku razinu razumijevanja okoline i objekata u njoj. Arhitekture dubokog učenja uče temeljne obrasce u podacima i automatski izdvajaju najrelevantnije i najzanimljivije značajke iz podataka, čime se poboljšava točnost u određenim primjenama računalnog vida poput detekcije objekata [9].

Jedan od temeljnih problema računalnog vida je detekcija objekata (*eng. object detection*). Cilj detekcije objekata je otkriti sve instance unaprijed definiranih klasa i odrediti njihov približni položaj na slici uz pomoć okvira poravnatih s osima [41]. Složenost ovog zadatka povećavaju čimbenici poput različitih vremenskih uvjeta, kuta gledanja, osvjetljenja itd. Međutim, unatoč tome, detekcija objekata značajno je napredovala u posljednjem desetljeću. Brzi razvoj tehnika za detekciju objekata, brže grafičke procesorske jedinice (*eng. GPUs*) i dostupnost velikih količina podataka doveli su do toga

[2]Ovo poglavlje je prezentirano kao znanstveni rad na konferenciji 2023 8th International Conference on Machine Learning Technologies [34]

da sada možemo lako trenirati računala da prepoznaju i klasificiraju više objekata na slici ili videu s visokim stupnjem točnosti.

Važnost detekcije objekata leži u činjenici da je vizualna percepcija jedno od najvažnijih ljudskih osjetila. Uvelike se oslanjamo na vid kad god smo u interakciji s okolinom, ali ponekad nas u tome ometaju brojni čimbenici. Sve veća potreba za visokom točnošću potaknula je ljude da istražuju i primjenjuju detekciju objekata za rješavanje problema iz stvarnog svijeta.

Primjene detekcije objekata su neograničene. Kao jedan od temeljnih zadataka računalnog vida, detekcija objekata može pružiti vrijedne informacije za semantičko razumijevanje slika i videa i povezana je s mnogim primjenama, uključujući klasifikaciju slika, analizu ljudskog ponašanja, prepoznavanje lica i autonomnu vožnju [44]. Naširoko se koristi u mnogim područjima, kao što su sigurnost, medicina, vojska i transport.

Ovo istraživanje usmjereno je na primjenu detekcije objekata u prometu. Prema podacima Svjetske zdravstvene organizacije, otprilike 1,3 milijuna života svake se godine gasi zbog nesreća u cestovnom prometu, dok između 20 i 50 milijuna ljudi pretrpi ozljede koje nisu smrtonosne. Sustavi autonomne vožnje trenutno predstavljaju glavni izazov u znanstvenoj zajednici i industriji; jedan od glavnih ciljeva takvih sustava je smanjiti postotak nesreća uzrokovanih ljudskim faktorom. U posljednjih nekoliko godina, procvat istraživanja pokazao je da su mnoge tehnologije računalnog vida učinkovite za sigurnu i pouzdanu samostalnu vožnju [1]. Sustavi autonomne vožnje uključuju mnoge zadatke, poput detekcije prometnih traka, detekcije objekata, semantičke segmentacije, simultane lokalizacije, mapiranja i sl.

Zbog velikog broja dostupnih modela za detekciju objekata i nedostatka sustavnih usporedbi, odabir odgovarajućeg modela može biti vrlo zahtjevan. Osnovni cilj ovog istraživačkog rada je popuniti tu prazninu pružajući usporedbu performansi novih modela. Doprinos istraživanja ogleda se kroz definirane preporuke za odabir modela prikladnih za detekciju objekata.

7.2 Pregled literature

Istraživači su predložili mnoge okvire za zadatak detekcije objekata, koji se uglavnom mogu klasificirati u dvije vrste: jednostupanjski i dvostupanjski. Jednostupanjski modeli zahtijevaju samo jedan prolaz za klasifikaciju i lokalizaciju objekata, što ih čini puno bržima i prikladnima za detekciju objekata u stvarnom vremenu. Kod dvostupanjskih detektora, u prvom koraku generira se rijedak skup prijedloga, dok se u drugom koraku značajke generiranih prijedloga kodiraju pomoću dubokih konvolucijskih neuronskih mreža, nakon čega slijedi predikcija klase objekta [39]. Veliki uspjeh transformatora u obradi prirodnog jezika (NLP) motivirao je mnoge istraživače da istraže njihovu primjenu u računalnom vidu, što je otvorilo nove smjerove istraživanja. U nastavku će se analizirati literatura prema spomenutoj klasifikaciji.

Jednostupanjski model koji je revolucionirao detekciju objekata je YOLO [26]. Ovaj algoritam detekciji objekata pristupa kao problemu regresije i zahtijeva samo jedan prolaz kroz neuronsku mrežu za predviđanje višestrukih okvira i vjerojatnosti klasa za te okvire. YOLO je puno brži algoritam od konkurentnih, ali ima jedan nedostatak: teško raspoznaje male objekte koji se pojavljuju u skupinama.

Još jedan model koji se dobro pokazao u pogledu točnosti i vremenske učinkovitosti je RetinaNet [18]. Ovaj jednostupanjski model postigao je izvrsne rezultate zahvaljujući unaprijeđenoj verziji klasične funkcije gubitka pod nazivom *Focal loss*. *Focal loss* funkcija smanjuje nesrazmjernost između pozitivnih i negativnih uzoraka, čime poboljšava preciznost i performanse modela na malim gusto raspoređenim objektima.

Unatoč jako dobrim performansama koje je postigao model RetinaNet, model FCOS, koji je bez sidrenja (*eng. anchor-free*), uspio je ostvariti značajna poboljšanja. Koristeći istu osnovnu mrežu ResNet-101-FPN, FCOS je povećao AP za 2.4% u odnosu na RetinaNet model [31]. FCOS eliminira unaprijed definirani skup sidrenih okvira i izbjegava sve povezane hiperparametre, rješavajući detekciju objekata predviđanjem na razini piksela.

Kasnije su uvedene deformabilne konvolucijske mreže (*eng. Deformable ConvNets*) [6] koje se prilagođavaju različitim geometrijskim varijacijama uzrokovanim skalom, položajem i kutom gledanja. Njihova prostorna podrška proteže se daleko izvan područja interesa, što može uzrokovati da značajke pod utjecajem irelevantnog sadržaja slike. Druga inačica deformabilnih konvolucijskih mreža (*eng. Deformable ConvNets v2*) [45] poboljšavaja sposobnost fokusiranja na relevantna područja slike, a njezini deformabilni moduli mogu se integrirati u postojeće arhitekture mreža poput Faster R-CNN i Mask R-CNN.

Jedan od modela temeljenih na dvostupanjskoj paradigmi je Faster R-CNN [27]. Faster R-CNN koristi duboku konvolucijsku neuronsku mrežu za generiranje prijedloga regija, pri čemu je izračun prijedloga gotovo besplatan u odnosu na računanje koje zahtijeva mreža za detekciju. Sastoji se od dva modula. Prvi je duboka potpuno konvolucijska mreža koja se koristi kao mreža za prijedloge regija (*eng. Region Proposal Network, RPN*), dok je drugi u biti Fast R-CNN [10], koji zatim koristi ove prijedloge.

