Raspoznavanje prednaučenim samonadziranim i multimodalnim modelima

Kuleš, Jurica

Master's thesis / Diplomski rad

2025

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Electrical Engineering and Computing / Sveučilište u Zagrebu, Fakultet elektrotehnike i računarstva**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:168:776667

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2025-03-23**

Repository / Repozitorij:

FER Repository - University of Zagreb Faculty of Electrical Engineering and Computing repozitory

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 729

RASPOZNAVANJE PREDNAUČENIM SAMONADZIRANIM I MULTIMODALNIM MODELIMA

Jurica Kuleš

Zagreb, veljača 2025.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 729

RASPOZNAVANJE PREDNAUČENIM SAMONADZIRANIM I MULTIMODALNIM MODELIMA

Jurica Kuleš

Zagreb, veljača 2025.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 30. rujna 2024.

DIPLOMSKI ZADATAK br. 729

Pristupnik:	**Jurica Kuleš (0036524071)**
Studij:	Računarstvo
Profil:	Računarska znanost
Mentor:	prof. dr. sc. Siniša Šegvić

Zadatak: **Raspoznavanje prednaučenim samonadziranim i multimodalnim modelima**

Opis zadatka:

Raspoznavanje slika važan je problem računalnog vida s mnogim zanimljivim primjenama. U posljednje vrijeme stanje tehnike postižu duboki modeli zasnovani na konvolucijama i slojevima pažnje. Ipak, standardni posutpci ne uspijevaju postići zadovoljavajuću generalizaciju na malenim skupovima podataka za učenje. U okviru rada, potrebno je odabrati okvir za automatsku diferencijaciju te upoznati biblioteke za rukovanje tenzorima i slikama. Proučiti i ukratko opisati postojeće duboke arhitekture za raspoznavanje slika s posebnim naglaskom na samonadzirane i multimodalne modele. Odabrati slobodno dostupne skupove slika te oblikovati podskupove za učenje, validaciju i testiranje. Primijeniti naučene modele, prikazati eksperimente na nekom javnom skupu podataka te usporediti generalizacijsku izvedbu sa stanjem tehnike. Komentirati učinkovitost učenja i zaključivanja. Preporučiti smjernice za ugradnju modela u praktičan tehnički sustav. Predložiti pravce za budući rad. Radu priložiti izvorni i izvršni kod razvijenih postupaka, ispitne slijedove i rezultate, uz potrebna objašnjenja i dokumentaciju. Citirati korištenu literaturu i navesti dobivenu pomoć.

Rok za predaju rada: 14. veljače 2025.

Zahvaljujem se svojoj užoj obitelji na podršci iskazanoj tijekom moga školovanja, kao i mentoru Siniši Šegviću pri izradi ovoga rada.

Sadržaj

1. Uvod **3**

2. Paradigme strojnog učenja **5**

2.1. Nadzirano učenje 5

2.2. Nenadzirano učenje 6

2.3. Samonadzirano učenje 6

3. Duboko učenje **7**

3.1. Višeslojni perceptron 7

3.2. Aktivacijske funkcije 9

3.2.1. Sigmoida 9

3.2.2. Softmax 10

3.2.3. Zglobnica 10

3.2.4. GELU 11

3.3. Normalizacija nad grupom 12

3.4. Konvolucijski modeli 14

3.5. Rezidualne mreže 16

3.6. Modeli temeljeni na pažnji 17

3.6.1. Mehanizam pažnje 17

3.6.2. Arhitektura Transformer 19

3.6.3. Arhitektura ViT 22

4. Kontrastno predučenje uz nadzor iz prirodnog jezika **24**

4.1. CLIP 24

4.2. SigLIP 27

5. Samonadzirano učenje vizualnih transformera **30**
5.1. DINO 30
5.2. DINOv2 35

6. Skupovi podataka **38**
6.1. TACO 38
6.1.1. TACO-10 38
6.1.2. TACO-56 39

7. Implementacija **41**

8. Eksperimenti **43**
8.1. Klasifikacija bez ugađanja 43
8.2. Klasifikacija linearnom projekcijom 45

9. Zaključak **47**

Literatura **49**

Sažetak **54**

Abstract **55**

1. Uvod

Već dugi niz godina, raspoznavanje slika jedan je od glavnih izazova u računalnom vidu, s primjenama u brojnim područjima poput medicinske dijagnostike, autonomnih vozila, sigurnosnog nadzora i robotike. Sposobnost automatskog i preciznog tumačenja vizualnih podataka ključna je za razvoj inteligentnih sustava koji mogu smisleno komunicirati s okolinom. Tradicionalne tehnike računalnog vida, koje se oslanjaju na ručno dizajnirane značajke i klasične algoritme strojnog učenja, uvelike su nadmašene modelima dubokog učenja koji koriste velike količine podataka i računalne snage za učenje semantički bogatih reprezentacija.

Arhitekture dubokog učenja, posebno konvolucijske neuronske mreže i modeli temeljeni na mehanizmu pažnje, postigli su vrhunske rezultate u zadacima prepoznavanja slika. Duboki konvolucijski modeli dobro rade s kompozitnim podacima jer hijerarhijski uče i kombiniraju lokalne uzorke u značajke viših razina, pritom čuvajući prostorne odnose, što omogućuje učinkovito i robusno prepoznavanje složenih struktura. S druge strane, mehanizmi pažnje popularizirani modelima Transformer, omogućuju globalno modeliranje konteksta, poboljšavajući mogućnost prepoznavanja. Unatoč njihovim uspjesima, ovi modeli zahtijevaju opsežne skupove označenih podataka za učenje, što ih čini manje učinkovitima u scenarijima gdje je označavanje podataka skupo ili ograničeno. Ovo ograničenje, kao i prisutnost pristranosti u oznakama, krivo označavanje te općenito veća količina neoznačenih podataka, potaknuli su istraživanja u području samonadziranog učenja (engl. *self-supervised learning*, SSL) i multimodalnih modela, koji mogu koristiti neoznačene podatke ili više modaliteta kako bi poboljšali učinkovitost učenja te generalizaciju.

Samonadzirano učenje dobilo je značajan zamah kao paradigma koja omogućuje modelima učenje značajnih reprezentacija iz samih podataka, bez eksplicitnog nadzora. Ko-

rištenje prikladnih pomoćnih zadataka (engl. *pretext tasks*), samonadziranim modelima omogućuje otkrivanje struktura svojstvenih podacima, što dovodi do boljih performansi na naknadnim zadacima (engl. *downstream tasks*) uz smanjenu količinu označenih primjera. S druge strane, multimodalni modeli integriraju informacije iz različitih izvora, poput teksta i slika, kako bi poboljšali sposobnosti prepoznavanja. Kombinacija samonadziranog i multimodalnog učenja predstavlja obećavajući smjer za poboljšanje robusnosti i prilagodljivosti sustava za raspoznavanje slika.

Cilj je ovog rada, istražiti sposobnost raspoznavanja slika pomoću samonadzirano predučenog modela DINOv2 i kontrastno predučenog modela CLIP. U 2. poglavlju predstavit ćemo ključne paradigme učenja modela strojnog i dubokog učenja. U poglavlju nakon toga, proći ćemo kroz osnovne modele i metode korištene u suvremenom računalnom vidu, s naglaskom na modele temeljene na mehanizmu pažnje. U 4. poglavlju pobliže ćemo se upoznati s konceptom kontrastnog predučenja uz nadzor iz prirodnog jezika, odnosno model CLIP te njegovu inačicu SigLIP. U poglavlju nakon toga imat ćemo priliku vidjeti kako provesti postupak samonadziranog predučenja nad modelima temeljenim na arhitekturi Visual Transformer (ViT). U 6. poglavlju predstavit ćemo skupove podataka nad kojima ćemo provesti eksperimente pomoću modela definiranih u poglavlju 7. Konačno, u 8. poglavlju analizirat ćemo rezultate provedenih eksperimenata.