Poznato je da performanse detekcije imaju tendenciju pogoršanja s povećanjem IoU (*eng. Intersection over Union*) pragova. Ovaj problem riješen je novom arhitekturom detektora koju su predložili Cai i Vasconcelos, nazvanom Cascade R-CNN [3]. Arhitektura se sastoji od niza detektora koji su trenirani s postepeno višim IoU pragovima, kako bi postupno bili selektivniji prema lažno pozitivnim rezultatima koji su blizu stvarnih, te kako bi se mogla izgraditi koristeći bilo koji dvostupanjski detektor objekata temeljen na R-CNN frameworku.

Pang et al. [22] pokazuju da performanse detekcije trpe zbog neravnoteže tijekom procesa treniranja, koja se javlja na tri razine: razini uzorka, razini značajki i razini cilja. Kako bi uravnotežili proces treniranja, predlažu Libra R-CNN okvir, koji integrira tri nove komponente: IoU-uravnoteženo uzorkovanje, uravnoteženu piramidu značajki i uravnoteženi L1 gubitak.

Sparse R-CNN [30] predstavlja novu vrstu detektora koja se ne oslanja na guste kandidate za objekte, već na paradigmu rijetko-ulazno rijetko-izlazno (*eng. sparse-in sparse-out*). Njegove su performanse usporedive sa glavnim detektorima kao što su FCOS, Faster R-CNN i RetinaNet, dok u scenama s puno objekata postiže čak još bolje rezultate.

HRNet [13] je snažnija osnovna mreža za probleme računalnog vida koja postiže još bolje rezultate od ResNet-a i ResNeXt-a. Ova arhitektura koristi dvostupanjsku *top-down* paradigmu: prvo detektira instancu osobe koristeći detektor osoba, a zatim predviđa ključne točke detekcije. Tijekom cijelog procesa, HRNet povezuje konvolucije serijski, od visoke do niske rezolucije, čime održava prikaze visoke rezolucije. Nedostatak je taj što je memorijski trošak treniranja u detekciji objekata nešto viši u odnosu na state-of-the-art modele.

Swin Transformer [19] predstavlja drugu vrstu osnovne mreže, čija je primjena u zadacima računalnog vida tek na početku. Ovaj transformer konstruira hijerarhijske mape značajki i ima linearnu računalnu složenost u odnosu na veličinu slike. Iako postiže *state-of-the-art* performanse u detekciji objekata na COCO datasetu, i dalje koristi više parametara nego konvolucijski modeli.

Druga vrsta osnovne mreže temeljene na transformerima je PVT [38], ili Pyramid Vision Transformer, nekonvolucijska mreža korisna za mnoge zadatke guste predikcije. PVT postiže visoku izlaznu rezoluciju, čime se prevladavaju poteškoće primjene transformera na različitim zadacima guste predikcije. Ograničenja i računalna složenost PVT v1 poboljšani su verzijom Pyramid Vision Transformer v2 [37], koja može bolje očuvati lokalni kontinuitet slika i mapa značajki, fleksibilnije obrađivati ulaz različite rezolucije i, što je najvažnije, uživati istu linearnu složenost kao i CNN.

Kako bi smanjili razliku u performansama između detektora temeljenih na sidrištima (*eng. anchor-based*) i onih bez sidrišta (*eng. anchor-free*), koja proizlazi iz načina na koji definiraju pozitivne i negativne uzorke tijekom treniranja, Zhang et al. predložili su Adaptive Training Sample Selection (ATSS) [43]. Ova metoda automatski odabire pozitivne i negativne uzorke na temelju statističkih karakteristika objekta i značajno poboljšava performanse vrhunskih detektora, dosižući 50.7% AP bez dodavanja dodatnog opterećenja sustavu.

Uz navedene radove, postoji nekoliko drugih radova koji pružaju pregled metoda detekcije objekata i njihove primjene [12, 44, 41]. S obzirom na brz razvoj dubokog učenja, neki od tih radova su već zastarjeli i ne uključuju mnoge nove modele detekcije objekata, kao što su PVT modeli.

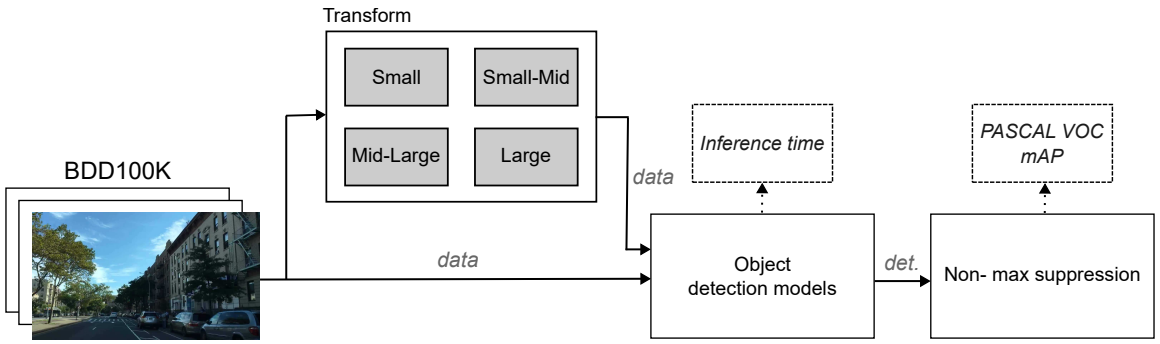
7.3 Metodologija istraživanja

Ovo poglavlje pruža detaljan opis našeg eksperimenta, koji ima za cilj usporediti performanse modela za detekciju objekata na skupu prometnih podataka. Kako bi se postigao cilj ovog istraživanja i obogatila analiza, prvo su dane dodatne informacije o odabranim modelima za detekciju objekata.

Tablica 3: Popis novih modela za detekciju objekata s njihovim osnovnim karakteristikama

Called	Method	Backbone	Model type	Method parameters (M)	Backbone parameters (M)	Total parameters (M)
OS1	ATSS [43]	ResNet101-FPN	one-stage	9	43	52
OS2	ATSS [43]	ResNet101-FPN + Dyhead	one-stage	15	43	58
TS3	Cascade R-CNN [3]	ResNet101-FPN	two-stage	46	43	89
TS4	Cascade R-CNN [3]	Swin-T	two-stage	45	27	72
TS5	Faster R-CNN [27]	HRNet32	two-stage	18	29	47
TS6	Faster R-CNN [27]	ResNet50-FPN	two-stage	18	24	42
TS7	Faster R-CNN [27]	ResNet101-FPN	two-stage	18	43	61
OS3	Deformable ConvNets [45]	ResNet101-FPN	one-stage	18	44	62
OS4	FCOS [31]	ResNet101-FPN	one-stage	9	43	52
TS8	Libra R-CNN [22]	ResNet101-FPN	two-stage	18	43	61
OS5	RetinaNet [18]	PVT-S	one-stage	9	24	33
OS6	RetinaNet [18]	PVTv2-B1	one-stage	9	25	34
OS7	RetinaNet [18]	ResNet101-FPN	one-stage	13	43	56
TS9	Sparse R-CNN [30]	ResNet101-FPN	two-stage	83	43	126

Tablica 3 sistematično prikazuje osnovne karakteristike novih modela za detekciju objekata opisanih u prethodnom poglavlju. Svaki model se razlikuje u arhitekturi i broju parametara, koji su konstantni bez obzira na veličinu slike ili serije podataka. Ukupan broj parametara podijeljen je na parametre metode i osnovne mreže kako bi se prikazala složenost arhitekture. Stupac ”Called” odnosi se na skraćenicu koja će se koristiti za modele u daljnjim grafičkim prikazima i tablicama.



Slika 17: Grafički prikaz eksperimenta

Kao što je prikazano na Slici 17, istraživanje je provedeno na skupu podataka BDD100K; podaci su proslijeđeni kroz sve modele za detekciju objekata spomenute u prethodnom poglavlju i navedene u Tablici 3. Odabrali smo modele za detekciju dostupne u BDD100K model zoo. Potiskivanje ne-maksimalne vrijednosti (*eng. Non-max suppression, NMS*) primjenjuje se kao postprocesorski korak za filtriranje dobivenih rezultata i odabir najprikladnijeg obuhvatnog okvira za objekt. Neki modeli, kao što su Faster R-CNN, FCOS, RetinaNet itd., već koriste potiskivanje ne-maksimalne vrijednosti; za ostale modele, ona će se primijeniti u ovom eksperimentu. Potiskivanje ne-maksimalne vrijednosti započinje odabirom obuhvatnog okvira s najvišim rezultatom vjerojatnosti s popisa obuhvatnih okvira povezanih s objektom. Svi ostali okviri uspoređuju se i potiskuju ako je međusobno preklapanje (IoU) između njih veće od unaprijed definiranog praga. IoU, poznat i kao Jaccardov indeks, najčešće je korištena metrika za usporedbu sličnosti između dva proizvoljna oblika. IoU kodira svojstva oblika objekata koji se uspoređuju, npr. širine, visine i lokacije dvaju obuhvatnih okvira, u svojstvo područja, a zatim izračunava normaliziranu mjeru koja se fokusira na njihove površine (ili volumene) [28]. Proces NMS ponavlja se dok se ne obradi posljednji obuhvatni okvir povezan s objektom.

Nakon ovih koraka, modeli za detekciju objekata mogu se usporediti prema različitim kriterijima. Vrijeme izvođenja (*eng. inference time*) važna je metrika kada je u pitanju evaluacija performansi detekcije objekata, pa će kriterij za prvu usporedbu biti vrijeme potrebno za završetak prolaza kroz mrežu (*eng. forward pass*). U drugoj usporedbi, usporedit ćemo točnost modela na temelju veličine objekta.

Vrijeme izvođenja izračunava se zasebno za svaku metodu, koristeći istu hardversku i softversku konfiguraciju. Svi testovi provedeni su na prijenosnom računalu s i7 procesorom, RTX 3070 grafičkom karticom i 16 GB RAM-a, koristeći PyTorch, MMDET i MMCV biblioteku. Vrijeme je izmjereno za prvu sliku veličine 1280x720x3. Navedeno vrijeme uključuje vrijeme potrebno za postavljanje resursa sustava, na primjer, postavljanje CUDA jezgri i memorije, te bi se značajno smanjilo kada bi se ista sesija koristila za više slika. Iako navedeno vrijeme uključuje dodatne operacije i ne odražava broj sličica u sekundi (*eng. frames per second, FPS*), relativan odnos između metoda u smislu trajanja prolaza kroz mrežu ostaje isti.

Drugi kriterij usporedbe je performansa metoda koristeći srednju prosječnu preciznost (mAP) s pragom IoU=0.5, kako je definirano u PASCAL VOC izazovu iz 2012. godine. PASCAL VOC izazov koristio je 11-točkastu interpoliranu preciznost i mAP preko svih klasa kako bi rangirao performanse modela [20]. Ista procedura primijenjena je u našem istraživanju kroz Python alat/kod koji su objavili Pardilla et al. [21].

Na Slici 17 može se vidjeti da je početni skup podataka podijeljen u četiri kategorije na temelju veličine objekata. Svaki objekt čija je dijagonala manja ili jednaka 10. percentilu klasificira se kao mali objekt; od 10. percentila do medijana kao mali-srednji objekt; od medijana do 90. percentila kao srednji-veliki objekt. Na kraju, objekti s dijagonalom u 90. percentilu ili većoj klasificirani su kao veliki. Ova klasifikacija korištena je za ispitivanje performansi svake metode u različitim scenarijima prometnog okruženja. Učestalost klasa u skupinama značajno se razlikuje. Na primjer, većina malih objekata su semafori, udaljeni automobili i semafori, dok su veliki objekti automobili, kamioni i pješaci u blizini.

Popis tri najčešće prisutne klase u svakoj kategoriji prikazan je u Tablici 4. Cilj ove klasifikacije je istražiti performanse na različitim veličinama objekata, tj. vidjeti kako se modeli ponašaju na malim i udaljenim objektima u usporedbi s velikim i bliskim objektima. Dodatno, modeli su testirani na izvornim podacima bez ikakve augmentacije.