2. Paradigme strojnog učenja

Strojno učenje (engl. *machine learning*) je grana umjetne inteligencije koja se bavi razvojem algoritama i modela koji omogućuju računalima da uče iz podataka i donose odluke bez eksplicitnog programiranja. Ključni koncept strojnog učenja je generalizacija, odnosno sposobnost modela da se prilagodi novim, nepoznatim podacima na temelju prethodno stečenog znanja. Modeli znanje stječu kroz podatke, a ovisno o njihovoj dostupnosti i vrsti te tipu problema kojeg pokušavamo riješiti, strojno se učenje dijeli na dvije osnovne paradigme, koje ćemo opisati u nastavku poglavlja: nadzirano i nenadzirano učenje. Također ćemo se upoznati i sa samonadziranim učenjem, svojevrsnim presjekom dva prethodno spomenuta pristupa.

2.1. Nadzirano učenje

Svaki algoritam strojnog učenja na svom ulazu ima primjer (engl. *instance*) koji se sastoji od niza značajki na temelju kojih algoritam donosi odluke. Prirodno je stoga primjere promatrati kao vektore njihovih značajki (engl. *feature vector*): $\mathbf{x} = (x_1, x_2, ..., x_n)$, pri čemu su x_i pojedine značajke, a n ukupan broj značajki. U nadziranom učenju primjerima je osim njihove vektorske reprezentacije pridružena i oznaka (engl. *label*) pomoću koje algoritam strojnog učenja dobiva informaciju o pouzdanosti svoje predikcije. Oznake mogu biti brojčane vrijednosti (kod regresije) ili nebrojčane vrijednosti koje označavaju pripadnost jednom ili više razreda (engl. *class*) u klasifikacijskim problemima. Uobičajeno skupovi podataka sadrže mnoštvo primjera, gdje ukupan broj primjera označavamo s N. Cijeli skup označenih primjera (engl. *labeled dataset*) možemo prikazati skraćenom notacijom: $D = \{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$ [1].

2.2. Nenadzirano učenje

Nekada informacije o pripadnosti primjera određenom razredu nisu dostupne. Stoga u nenadziranom učenju raspolažemo skupom neoznačenih primjera $D = \{(\mathbf{x}^{(i)})\}_{i=1}^{N}$. Razlozi izostanka oznake mogu biti ti, da naprosto ne znamo kako unaprijed označiti primjere ili ih znamo označiti, ali je taj proces preskup. Neki su od uobičajenih zadataka za koje bismo koristili nenadzirano strojno učenje: grupiranje, procjena gustoće, otkrivanje stršećih vrijednosti (engl. *outlier detection*) i smanjenje dimenzionalnosti [2].

2.3. Samonadzirano učenje

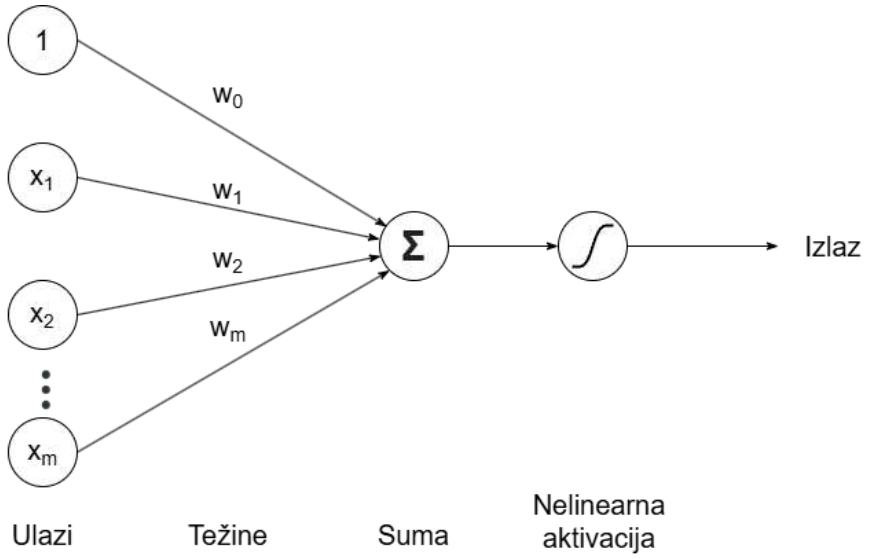
Tijekom godina modeli stanja tehnike (engl. *state-of-the-art models*) postavljali su zahtjeve za sve većim skupovima podataka, pri čemu su oni skupovi koji su bili sačinjeni isključivo od označenih primjera, u procesu treniranja činili svojevrsno usko grlo zbog dugotrajnosti i cijene procesa označavanja. S vremenom se samonadzirano učenje iskristaliziralo kao adekvatno rješenje tog problema kroz modele poput DINO-a [3]. Iako se kao i nenadzirano učenje koristi velikim količinama neoznačenih podataka, glavna razlika je u tome što prilikom treniranja modela ovaj pristup za nadzorni signal koristi temeljnu istinu (engl. *ground truth*) u obliku pseudo-oznaka dobivenih iz samih podataka, umjesto eksplicitno zadanih korelacija pomoću čovjekom-nametnutih oznaka kao u klasičnom nadziranom učenju.

3. Duboko učenje

Tradicionalne metode strojnog učenja bile su ključne za napredak umjetne inteligencije, no često se oslanjaju na ručno izrađene značajke i domensko znanje u određenom području, kako bi postigle optimalne rezultate. Takvi su modeli loše skalirali s porastom složenosti i količine podataka. Duboko učenje pojavilo se kao alternativno rješenje, korištenjem višeslojnih neuronskih mreža sposobnih za automatsko stvaranje hijerarhijskih reprezentacija iz sirovih podataka. Ova promjena paradigme ne samo da smanjuje potrebu za ručnom obradom značajki, već i omogućuje modelima učenje složenih obrazaca u visokodimenzionalnim prostorima ugrađivanja, što duboko učenje čini posebno učinkovitim u područjima poput računalnog vida, obrade prirodnog jezika i prepoznavanja govora. Korištenjem velikih skupova podataka i računalne snage, duboko učenje nadmašuje tradicionalne tehnike strojnog učenja, dovodeći do robusnijih, bolje skalirajućih i preciznijih modela za stvarne primjene. U nastavku ovog poglavlja napravit ćemo pregled osnovnih gradivnih elemenata dubokih modela te njihovih složenijih nasljednika, koje ćemo koristiti kao okosnice za provedbu naših eksperimenata.

3.1. Višeslojni perceptron

Višeslojni perceptron (engl. *multilayer perceptron*) ili duboke unaprijedne mreže (engl. *deep feedforward neural networks*) najjednostavnija su formulacija dubokog modela, a temelje se na konceptu umjetnog neurona ili perceptrona kao težinske sume svojih ulaza x_i uvećanih za pomak b i propuštenih kroz nelinearnu aktivacijsku funkciju (slika 3.1.). Jednim neuronom ne bismo mogli modelirati nikakvu suvislo složenu funkciju, stoga se u praksi koriste mreže sastavljene od slojeva takvih neurona pri čemu je svaki neuron u sloju l povezan sa svim ostalim ulazima ili neuronima u sloju $l-1$, zbog čega se za ovakve modele također ustalio naziv potpuno povezana mreža (engl. *fully connected network*).