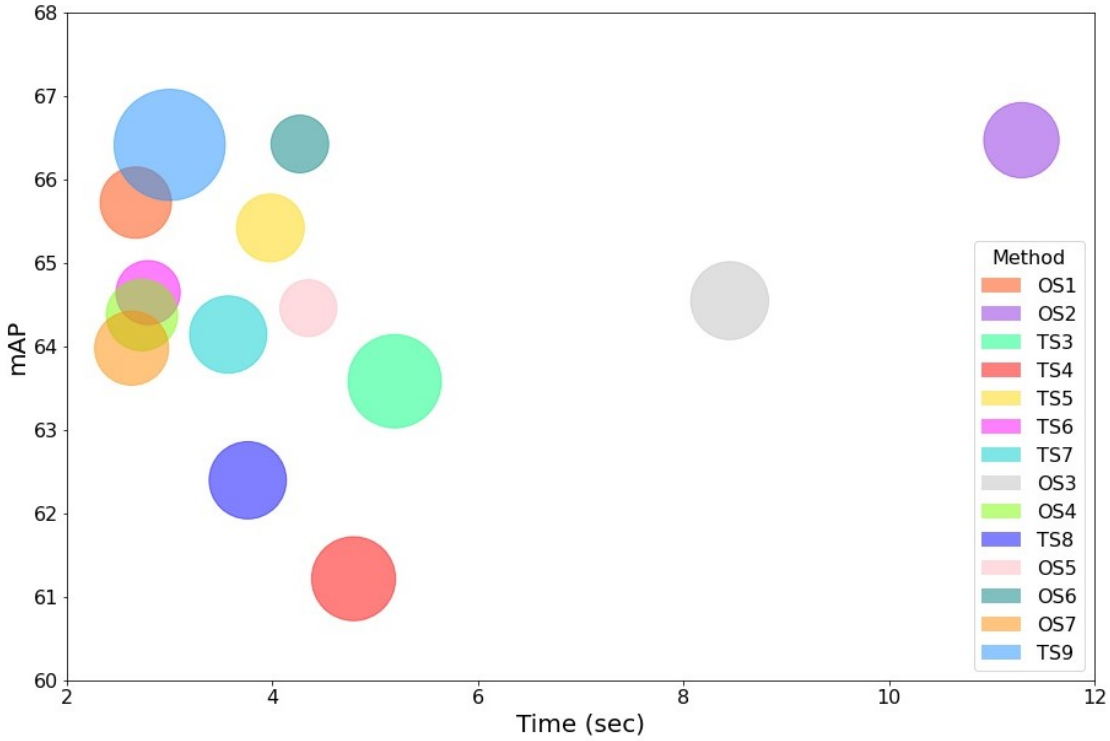
Tablica 4: Tri najzastupljenija objekta u svakoj skupini

	1st	2nd	3rd
Small objects	traffic_light	car	traffic_sign
Mid-small objects	car	traffic_sign	traffic_light
Mid-large	car	traffic_sign	pedestrian
Large	car	truck	pedestrian

7.4 Rezultati i rasprava

Naš pristup evaluiramo na skupu podataka BDD100K, najvećem skupu videozapisa vožnje sa 100 000 videozapisa i 10 zadataka za procjenu napretka algoritama prepoznavanja slika u autonomnoj vožnji [40]. Skup podataka sadrži različite vrste scena, kao što su gradske ulice, stambena područja i autoceste u različitim vremenskim uvjetima i u različito doba dana. Ovo je vrlo korisno za treniranje modela i bolje razumijevanje uličnih scena. Ukupno postoje anotacije za 100 000 slika, a svaka anotacija sadrži oznake okvira za 10 klasa objekata: bicikl, autobus, automobil, motocikl, pješak, vozač, semafor, prometni znak, vlak i kamion. Slike su dobivene iz videozapisa prikupljenih s više od 50 000 vožnji koje pokrivaju New York, područje zaljeva San Francisco i Berkeley. Zajedno sa skupom podataka, BDD100K pruža popularne modele za detekciju objekata koji su trenirani na 30 000 slika, zajedno s njihovim težinama, rezultatima evaluacije, predikcijama, vizualizacijama i skriptama za evaluaciju performansi i vizualizaciju. U usporedbi su korišteni upravo ti pretrenirani modeli.

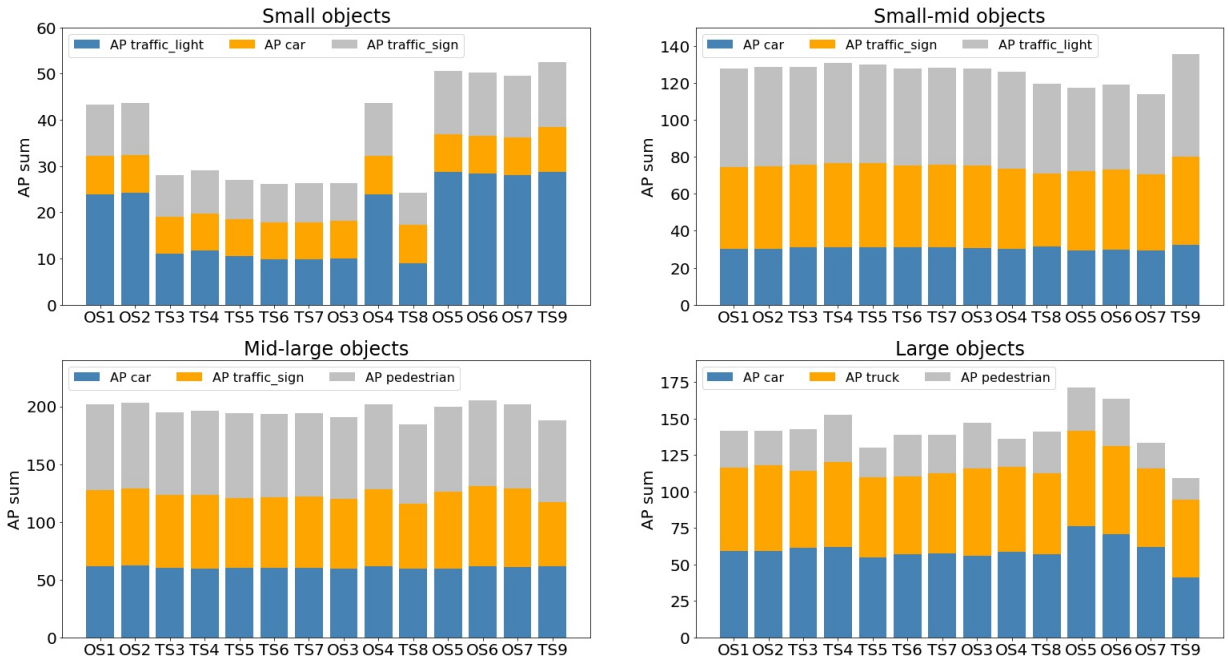
Slika 18 prikazuje generalne performanse modela za detekciju objekata na cijelom skupu podataka (svi objekti uključeni). Uočimo kako mnogi modeli postižu slične performanse sa vrlom malom razlikom u vremenu inferencije te tako zajedno čine klaster. Modeli iz klastera će biti dalje razmatrani, dok će ostalih pet modela biti odbijeni zbog nedostatka performansi ili visokog vremena inferencije. Nadalje, iz vizualizacije jasno vidimo da dvostupanjski modeli imaju veći broj parametara. Isti zaključak vrijedi i za transformere. Jednostupanjski transformeri imaju manje parametra, dok dvostupanjski transformeri imaju značajniji broj parametara.



Slika 18: Usporedba vremena izvođenja, srednje prosječne preciznosti i broja parametara. Grafikon je generiran na temelju Tablice 3 i Tablice 5.

Najbrži rezultati postignuti su s ResNet osnovom, što ukazuje da je ResNet dobar temelj za brzo vrijeme inferencije. Ukupno vrijeme izračunava se zbrajanjem vremena osnovne mreže i metode, što znači da ako je samo jedan dio brz, ne znači nužno da je cijeli model također brz. Arhitektura metode treba biti pažljivo odabrana kako ne bi utjecala na ukupno vrijeme. Zanimljivo je recimo napomenuti da je unatoč upotrebi ResNet osnove, ATSS s Dyheadom najsporiji model.

Iako je mAP sličan za sve modele i objekte unutar klastera, rezultati detekcije u definiranim klasterima značajno se razlikuju. Detaljni rezultati za svaku klasu svih modela u svakoj skupini mogu se pronaći u Tablicama 5, 6, 7, 8 i 9 u dodatku. Te razlike mogu se uočiti na Slici 19, koja prikazuje performanse tri najzastupljenije klase objekata u svakoj skupini. Detekcija malih objekata jedan je od najizazovnijih zadataka, gdje općenito jednostupanjski modeli postižu puno bolje rezultate od dvostupanjskih modela. Izuzetak je Sparse R-CNN koji ima dobre performanse na malim objektima.



Slika 19: Usporedba AP-a na tri najzastupljenija objekta u svakoj skupini prema veličini objekta

Objekti srednje veličine, kako u skupinama malih-srednjih, tako i u srednjih-velikih, pokazuju najmanje razlike između predikcija modela. S druge strane, transformeri su znatno bolji na velikim objektima. Ove razlike treba uzeti u obzir prilikom odabira modela i odluku treba donijeti na temelju karakteristika skupa podataka.

7.5 Zaključak istraživanja

Ovaj je rad imao za cilj usporediti nove napredne modele za detekciju objekata na prometnim podatcima te dati preporuku temeljenu na njihovim performansama. Performanse smo mjerili koristeći dva kriterija: vrijeme izvođenja i mAP. Osim toga, skup podataka podijelili smo u četiri kategorije kako bismo istražili koliko dobro modeli predviđaju granice okvira na temelju veličine objekta.

Modeli koji koriste ResNet postigli su najbolje rezultate u vremenu izvođenja, dok su modeli s transformerima kao osnovnom mrežom dosljedno postizali slična vremena, ali su bili sporiji od modela s CNN osnovnim mrežama. Složenost metode također pridonosi ukupnom vremenu, no pažljivim odabirom arhitekture temelje na ResNet-u može se postići dobra vremenska izvedba. Na primjer, ResNet u kombinaciji sa Sparse

R-CNN-om rezultira jednim od najboljih modela koje smo testirali, s najvećim brojem parametara.

Različite metode temeljene na različitim osnovama postigle su visoku točnost. Najbolji modeli prema mAP metrici su TS9, OS6 i OS2. Njihove su performanse na cijelom skupu podataka slične, ali postoje značajne razlike u performansama na malim i velikim objektima. TS9, dvostupanjski model, znatno bolje prepoznaje male objekte, dok transformer OS6 najbolje prepoznaje velike objekte. Oba modela također imaju dobru vremensku izvedbu, pa ih treba razmotriti kao najbolje kandidate pri odabiru modela za detekciju objekata na prometnim podatcima.

8 Zaključak

U ovom radu istražili smo dvije ključne arhitekture koje se primjenjuju u raznim zadatcima i domenama dubokog učenja — konvolucijske neuronske mreže i transformere. Konvolucijske neuronske mreže su svoju popularnost stekle još 2012. godine objavom AlexNet mreže i od tada predstavljaju prvi izbor za zadatke računalnog vida. S druge strane, transformeri predstavljaju relativno noviju arhitekturu koja je svoju prvu primjenu pronašla u zadatcima obrade prirodnog jezika. Njihova primjena u području računalnog vida započela je objavom ViT modela 2020. godine, te vrlo brzo stekla veliku popularnost i postigla performanse na razini *state-of-the-art* modela.

Osim temeljnog pregleda arhitektura i njihovog razvoja, u nastavku rada pružili smo detaljnu analizu njihovih ključnih razlika te proveli istraživanje na prometnim podatcima uzimajući u obzir različite kriterije usporedbe. Istraživanje je pokazalo da ako uzmemo u obzir cijeli skup podataka BDD1OOK, modeli postižu slične preformanse s vrlo malom razlikom u vremenu inferencije, ali razlike su veće ako mjerimo performanse detekcije različitih veličina objekata.

Analiza arhitektura i njihovih različitih pristupa pokazuje značajne razlike između transformera i konvolucijskih neuronskih mreža, što objašnjava varijacije u performansama modela temeljenih na ovim arhitekturama. Konvolucijske neuronske mreže koriste konvolucijske slojeve za prepoznavanje lokalnih značajki kroz hijerarhijski složene filtere, što im omogućuje da učinkovito prepoznaju detalje poput rubova i tekstura. Modeli poput TS9 modela koji kombinira Sparse R-CNN s ResNet101-FPN osnovom, pokazuje iznimnu učinkovitost u prepoznavanju malih objekata zahvaljujući svojoj sposobnosti da izgradi složene hijerarhijske značajke.

S druge strane, transformeri obrađuju slike kao sekvence i koriste mehanizme pažnje za integraciju globalnih i lokalnih informacija. Na primjer, model OS6, koji kombinira RetinaNet s PVTv2-B1 osnovom, pokazuje superiorne performanse u prepoznavanju velikih objekata upravo zbog globalne perspektive koja mu omogućuje da bolje hvataju dugoročne ovisnosti i kontekstualne informacije.

U zaključku, dok su konvolucijske neuronske mreže izuzetno učinkovite u hvatanju i izgradnji hijerarhijskih lokalnih značajki, transformeri nude globalniji pristup ekstrakciji značajki, s većom ujednačenošću reprezentacija kroz slojeve i sposobnošću da učinkovitije koriste velike skupove podataka. Obje arhitekture imaju svoje prednosti, a izbor između njih često ovisi o specifičnim zahtjevima zadatka i prirodi podataka. Možemo reći da su konvolucijske arhitekture bolje za zadatke koji zahtijevaju detaljno prepoznavanje lokalnih uzoraka, dok transformerske arhitekture pokazuju svoje prednosti u situacijama gdje je globalni kontekst ključan.

Literatura

[1] Azzedine Boukerche i Zhijun Hou. “Object Detection Using Deep Learning Methods in Traffic Scenarios”. *ACM Comput. Surv.* 54.2 (ožujak 2021.). ISSN: 0360-0300. DOI: 10.1145/3434398. URL: https://doi.org/10.1145/3434398.

[2] Tom B. Brown i dr. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL]. URL: https://arxiv.org/abs/2005.14165.

[3] Zhaowei Cai i Nuno Vasconcelos. *Cascade R-CNN: Delving into High Quality Object Detection*. 2017. DOI: 10.48550/ARXIV.1712.00726. URL: https://arxiv.org/abs/1712.00726.

[4] Nicolas Carion i dr. “End-to-End Object Detection with Transformers”. *European Conference on Computer Vision*. 2020., str. 213–229.

[5] *Convolutional Neural Networks*. Accessed: 2024-06-22. IBM. URL: https://www.ibm.com/topics/convolutional-neural-networks.