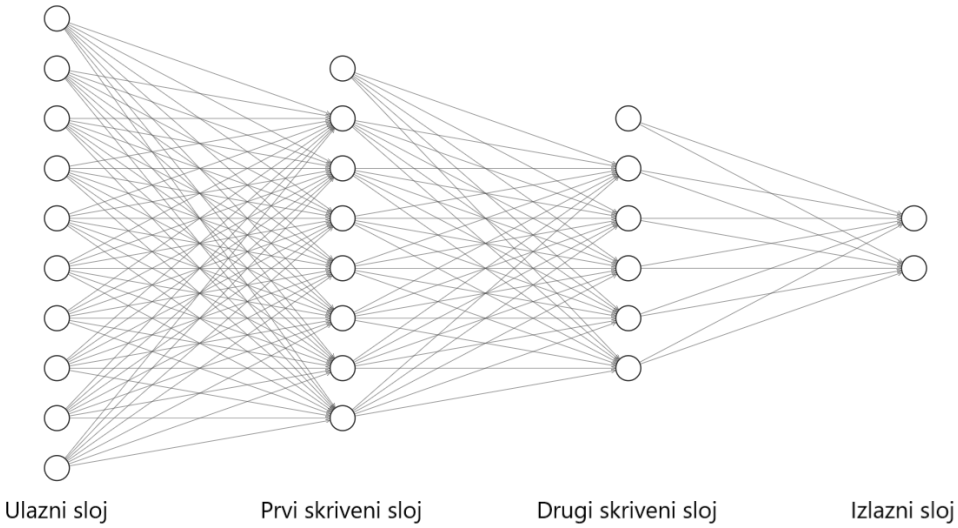


Slika 3.1. Umjetni neuron izražen u obliku težinske sume svojih ulaza, propuštene kroz nelinearnu aktivacijsku funkciju.

Radi praktičnosti, računanje izlaza svih neurona jednoga sloja možemo izraziti umnoškom matrice težina $\mathbf{W}$ i vektora značajki pojedinog primjera $\mathbf{x_i}$, na što dodamo vektor pomaka $\mathbf{b}$:

$$\mathbf{h_i} = \mathbf{W}\mathbf{x_i} + \mathbf{b}. \tag{3.1}$$

Pritom svaki redak matrice $\mathbf{W}$ predstavlja vektor težina jednog neurona, čiji se pomak nalazi u odgovarajućem elementu vektora $\mathbf{b}$.



Slika 3.2. Višeslojni perceptron kao niz potpuno povezanih slojeva umjetnih neurona.

3.2. Aktivacijske funkcije

Aktivacijske funkcije predstavljaju ključnu komponentu dubokih neuronskih mreža, jer omogućuju modelima da uče složene nelinearne odnose unutar podataka. Bez njih, neuronske bi mreže bile ograničene na linearne transformacije, što bi značajno smanjilo njihovu sposobnost rješavanja složenih problema. Aktivacijske funkcije uvode nelinearnost u model, čime omogućuju višeslojnim mrežama da aproksimiraju složene funkcije i generiraju precizne predikcije. Različite vrste aktivacijskih funkcija poput sigmoidne, zglobnice ili tangensa hiperbolnog, imaju različite karakteristike koje ih čine pogodnima za specifične zadatke i arhitekture. Odabir odgovarajuće aktivacijske funkcije ključan je za postizanje optimalnih performansi modela i stabilnosti procesa učenja. Propuštanjem izlaza neurona čitavog sloja kroz aktivacijsku funkciju f, izraz 3.1 poprima sljedeći oblik:

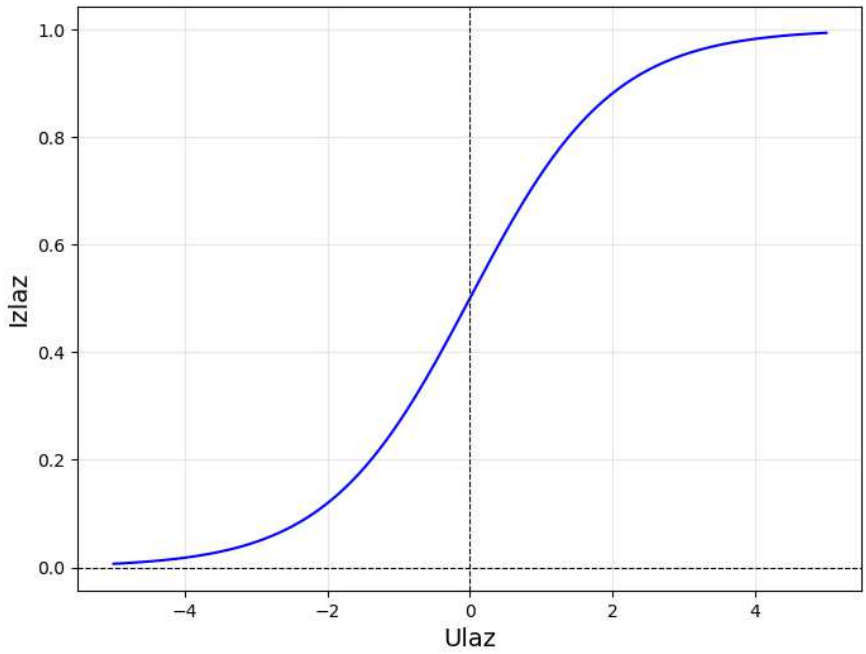
$$\mathbf{h_i} = f(\mathbf{W}\mathbf{x_i} + \mathbf{b}). \tag{3.2}$$

3.2.1. Sigmoida

Sigmoida preuzeta iz logističke regresije, definirana izrazom 3.3 te čiji je graf prikazan na slici 3.3., zbog svojih je poželjnih svojstava vrlo intuitivan odabir za aktivacijsku funkciju. Prvo lijepo svojstvo je to da ograničava svoje izlaze na otvoreni interval (0,1), što sigmoidi daje svojevrsnu vjerojatnosnu interpretaciju. Drugo, primjerima jedne klase davat će vrijednosti blizu 0, a primjerima druge klase vrijednosti blizu 1, čime dobivamo intuitivnu decizijsku granicu između dviju klasa. I treće, derivabilna je funkcija što je čini pogodnom za gradijentne metode optimizacije parametara mreže [4].

$$\sigma(x) = \frac{1}{1 + e^{-x}}. \tag{3.3}$$

No, unatoč svojim pozitivnim stranama, danas je većinski istisnuta iz uporabe zbog činjenice da u slučaju zasićenja njene vrijednosti ($x \ll 0$ ili $x \gg 0$), rezultira nestajućim gradijentima (engl. *vanishing gradients*), što nam onemogućava normalno učenje modela.



Slika 3.3. Graf sigmoidne funkcije.

3.2.2. Softmax

Funkcija softmax je poopćenje sigmoide na K razreda, a vrijednost jedne od njenih komponenata računa se izrazom:

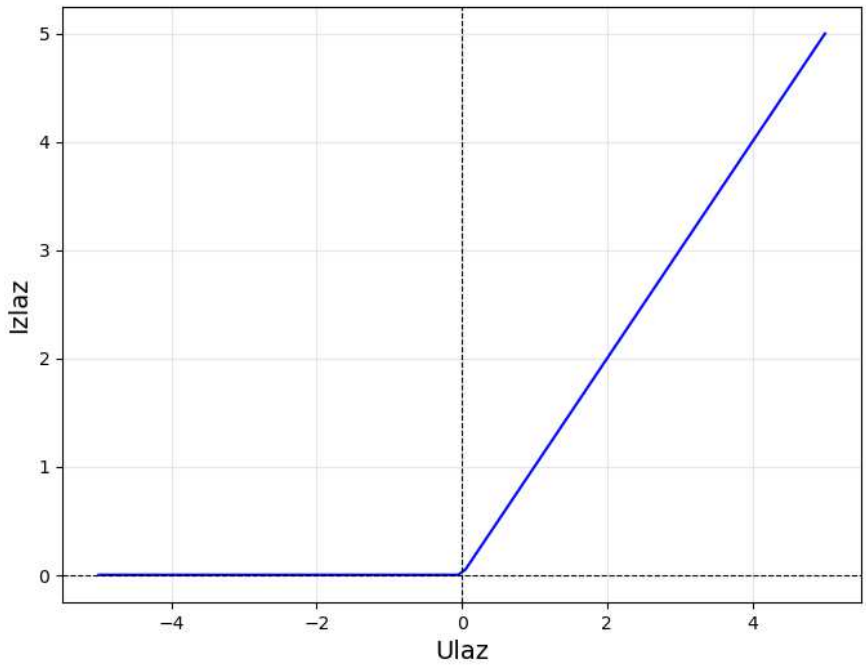
$$\text{softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}}. \tag{3.4}$$

Uobičajeno ju koristimo u zadnjem, klasifikacijskom sloju mreže jer na svojem izlazu daje vektor normaliziranih vjerojatnosti čiji se elementi mogu interpretirati kao vjerojatnosti pripadanja primjera j-tom od K razreda.

3.2.3. Zglobnica

Zglobnica (engl. *Rectified Linear Unit*, ReLU), prikazana grafom na slici 3.4., jedna je od najčešće korištenih aktivacijskih funkcija u suvremenim modelima dubokog učenja. Tomu je slučaj iz nekoliko razloga: brzo se računa, kao što je vidljivo u izrazu 3.5, u aktivnom stanju propušta signal unaprijed i gradijent unazad te podržava propagiranje gradijenata s obzirom na ulazne i izlazne aktivacije.

$$\mathrm{ReLU}(x) = \max(0, x). \tag{3.5}$$



Slika 3.4. Graf zglobnice.

No i ona ima svoje nedostatke na koje treba obratiti pažnju. Prvo, nije definirana za $x = 0$, stoga pri implementaciji treba definirati da je gradijent u toj točci jednak gradijentu s lijeva (0) ili gradijentu s desna (1). Drugo, neuroni aktivirani ovom funkcijom mogu biti krhki u fazi treniranja i "umrijeti" ukoliko kroz njih prođe dovoljno veliki gradijent koji će težine ažurirati na takav način da ni za jedan naredni primjer ovaj neuron neće biti aktiviran, odnosno zauvijek će biti jednak nuli. Ovaj se problem može ublažiti pravilnim namještanjem stope učenja [5].

3.2.4. GELU

GELU (engl. *Gaussian Error Linear Unit*) predstavlja značajno unaprijeđenje u odnosu na svoje prethodnice u domeni aktivacijskih funkcija te je s vremenom postala standardna aktivacija za modele stanja tehnike temeljenih na arhitekturi Transformer, poput BERT-a [6] ili GPT-a [7]. Ukratko, ideja je iza ove funkcije da su njeni ulazi otežani po svojim vrijednostima, a ne propušteni ovisno o svom predznaku kao što je to slučaj kod zglobnice. Autori članka [8], kao njene glavne prednosti, navode njenu vjerojatnosnu interpretaciju te zakrivljenost popraćenu izostankom monotonosti, što ju čini

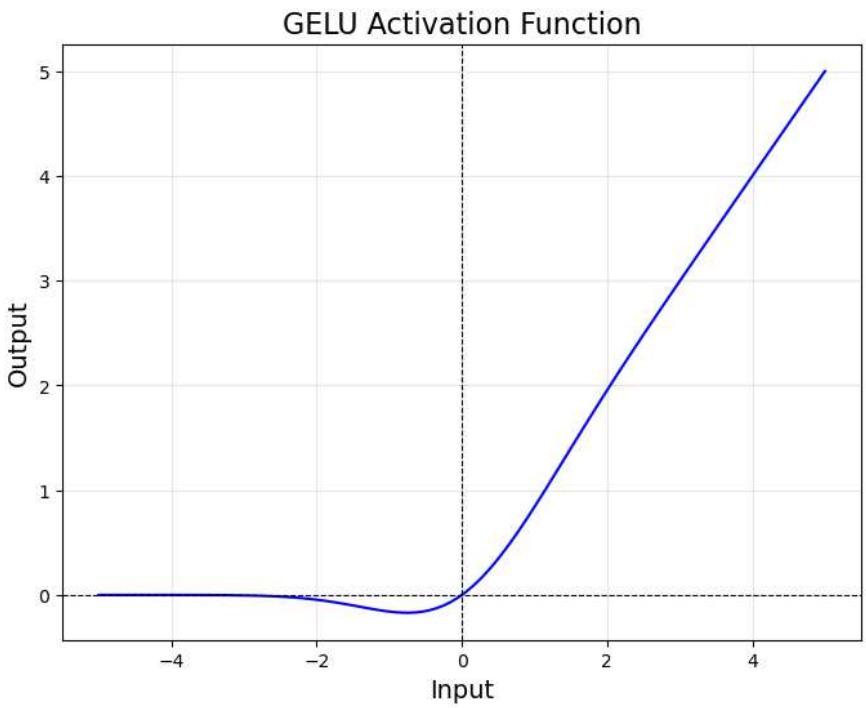
boljim aproksimatorom složenih funkcija nego što je to zglobnica. GELU se računa izrazom:

$$\mathrm{GELU}(x) = xP(X \leq x) = x\Phi(x) = x \cdot \frac{1}{2}\left[1 + \mathrm{erf}\left(\frac{x}{\sqrt{2}}\right)\right], \tag{3.6}$$

$$\mathrm{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt. \tag{3.7}$$

GELU možemo procijeniti pojednostavljenim izrazom 3.8 [8].

$$\mathrm{GELU}(x) \approx 0.5x\left(1 + \tanh\left[\sqrt{\frac{2}{\pi}}\left(x + 0.044715x^3\right)\right]\right). \tag{3.8}$$



Slika 3.5. Graf funkcije GELU.

3.3. Normalizacija nad grupom

Normalizacija nad grupom (engl. *batch norm*, BN) često je korištena tehnika u dubokom učenju, koja omogućuje stabilizaciju i ubrzavanje učenja neuronskih mreža smanjenjem fenomena poznatog kao unutrašnji kovarijantni pomak (engl. *internal covariate shift*).

Ovaj pojam odnosi se na sklonost da se distribucija aktivacija unutar mreže mijenja tijekom učenja, što može biti osobito izraženo u vrlo dubokim arhitekturama. Kako se parametri mreže ažuriraju propagacijom gradijenta unatrag, ulazi u svaki sloj se mijenjaju, što često zahtijeva pažljivu inicijalizaciju težina i korištenje niskih stopa učenja kako bi se osiguralo stabilno učenje. Normalizacija po grupi rješava ovaj problem tako što normalizira aktivacije unutar svake mini-grupe, osiguravajući da svaka aktivacija ima srednju vrijednost 0 i varijancu 1. Uzmimo da promatrani sloj ima d-dimenzionalni ulaz $\mathbf{x} = (x_1, ..., x_d)$. Procjenu srednje vrijednosti i varijance k-te značajke za mini-grupu $\mathcal{B} = \{\mathbf{x}^{(1)}, ..., \mathbf{x}^{(m)}\}$ računamo pomoću sljedećih izraza:

$$\mu_{\mathcal{B}} = \frac{1}{m}\sum_{i=1}^{m} x_k^{(i)}, \tag{3.9}$$

$$\sigma_{\mathcal{B}}^2 = \frac{1}{m}\sum_{i=1}^{m}(x_k^{(i)} - \mu_{\mathcal{B}})^2. \tag{3.10}$$

Sada aktivacije možemo normalizirati izrazom:

$$\hat{x}_k = \frac{x_k - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}, \tag{3.11}$$

gdje je ϵ konstanta koja osigurava numeričku stabilnost. Kako bi mreži omogućili da zadrži fleksibilnost učenja složenih značajki te spriječili da normalizacija ograniči model u izražavanju složenih odnosa u podacima, normalizirane aktivacije skaliramo te uvećamo slobodnim parametrima γ i β:

$$y_k = \gamma\hat{x}_k + \beta. \tag{3.12}$$

Tijekom faze zaključivanja želimo moći izračunati izlaz modela za samo jedan primjer te osigurati da izlaz deterministički ovisi o ulazu. Stoga u fazi zaključivanja prilikom izračuna procjenitelja koristimo statistike na razini cijeloga skupa podataka za učenje, odnosno čitave populacije. Alternativni pristup je korištenje pomičnog prosjeka (engl. *moving average*) očekivanja i varijance, koje računamo u fazi učenja. Jedna od glavnih prednosti normalizacije nad grupom je ta što omogućuje korištenje većih stopa učenja, čime se značajno ubrzava konvergencija modela. Također, BN smanjuje problem nestajućeg i eksplodirajućeg gradijenta, koji se često pojavljuje u dubokim mrežama s

aktivacijskim funkcijama poput sigmoide i tangensa hiperbolnog. Održavanjem aktivacija unutar stabilnog raspona, BN osigurava da gradijenti ostanu dovoljno veliki za učinkovitu optimizaciju. Još jedna značajna prednost BN-a je to što djeluje kao implicitna regularizacijska tehnika, često smanjujući potrebu za metodama poput Dropouta [9]. Uvođenjem varijabilnosti putem statistika na razini mini-grupe, BN unosi određenu dozu šuma u aktivacije, što može pomoći u smanjenju prevelike prilagodbe podacima za učenje i poboljšanju generalizacijske sposobnosti modela.