[6] Jifeng Dai i dr. *Deformable Convolutional Networks*. 2017. DOI: 10.48550/ARXIV.1703.06211. URL: https://arxiv.org/abs/1703.06211.

[7] Jacob Devlin i dr. *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. 2019. arXiv: 1810.04805 [cs.CL]. URL: https://arxiv.org/abs/1810.04805.

[8] Alexey Dosovitskiy i dr. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. arXiv: 2010.11929 [cs.CV]. URL: https://arxiv.org/abs/2010.11929.

[9] Gopi Krishna Erabati, Nuno Gonçalves i Helder Araujo. “Object Detection in Traffic Scenarios - A Comparison of Traditional and Deep Learning Approaches”. Srpanj 2020., str. 225–237. DOI: 10.5121/csit.2020.100918.

[10] Ross Girshick. “Fast R-CNN”. *2015 IEEE International Conference on Computer Vision (ICCV)*. 2015., str. 1440–1448. DOI: 10.1109/ICCV.2015.169.

[11] Kaiming He i dr. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385.

[12] Licheng Jiao i dr. “A Survey of Deep Learning-Based Object Detection”. *IEEE Access* 7 (2019.), str. 128837–128868. DOI: 10.1109/ACCESS.2019.2939201.

[13] Sun ke i dr. “Deep High-Resolution Representation Learning for Human Pose Estimation”. Lipanj 2019., str. 5686–5696. DOI: 10.1109/CVPR.2019.00584.

[14] Salman Khan i dr. “Transformers in Vision: A Survey”. *ACM Computing Surveys* 54.10s (siječanj 2022.), str. 1–41. ISSN: 1557-7341. DOI: 10.1145/3505244. URL: http://dx.doi.org/10.1145/3505244.

[15] Alex Krizhevsky, Ilya Sutskever i Geoffrey E. Hinton. “ImageNet classification with deep convolutional neural networks”. *Commun. ACM* 60.6 (svibanj 2017.), str. 84–90. ISSN: 0001-0782. DOI: 10.1145/3065386. URL: https://doi.org/10.1145/3065386.

[16] V7 Labs. *Activation Functions in Neural Networks [12 Types & Use Cases]*. Accessed: 2024-06-25. 2021. URL: https://www.v7labs.com/blog/neural-networks-activation-functions.

[17] Zewen Li i dr. “A Survey of Convolutional Neural Networks: Analysis, Applications, and Prospects”. *IEEE Transactions on Neural Networks and Learning Systems* 33.12 (2022.), str. 6999–7019. DOI: 10.1109/TNNLS.2021.3084827.

[18] Tsung-Yi Lin i dr. “Focal Loss for Dense Object Detection”. *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017., str. 2999–3007. DOI: 10.1109/ICCV.2017.324.

[19] Ze Liu i dr. “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”. *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021., str. 9992–10002. DOI: 10.1109/ICCV48922.2021.00986.

[20] R. Padilla, S. L. Netto i E. A. B. da Silva. “A Survey on Performance Metrics for Object-Detection Algorithms”. *2020 International Conference on Systems, Signals and Image Processing (IWSSIP)*. 2020., str. 237–242.

[21] Rafael Padilla i dr. “A Comparative Analysis of Object Detection Metrics with a Companion Open-Source Toolkit”. *Electronics* 10.3 (2021.). ISSN: 2079-9292.

DOI: 10.3390/electronics10030279. URL: https://www.mdpi.com/2079-9292/10/3/279.

[22] Jiangmiao Pang i dr. “Libra R-CNN: Towards Balanced Learning for Object Detection”. *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019., str. 821–830. DOI: 10.1109/CVPR.2019.00091.

[23] Alec Radford i dr. “Improving language understanding by generative pre-training”. *OpenAI* (2018.).

[24] Alec Radford i dr. “Language Models are Unsupervised Multitask Learners”. 2019. URL: https://api.semanticscholar.org/CorpusID:160025533.

[25] Maithra Raghu i dr. *Do Vision Transformers See Like Convolutional Neural Networks?* 2022. arXiv: 2108.08810 [cs.CV]. URL: https://arxiv.org/abs/2108.08810.

[26] Joseph Redmon i dr. *You Only Look Once: Unified, Real-Time Object Detection*. 2015. DOI: 10.48550/ARXIV.1506.02640. URL: https://arxiv.org/abs/1506.02640.

[27] Shaoqing Ren i dr. *Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks*. 2015. DOI: 10.48550/ARXIV.1506.01497. URL: https://arxiv.org/abs/1506.01497.

[28] Hamid Rezatofighi i dr. “Generalized intersection over union: A metric and a loss for bounding box regression”. *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 2019., str. 658–666.

[29] Karen Simonyan i Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV]. URL: https://arxiv.org/abs/1409.1556.

[30] Peize Sun i dr. “Sparse R-CNN: End-to-End Object Detection with Learnable Proposals”. Lipanj 2021., str. 14449–14458. DOI: 10.1109/CVPR46437.2021.01422.

[31] Zhi Tian i dr. *FCOS: Fully Convolutional One-Stage Object Detection*. 2019. DOI: 10.48550/ARXIV.1904.01355. URL: https://arxiv.org/abs/1904.01355.

[32] Hugo Touvron i dr. *Training data-efficient image transformers distillation through attention*. 2021. arXiv: 2012.12877 [cs.CV]. URL: https://arxiv.org/abs/2012.12877.

[33] Shikhar Tuli i dr. *Are Convolutional Neural Networks or Transformers more like human vision?* 2021. arXiv: 2105.07197 [cs.CV]. URL: https://arxiv.org/abs/2105.07197.

[34] Lucia Varesko i Goran Oreski. "Performance comparison of novel object detection models on traffic data". *Proceedings of the 2023 8th International Conference on Machine Learning Technologies*. ICMLT '23. Stockholm, Sweden: Association for Computing Machinery, 2023., str. 177–184. ISBN: 9781450398329. DOI: 10.1145/3589883.3589910. URL: https://doi.org/10.1145/3589883.3589910.

[35] Ashish Vaswani i dr. *Attention Is All You Need*. 2023. arXiv: 1706.03762 [cs.CL]. URL: https://arxiv.org/abs/1706.03762.

[36] Guanqun Wang i dr. "P2FEViT: Plug-and-Play CNN Feature Embedded Hybrid Vision Transformer for Remote Sensing Image Classification". *Remote Sensing* 15.7 (2023.). ISSN: 2072-4292. DOI: 10.3390/rs15071773. URL: https://www.mdpi.com/2072-4292/15/7/1773.

[37] Wenhai Wang i dr. "PVT v2: Improved baselines with Pyramid Vision Transformer". *Computational Visual Media* 8.3 (ožujak 2022.), str. 415–424. DOI: 10.1007/s41095-022-0274-8. URL: https://doi.org/10.1007%2Fs41095-022-0274-8.

[38] Wenhai Wang i dr. "Pyramid Vision Transformer: A Versatile Backbone for Dense Prediction without Convolutions". *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021., str. 548–558. DOI: 10.1109/ICCV48922.2021.00061.