3.4. Konvolucijski modeli

Konvolucijske neuronske mreže (engl. *convolutional neural networks*, CNN) su vrsta modela dubokog učenja dizajnirana za obradu podataka s rešetkastom strukturom, što ih čini posebno učinkovitim za razne zadatke računalnog vida poput raspoznavanja slika i detekcije objekata. Ideja CNN-ova prvi je put ozbiljno predstavljena u radu [10], gdje je mreža LeNet-5 korištena za prepoznavanje ručno napisanih znamenki. CNN možemo definirati kao model, koji uz potpuno povezane slojeve, ima barem jedan konvolucijski sloj. Pogledajmo sljedeći primjer. Pretpostavimo da imamo crno-bijelu sliku $\mathbf{I}$, čije su vrijednosti piksela na poziciji (i, j) izražene funkcijom $f(i, j)$. Nadalje, neka je $w(k, l)$ vrijednost težine na poziciji (k, l) filtra, odnosno jezgre (engl. *kernel*) w. Također možemo pretpostaviti da je vrijednost obaju navedenih funkcija van njihove domene, odnosno okvira slike i jezgre, jednaka 0. Tada konvoluciju funkcija f i w možemo promatrati kao težinsku sumu piksela slike $\mathbf{I}$ unutar lokalnog susjedstva $\mathcal{N}$, pri čemu je $g(i, j)$ vrijednost piksela u rezultantnoj slici:

$$g(i, j) = f * w = \sum_{k,l} f(i - k, j - l)w(k, l). \tag{3.13}$$

Ipak, u dubokom se učenju pod pojmom konvolucije podrazumijeva primjena funkcije korelacije, koja rezultira reflektiranim signalom [5]:

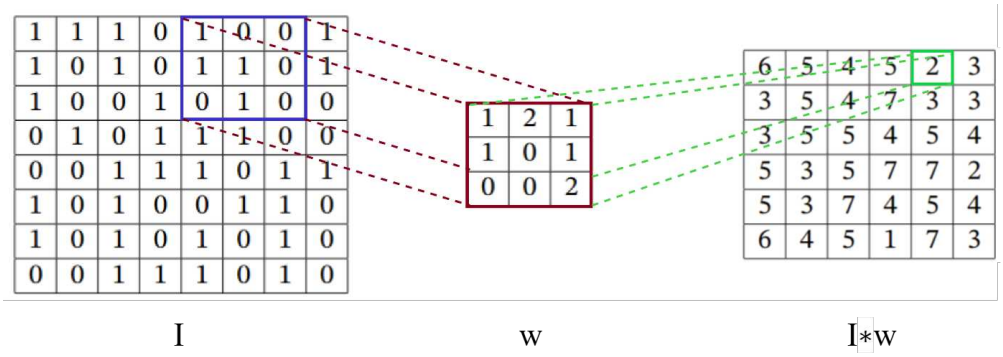
$$g(i, j) = \sum_{k,l} f(i + k, j + l)w(k, l). \tag{3.14}$$

Operacija korelacije prikazana je slikom 3.6., a ovakvog ćemo se nazivlja i mi pridržavati u ostatku ovoga rada. Uobičajeno u dubokom učenju raspolažemo sa slikama u RGB

formatu, tako da umjesto jednog, imamo tri ulazna sloja slike, odnosno kanala. S obzirom da je glavna prednost CNN-a mogućnost modeliranja lokalnih interakcija i njihovo ugrađivanje u diskriminativne i semantički bogate mape značajki (engl. *feature maps*), nad svakim kanalom prethodnog sloja, primjenjujemo više jezgri kanala narednog konvolucijskog sloja. Formalno, ako je C_h broj kanala u sloju h, a C_{h+1} broj kanala u sloju $h+1$, onda je vrijednost piksela na poziciji (i, j) u kanalu c_{h+1} sloja $h+1$ jednaka:

$$s(i, j, c_{h+1}) = \sum_{c_h \in \{C_h\}} \sum_{(k,l) \in \mathcal{N}} w(k, l, c_h, c_{h+1}) x(i+k, j+l, c_h) + b(c_{h+1}). \tag{3.15}$$

Pritom su $x(i, k, j)$ aktivacije prethodnog sloja, (k, l) pomaci unutar lokalnog susjedstva $\mathcal{N}$, $w(k, l, c_h, c_{h+1})$ jezgra c_{h+1}-tog kanala sloja $h+1$ koja se primjenjuje na c_h-ti kanal sloja h, $b(c_{h+1})$ vektor pomaka odgovarajućeg kanala sloja $h+1$, a $c_h \in [0, C_h)$. Tako su dobivene izlazne reprezentacije ulazne slike ekvivarijantne s obzirom na pomak [5]. Kako bi se očuvala izvorna rezolucija slika, njene rubove možemo nadopuniti nulama.



Slika 3.6. Konvolucija (korelacija) ulazne slike **I** s jezgrom **w**.

Neka su W_1 i H_1 prostorne dimenzije prethodnog sloja, P debljina nadopunjavanja na svakom bridu slike, a F prostorna dimenzija jezgre s korakom S. Tada prostorne dimenzije idućeg sloja dobijemo izrazima:

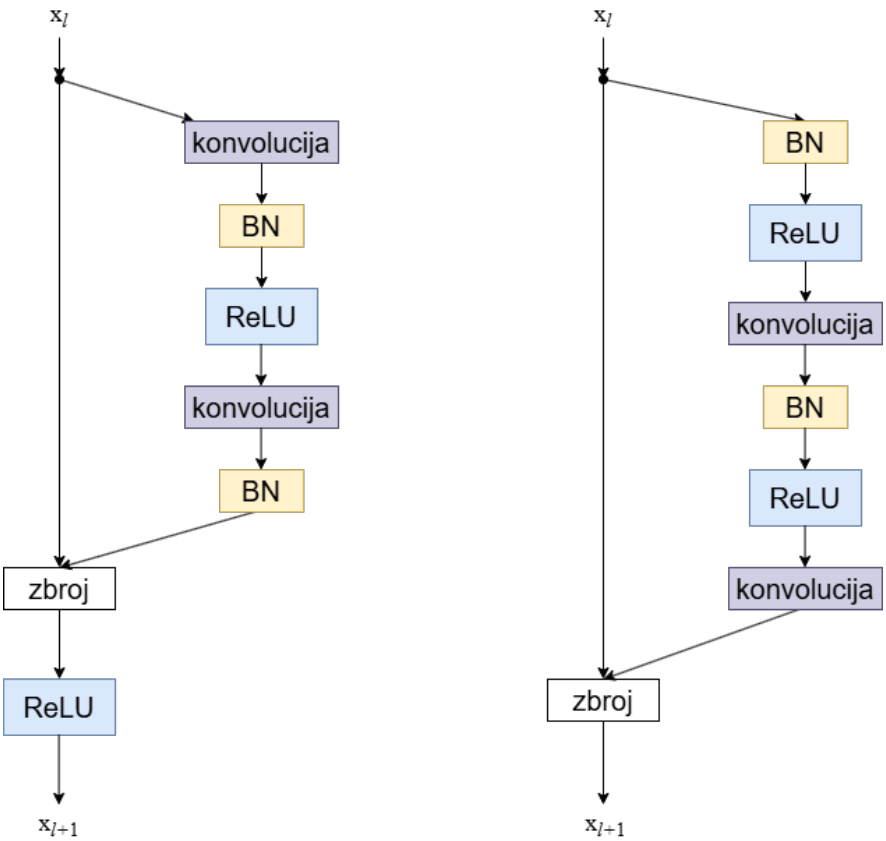
$$W_2 = (W_1 - F + 2P)/S + 1, \tag{3.16}$$

$$H_2 = (H_1 - F + 2P)/S + 1. \tag{3.17}$$

Osim potpuno povezanih i konvolucijskih slojeva, u dubokim konvolucijskim modelima u pravilu se koriste i slojevi sažimanja [5].