[39] Xiongwei Wu, Doyen Sahoo i Steven C. H. Hoi. *Recent Advances in Deep Learning for Object Detection*. 2019. DOI: 10.48550/ARXIV.1908.03673. URL: https://arxiv.org/abs/1908.03673.

[40] Fisher Yu i dr. *BDD100K: A Diverse Driving Dataset for Heterogeneous Multitask Learning*. 2018. DOI: 10.48550/ARXIV.1805.04687. URL: https://arxiv.org/abs/1805.04687.

[41] Syed Sahil Abbas Zaidi i dr. *A Survey of Modern Deep Learning based Object Detection Models*. 2021. DOI: 10.48550/ARXIV.2104.11892. URL: https://arxiv.org/abs/2104.11892.

[42] Aston Zhang i dr. *Dive into Deep Learning*. 2023. arXiv: 2106.11342 [cs.LG]. URL: https://arxiv.org/abs/2106.11342.

[43] Shifeng Zhang i dr. *Bridging the Gap Between Anchor-based and Anchor-free Detection via Adaptive Training Sample Selection*. 2019. DOI: 10.48550/ARXIV.1912.02424. URL: https://arxiv.org/abs/1912.02424.

[44] Zhong-Qiu Zhao i dr. "Object Detection With Deep Learning: A Review". *IEEE Transactions on Neural Networks and Learning Systems* 30.11 (2019.), str. 3212–3232. DOI: 10.1109/TNNLS.2018.2876865.

[45] Xizhou Zhu i dr. *Deformable ConvNets v2: More Deformable, Better Results*. 2018. DOI: 10.48550/ARXIV.1811.11168. URL: https://arxiv.org/abs/1811.11168.

Popis slika

Popis tablica

A Rezultati istraživanja

Tablica 5: Rezultati detekcije na BDD100K skupu podataka s vremenom inferencije

Method	Backbone	AP bicycle	AP bus	AP car	AP motorcycle	AP pedestrian	AP rider	AP traffic_light	AP traffic_sign	AP train	AP truck	mAP	Time (sec)
ATSS	ResNet101-FPN	62.30	68.40	91.93	55.83	79.26	60.21	84.48	80.81	2.78	71.28	65.73	2.67
ATSS	ResNet101-FPN + Dyhead	62.94	69.58	92.36	57.97	80.41	61.56	85.21	81.41	0.30	73.03	66.48	11.29
Cascade R-CNN	ResNet101-FPN	61.74	68.17	91.00	56.71	78.31	62.12	71.19	76.75	0.00	69.88	63.59	5.19
Cascade R-CNN	Swin-T	58.74	65.34	83.32	55.93	73.03	61.37	66.11	71.41	10.76	66.19	61.22	4.79
Faster R-CNN	HRNet32	65.37	70.00	91.28	61.29	80.10	66.04	71.37	76.89	0.00	71.91	65.43	3.98
Faster R-CNN	ResNet50-FPN	63.34	68.31	90.74	60.76	78.79	64.29	70.27	75.90	4.39	69.70	64.65	2.79
Faster R-CNN	ResNet101-FPN	63.20	68.96	90.87	59.00	78.52	63.12	70.13	76.39	0.00	71.28	64.15	3.57
Deformable ConvNets	ResNet101-FPN	63.87	69.15	91.08	60.81	78.69	63.08	70.23	76.73	0.00	71.89	64.55	8.45
FCOS	ResNet101-FPN	61.11	67.95	91.40	54.71	78.26	57.90	83.87	79.85	0.05	68.68	64.38	2.73
Libra R-CNN	ResNet101-FPN	60.63	68.14	90.93	57.79	75.42	62.82	66.98	70.48	0.00	70.81	62.40	3.76
RetinaNet	PVT-S	61.51	67.68	90.47	57.50	76.22	58.70	83.29	79.00	2.01	68.25	64.46	4.35
RetinaNet	PVTv2-B1	65.84	68.93	91.10	60.42	79.28	63.07	83.84	80.55	2.04	69.24	66.43	4.27
RetinaNet	ResNet101-FPN	62.67	67.49	90.28	56.80	75.80	58.09	81.36	78.20	0.00	69.14	63.98	2.63
Sparse R-CNN	ResNet101-FPN	60.05	68.40	92.59	56.85	78.34	61.30	87.90	82.53	5.93	70.31	66.42	3.00

Tablica 6: Rezultati detekcije malih objekata na BDD100K skupu podataka

Method	Backbone	AP bicycle	AP bus	AP car	AP motorcycle	AP pedestrian	AP rider	AP traffic_light	AP traffic_sign	AP truck	Small objects mAP
ATSS	ResNet101-FPN	0.07	0.16	8.23	0.30	1.13	0.27	23.90	11.17	0.28	5.06
ATSS	ResNet101-FPN + Dyhead	0.06	0.19	8.21	0.33	1.19	0.35	24.18	11.32	0.31	5.13
Cascade R-CNN	ResNet101-FPN	0.09	0.19	7.99	0.46	1.07	0.32	11.11	9.03	0.31	3.40
Cascade R-CNN	Swin-T	0.03	0.23	8.09	0.44	1.11	0.59	11.70	9.30	0.31	3.53
Faster R-CNN	HRNet32	0.05	0.17	7.98	0.56	1.04	0.52	10.52	8.42	0.29	3.28
Faster R-CNN	ResNet50-FPN	0.03	0.24	8.03	0.91	0.97	0.49	9.80	8.27	0.30	3.22
Faster R-CNN	ResNet101-FPN	0.07	0.17	7.98	0.51	0.99	0.53	9.88	8.42	0.33	3.21
Deformable ConvNets	ResNet101-FPN	0.08	0.17	8.05	0.42	0.98	0.30	10.03	8.27	0.27	3.17
FCOS	ResNet101-FPN	0.17	0.41	8.22	0.41	1.15	0.30	23.95	11.42	0.22	5.14
Libra R-CNN	ResNet101-FPN	0.15	0.23	8.30	0.65	0.95	0.52	8.94	6.94	0.30	3.00
RetinaNet	PVT-S	0.05	0.18	8.23	0.38	1.24	0.38	28.72	13.60	0.30	5.90
RetinaNet	PVTv2-B1	0.25	0.14	8.09	0.45	1.50	0.45	28.43	13.70	0.38	5.93
RetinaNet	ResNet101-FPN	0.06	0.16	8.16	0.49	1.30	0.39	27.96	13.37	0.25	5.79
Sparse R-CNN	ResNet101-FPN	0.41	0.24	9.71	1.32	1.54	0.57	28.82	14.05	0.43	6.34

Tablica 7: Rezultati detekcije objekata srednje veličine na BDD100K skupu podataka