3.5. Rezidualne mreže

Rezidualne neuronske mreže (engl. *residual neural networks*, ResNet) predstavljene člankom [11], označile su značajan iskorak u dubokom učenju, rješavanjem problema degradacije točnosti koji se pojavljuje pri učenju vrlo dubokih neuronskih mreža. Prije ResNeta, povećavanje dubine konvolucijskih mreža često je dovodilo do smanjenih performansi, pri čemu dublji modeli nisu nužno postizali bolje rezultate od svojih plićih verzija. Neočekivano se pokazalo da degradacija nije bila uzrokovana prekomjernim prilagođavanjem modela podacima (engl. *overfitting*), već bi samim dodavanjem slojeva na modele već adekvatne dubine, padala točnost na skupu za učenje [11]. Rezidualne mreže su ovaj problem riješile uvođenjem rezidualnih veza, koje omogućuju mreži da uči rezidualnu funkciju umjesto izravnog modeliranja transformacija. Ideja u pozadini rezidualnih jedinica, bila je da duboki slojevi aditivno popravljaju približnu predikciju prethodnika.



Slika 3.7. Prikaz arhitektura izvorne rezidualne jedinice (lijevo) te njene poboljšane inačice (desno), predložene člankom [12]. Kao što se vidi, glavna razlika između njih je ta da poboljšana inačica provodi operacije normalizacije po grupi (BN) i aktivacije prije konvolucija, dok obrnuto vrijedi za originalnu inačicu. Slika je napravljena po uzoru na [12].

Neka je $\mathbf{x}_l$ ulaz, a $\mathbf{x}_{l+1}$ izlaz l-tog sloja dubokog modela. Nadalje, ako funkciju $\mathcal{F}$ de-

finiramo kao rezidualno preslikavanje ostvareno nizom uzastopnih transformacija ulaznog podatka, onda rezidualnu jedinicu možemo modelirati izrazima:

$$\mathbf{y}_l = h(\mathbf{x}_l) + \mathcal{F}(\mathbf{x}_l, W_l), \tag{3.18}$$

$$\mathbf{x}_{l+1} = f(\mathbf{y}_l). \tag{3.19}$$

Pritom je glavna ideja naučiti $\mathcal{F}$ s obzirom na funkciju identiteta $h(\mathbf{x}_l)$, što se može ostvariti korištenjem preskočne veze (engl. *skip connection*). Konačno, funkcija f je zglobnica. Ovako definirane rezidualne veze glade funkciju gubitka i poboljšavaju tok gradijenta. U članku [12] provedeno je istraživanje nad različitim konfiguracijama rezidualnih jedinica te su autori navedenog članka došli do zaključka da se promjenom funkcije f iz zglobnice u funkciju identiteta te provođenjem normalizacije nad grupom i aktivacije prije konvolucije unutar preslikavanja $\mathcal{F}$, dolazi do dodatnog poboljšanja performansi pri učenju dubokih modela. Arhitekture izvorne i poboljšane inačice rezidualne jedinice prikazane su slikom 3.7.

3.6. Modeli temeljeni na pažnji

3.6.1. Mehanizam pažnje

Arhitekture koder-dekoder temeljene na povratnim neuronskim mrežama (engl. *recurrent neural network*, RNN) [13, 14], netom nakon svoje pojave polučile su interes brojnih istraživača zbog svojih obećavajućih rezultata na zadacima strojnog prevođenja (engl. *machine translation*). No takve su arhitekture bile opterećene problemom degradirajućih performansi, proporcionalno s rastom duljine ulaznog niza. Razlog takvog ponašanja nalazio se u činjenici da bi neuronska mreža morala sažeti informaciju čitave izvorne rečenice u vektor fiksne duljine.

Pogledajmo sljedeći primjer. Koder iz koder-dekoder arhitekture u vektor c kodiranjem očita ulaznu rečenicu sastavljenu od T riječi, u obliku niza vektora $\mathbf{x} = (\mathbf{x_1}, ..., \mathbf{x_T})$. U tu se svrhu općenito koristi povratna neuronska mreža, čije se skriveno stanje $h_t \in R^n$ u trenutku t računa izrazom :

$$h_t = f(x_t, h_{t-1}), \tag{3.20}$$

gdje je f nelinearna funkcija. Nadalje, vektor c računa se primjenom također neke nelinearne funkcije q na niz vektora skrivenih stanja:

$$c = q(\{h_1, ..., h_T\}). \tag{3.21}$$

Nasuprot toga, za dani vektor konteksta c i sve prethodne riječi $\{y_1, ..., y_{t'-1}\}$, dekoder se uči s ciljem predviđanja sljedeće riječi $y_{t'}$. Rad dekodera možemo još promatrati kao maksimizaciju čitavog prijevoda $\mathbf{y}$ preko umnoška uvjetnih vjerojatnosti pojedinih riječi:

$$p(\mathbf{y}) = \prod_{t=1}^{T} p(y_t | \{y_1, \dots, y_{t-1}\}, c). \tag{3.22}$$

Uvjetna vjerojatnost svake riječi iz izraza 3.22 može se implementirati pomoću povratne ćelije:

$$p(y_t | \{y_1, \dots, y_{t-1}\}, c) = g(y_{t-1}, s_t, c), \tag{3.23}$$

pri čemu je g nelinearna funkcija, a s_t skriveno stanje dekodera u koraku t. Kao što je spomenuto u uvodnom dijelu ovoga odjeljka, a sada vidljivo i kroz izraz 3.23, vektor konteksta c raste s veličinom rečenice, što sa sobom nosi neželjene posljedice.

Kako bi premostili taj problem, autori članka [15] predložili su novu arhitekturu za strojno prevođenje, u kojoj dekoder vjerojatnosti sljedeću riječ u prijevodu računa pomoću izraza:

$$p(y_i | y_1, \dots, y_{i-1}, \mathbf{x}) = g(y_{i-1}, s_i, c_i). \tag{3.24}$$

s_i je skriveno stanje RNN-a u trenutku i, a računa se izrazom:

$$s_i = f(s_{i-1}, y_{i-1}, c_i). \tag{3.25}$$

Može se primijetiti da su ovdje, za razliku od izraza 3.24, vjerojatnosti y_i uvjetovane zasebnim vektorima konteksta c_i. Vektor konteksta c_i dobije se težinskom sumom skrivenih stanja kodera h_j:

$$c_i = \sum_{j=1}^{T_x} \alpha_{ij} h_j. \tag{3.26}$$

Vektor težine α_{ij} svakog stanja h_j računa se pomoću izraza:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}. \tag{3.27}$$

Pritom e_{ij} dobiven izrazom 3.28 predstavlja kvalitetu podudaranja ulaznog niza oko pozicije j te izlaznog niza oko pozicije i.

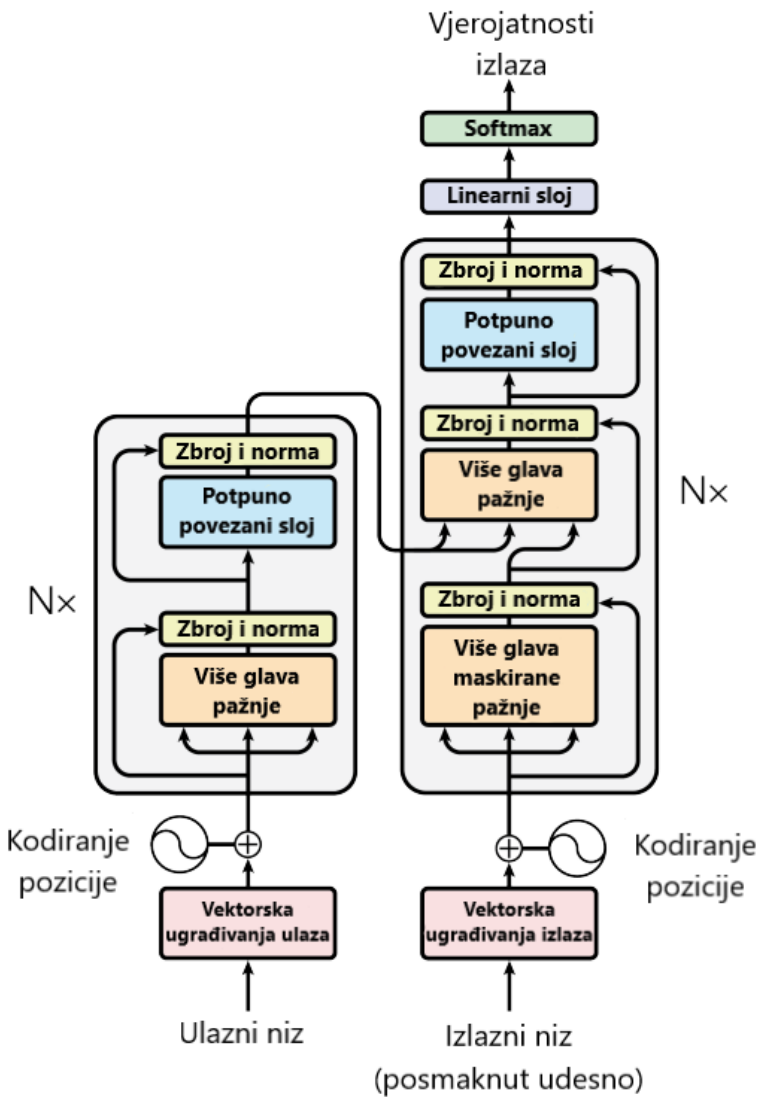
$$e_{ij} = a(s_{i-1}, h_j). \tag{3.28}$$

U prethodnom je izrazu funkcija a implementirana kao potpuno povezani sloj. Vjerojatnost α predstavlja važnost koja se pripisuje skrivenom stanju kodera h_j prilikom generiranja sljedeće riječi y_i. Ovime je načelno ostvaren mehanizam pažnje (engl. *attention mechanism*) u dekoderu, jer on sam bira dijelove izvorne rečenice na koje obraća pažnju.

3.6.2. Arhitektura Transformer

Unatoč implementaciji mehanizma pažnje u povratnim neuronskim mrežama, one su još uvijek imale suštinski problem ovisnosti računanja skrivenog stanja h_t u trenutku t, o prethodnom stanju h_{t-1}. Sekvencijalna priroda ovakvog pristupa onemogućava paralelno računanje skrivenih stanja, što postaje izraženije rastom ulaznog niza. Revolucionarno rješenje ovog problema, predstavljeno je u obliku arhitekture Transformer [16], koja se temelji isključivo na mehanizmu pažnje za računanje globalnih međuovisnosti ulaznog i izlaznog niza te u potpunosti otklanja potrebu za povratnim slojevima. Kao i njegovi prethodnici, model Transformer temeljen je na koder-dekoder arhitekturi. U nastavku ćemo pobliže proći kroz arhitekturu Transformer, ilustriranu slikom 3.8.

Koder ulazni niz $(x_1, ...x_n)$ preslikava u skup odgovarajućih reprezentacija $(h_1, ..., h_n)$. S druge strane, dekoder u svakom vremenskom koraku t autoregresivno generira novu riječ, koristeći kao dodatan ulaz prethodno generirane riječi do trenutka $t-1$. Koder arhitekture Transformer sastoji se od 6 identičnih slojeva, gdje je svaki od njih sačinjen od dvaju dodatnih podslojeva. Prvi podsloj čini više glava mehanizma pažnje (engl. *multi-head attention mechanism*), a drugi je podsloj potpuno povezana neuronska mreža modelirana dvama linearnim transformacijama odvojenih zglobnicom. Dodatno se provodi normiranje izlaza podslojeva (engl. *layer normalization*) [17] povezanih rezidualnim ve-



Slika 3.8. Ilustracija arhitekture Transformer, sastavljene od stoga kodera (lijevo) i stoga dekodera (desno). Koder se sastoji se od N slojeva, koji su sastavljeni od podslojeva više glava mehanizma pažnje, rezidualnih veza, potpuno povezanog sloja i normalizacije nad slojem [17]. S druge strane, dekoder ima dva podsloja više glava mehanizma pažnje, pri čemu je jedan maskiran, a drugi za parove ključ-vrijednost koristi izlaze iz slojeva kodera. Na ulaze oba stoga dovode se vektorska ugrađivanja riječi zbrojenih sa svojim pozicijskim ugrađivanjima. Ovo je modificirana inačica slike iz [16].

zama [11] s ulazima odgovarajućeg podsloja. Kako bi se omogućilo postojanje rezidualnih slojeva, svi podslojevi modela su dimenzije $d_{model} = 512$. Dekoder nad postojećim, uvodi još jedan dodatan podsloj koji vrši mehanizam pažnje pomoću više glava nad izlazima slojeva kodera. Prvi podsloj samopažnje izmijenjen je na način da on prilikom generiranja naredne riječi ne vidi riječi koje dolaze nakon nje. Ovakvim maskiranjem dijela mehanizma pažnje osigurano je da se riječ na i-toj poziciji uistinu predviđa samo na temelju riječi koje su joj prethodile u ulaznom nizu.

Mehanizam pažnje ostvaren je mapiranjem vektoriziranih upita (engl. *query*) Q i skupova parova ključ-vrijednost (engl. *key-value*) $K - V$, na izlaz. Pritom se izlaz računa težinskom sumom vrijednosti, gdje su težine pridijeljene svakoj vrijednosti izračunate funkcijom koja mjeri kompatibilnost između parova upita i ključeva. Mehanizam pažnje temeljen na skaliranom skalarnom umnošku (engl. *scaled dot-product attention*) sastoji se od vektora upita i ključeva dimenzije d_k te vektora vrijednosti dimenzije d_v. Radi mogućnosti paralelizacije računanja, u praksi se upiti, ključevi i vrijednosti pakiraju u odgovarajuće matrice Q, K i V. U konačnici, pažnja temeljena na skalarnom umnošku (slika 3.9.a) računa se izrazom:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V. \tag{3.29}$$