Method	Backbone	AP bicycle	AP bus	AP car	AP motorcycle	AP pedestrian	AP rider	AP traffic_light	AP traffic_sign	AP truck	Small-mid objects mAP
ATSS	ResNet101-FPN	3.70	3.68	30.07	5.00	17.89	5.99	53.40	44.36	5.41	18.83
ATSS	ResNet101-FPN + Dyhead	4.06	3.63	30.31	4.83	18.42	6.14	53.90	44.55	5.86	19.08
Cascade R-CNN	ResNet101-FPN	5.50	4.17	31.04	5.68	19.20	7.91	52.86	44.65	6.57	19.73
Cascade R-CNN	Swin-T	5.65	4.07	31.25	6.91	19.80	8.63	53.95	45.57	6.90	20.30
Faster R-CNN	HRNet32	6.71	4.38	31.09	7.93	19.58	8.60	53.65	45.36	7.06	20.48
Faster R-CNN	ResNet50-FPN	5.83	3.94	30.95	7.27	19.17	8.05	52.39	44.49	6.40	19.83
Faster R-CNN	ResNet101-FPN	5.97	4.24	31.00	7.49	19.25	8.34	52.60	44.78	6.61	20.03
Deformable ConvNets	ResNet101-FPN	5.31	3.62	30.78	7.56	19.23	7.83	52.67	44.38	6.35	19.75
FCOS	ResNet101-FPN	4.75	3.31	30.17	4.58	17.83	5.43	52.50	43.48	5.08	18.57
Libra R-CNN	ResNet101-FPN	5.55	3.73	31.50	6.98	14.74	9.24	48.26	39.58	6.41	18.44
RetinaNet	PVT-S	4.41	3.60	29.50	6.16	16.67	4.91	45.40	42.66	5.27	17.62
RetinaNet	PVTv2-B1	4.52	3.78	29.82	6.39	18.50	6.18	45.82	43.43	6.10	18.28
RetinaNet	ResNet101-FPN	4.34	3.23	29.45	4.32	16.24	4.74	43.59	40.89	4.83	16.85
Sparse R-CNN	ResNet101-FPN	6.05	4.78	32.29	9.97	20.32	8.96	55.49	47.95	6.79	21.40

Tablica 8: Rezultati detekcije srednje velikih objekata u skupu podataka BDD100K

Method	Backbone	AP bicycle	AP bus	AP car	AP motorcycle	AP pedestrian	AP rider	AP traffic_light	AP traffic_sign	AP train	AP truck	Mid-large objects mAP
ATSS	ResNet101-FPN	56.04	30.38	61.90	47.02	74.36	48.53	58.80	65.33	0.00	36.21	47.86
ATSS	ResNet101-FPN + Dyhead	56.37	30.53	61.99	49.56	74.39	49.93	60.68	66.79	0.00	36.06	48.63
Cascade R-CNN	ResNet101-FPN	52.96	30.05	59.97	47.09	71.64	47.21	63.33	63.24	0.00	34.84	47.03
Cascade R-CNN	Swin-T	54.22	31.51	59.83	46.63	72.31	50.85	62.96	63.72	0.00	35.35	47.74
Faster R-CNN	HRNet32	56.05	32.06	60.30	50.97	73.26	50.68	55.67	60.22	0.00	35.93	47.51
Faster R-CNN	ResNet50-FPN	55.75	30.67	60.14	49.06	72.18	49.13	61.10	61.04	0.00	35.97	47.50
Faster R-CNN	ResNet101-FPN	55.27	31.10	60.12	49.53	71.94	48.86	61.95	61.87	0.00	35.86	47.65
Deformable ConvNets	ResNet101-FPN	55.28	29.98	59.19	44.26	71.21	48.53	57.78	60.35	0.00	33.24	45.98
FCOS	ResNet101-FPN	54.82	31.00	61.65	45.48	73.62	48.87	32.17	66.65	0.00	34.59	44.89
Libra R-CNN	ResNet101-FPN	52.85	30.04	59.30	47.80	69.26	47.99	54.02	56.18	0.00	34.60	45.20
RetinaNet	PVT-S	52.68	30.37	59.87	45.28	73.03	46.53	64.65	66.46	0.23	33.17	47.23
RetinaNet	PVTv2-B1	57.24	31.40	61.49	46.70	73.86	48.81	69.21	69.63	0.00	33.55	49.19
RetinaNet	ResNet101-FPN	56.37	30.30	60.64	49.84	72.91	49.83	67.95	68.12	0.00	35.59	49.15
Sparse R-CNN	ResNet101-FPN	51.50	29.75	61.61	43.02	70.95	44.46	49.28	55.36	0.00	34.50	44.04

Tablica 9: Rezultati detekcije velikih objekata na BDD100K skupu podataka

Method	Backbone	AP bicycle	AP bus	AP car	AP motorcycle	AP pedestrian	AP rider	AP traffic_light	AP traffic_sign	AP train	AP truck	Large objects mAP
ATSS	ResNet101-FPN	24.50	64.29	59.02	30.22	25.12	41.37	0.05	14.65	4.42	57.25	32.09
ATSS	ResNet101-FPN + Dyhead	20.56	65.38	58.95	32.41	23.77	32.89	0.09	7.75	0.40	58.95	30.12
Cascade R-CNN	ResNet101-FPN	22.43	57.27	61.68	24.89	28.65	45.12	0.09	22.40	0.00	52.45	31.50
Cascade R-CNN	Swin-T	28.06	66.27	62.13	39.53	32.17	48.02	4.73	19.96	15.97	58.19	37.50
Faster R-CNN	HRNet32	19.06	59.43	54.62	22.80	20.72	34.06	0.04	7.60	0.00	54.92	27.33
Faster R-CNN	ResNet50-FPN	18.87	57.79	56.83	22.27	28.89	41.24	0.10	11.77	0.00	53.35	29.11
Faster R-CNN	ResNet101-FPN	19.89	59.66	57.76	20.92	26.58	40.74	0.14	11.16	0.00	54.49	29.13
Deformable ConvNets	ResNet101-FPN	20.28	66.58	55.65	32.54	31.11	45.80	0.19	16.84	6.43	60.07	33.55
FCOS	ResNet101-FPN	18.37	61.11	58.76	26.27	19.58	31.62	3.51	18.85	0.07	57.95	29.61
Libra R-CNN	ResNet101-FPN	19.85	60.43	56.84	21.47	28.49	39.35	0.02	12.22	0.00	55.88	29.46
RetinaNet	PVT-S	24.94	68.53	76.35	29.34	29.94	42.59	6.30	22.54	2.66	65.01	36.82
RetinaNet	PVTv2-B1	23.47	65.33	70.93	37.22	32.14	51.65	1.01	26.72	3.62	60.26	37.24
RetinaNet	ResNet101-FPN	18.06	61.03	61.72	21.44	17.73	35.45	0.77	14.72	0.00	53.90	28.48
Sparse R-CNN	ResNet101-FPN	15.46	65.53	40.84	21.32	14.91	34.49	0.16	7.09	7.98	53.26	26.10