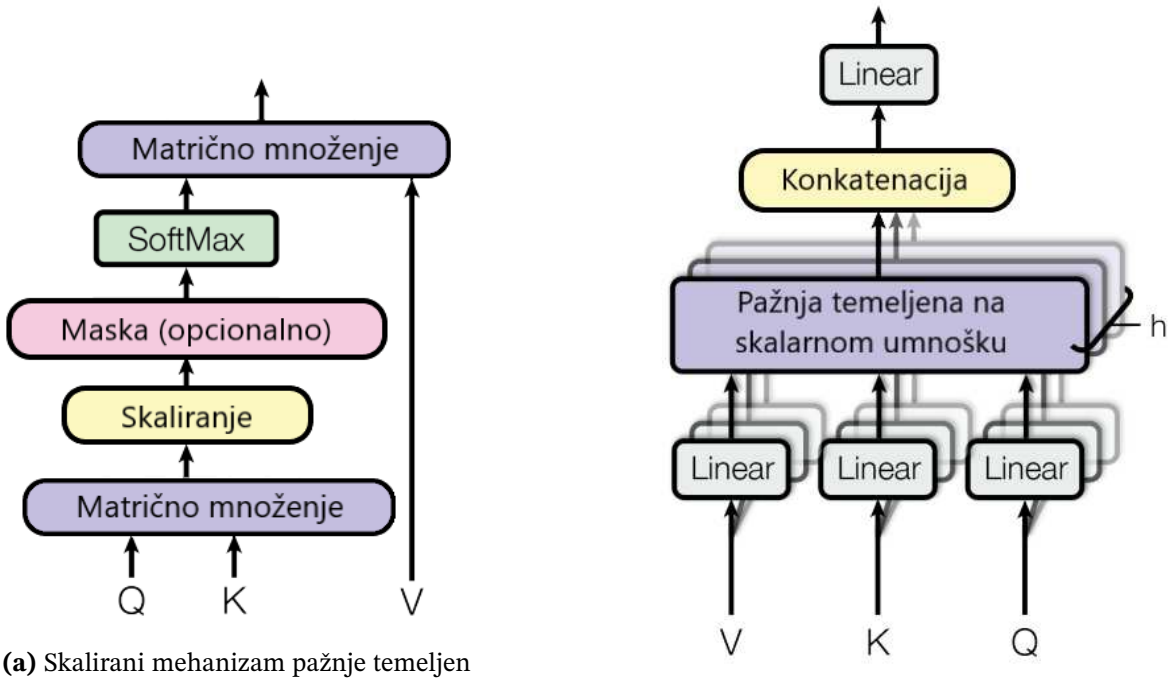
Autori članka [16] primijetili su bolje performanse zamjenom klasičnog mehanizma pažnje onim s više glava pažnje (engl. *multi-head attention*), što modelu omogućuje istovremeno praćenje više različitih informacija o ulaznom nizu. Umjesto korištenja jedne funkcije za izračun pažnje nad d_{model}-dimenzionalnim vektorima upita, ključeva i vrijednosti, h se puta naprave linearne projekcije upita, ključeva i vrijednosti u odgovarajuće vektore dimenzija d_k, d_k i d_v. Potom se paralelnim izračunom mehanizma pažnje nad tim vektorima, dobiju d_v-dimenzionalni vektori koji se u konačnici konkateniraju i ponovno linearno projiciraju, kao što je ilustrirano slikom 3.9.b. Linearne projekcije upita Q, ključeva K i vrijednosti V parametrizirane su matricama: $W_i^Q \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^K \in \mathbb{R}^{d_{\text{model}} \times d_k}, W_i^V \in \mathbb{R}^{d_{\text{model}} \times d_v}$ and $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$. Pažnja kroz više glava sada se računa izrazom:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O, \tag{3.30}$$

gdje je:

$$\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V). \tag{3.31}$$

Kako bi se očuvalo autoregresivno svojstvo slojeva samopažnje, nužno je spriječiti dotok informacije od riječi na poziciji i prema riječima koje se nalaze na pozicijama $< i$. Unutar slojeva skalirane pažnje temeljene na skalarnom umnošku, navedeno svojstvo implementira se maskiranjem, odnosno postavljanjem ovakvih nedozvoljenih veza na vrijednost $-\infty$, zbog čega nakon aktivacije softmaxom one poprimaju vrijednost 0. Model



(a) Skalirani mehanizam pažnje temeljen na skalarnom umnošku.

(b) Više glava mehanizma pažnje.

Slika 3.9. Prikaz skaliranog mehanizma pažnje (lijevo) i više glava pažnje (desno). Ovo je modificirana inačica slike iz [16].

Transformer zbog izostanka povratnih i konvolucijskih slojeva nema sposobnost praćenja informacije o poretku riječi u nizu. Rješenje ovog problema nalazi se u pozicijskom kodiranju (engl. *positional encoding*) riječi. Ugrađivanja pozicijski kodiranih riječi sumiraju se s ugrađivanjima ulaza prije propuštanja u slojeve kodera i dekodera. Autori članka [16], za pozicijsko su kodiranje koristili funkcije temeljene na sinusu i kosinusu:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right), \tag{3.32}$$

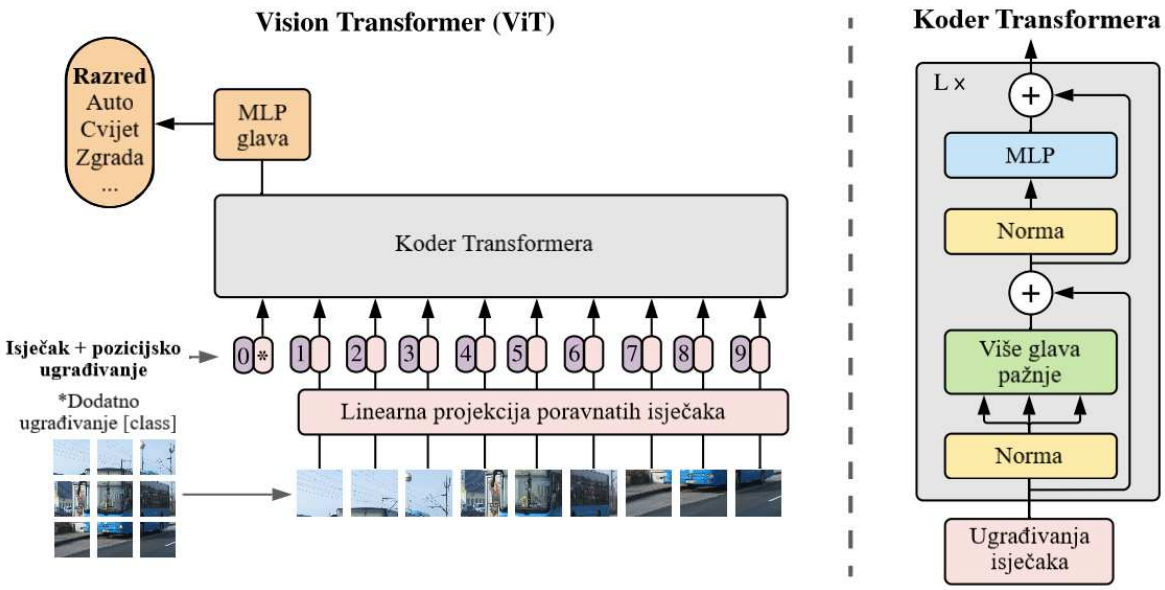
$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{\text{model}}}}\right). \tag{3.33}$$

Pritom je pos pozicija riječi unutar niza, a i dimenzija.

3.6.3. Arhitektura ViT

Netom po svojoj pojavi, arhitektura je Transformer postala neizostavnim dijelom modela stanja tehnike (enl. *state of the art*) za rješavanje problema obrade prirodnog jezika. Unatoč tome, njezina primjena je i dalje bila ograničena u području računalnog vida, gdje su konvolucijski slojevi još uvijek bili dominantna strategija za crpljenje vizualnih informacija iz slika. To se promijenilo pojavom arhitekture vizualnog transformera (engl. *Visual*

Transformer, ViT)[18], koja se primjenom uobičajenog modela Transformer na nizove slikovnih isječaka, odmaknula od paradigme oslanjanja na konvolucijske slojeve. Velika je prednost ovog pristupa što uz minimalne preinake postojećih arhitektura Transformer iz domene NLP-a, možemo iste koristiti za rješavanje problema računalnog vida. Model ViT je ilustriran slikom 3.10.



Slika 3.10. Prikaz arhitekture ViT. Izvorna slika podijeli se na isječke, čijim ugrađivanjima dodamo pozicijska ugrađivanja. Na početak niza vektora ugrađivanja dodamo [class] ugrađivanje koje će kasnije sadržavati globalnu reprezentaciju svih slikovnih isječaka. Takav niz predamo koderu arhitekture Transformer. Ovo je modificirana inačica slike iz [18].

Kako bi Transformer mogao obraditi dvodimenzionalne slike, izvornu sliku $\mathbf{x} \in \mathbb{R}^{H \times W \times C}$ pretvorimo u niz poravnatih dvodimenzionalnih isječaka (engl. *flattened 2D patches*) $\mathbf{x}_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$. Pritom je $H \times W$ rezolucija izvorne slike, a C broj njenih kanala. $P \times P$ je rezolucija isječaka, a $N \times (P^2 \cdot C)$ ukupan broj isječaka, koji ujedno predstavlja i duljinu ulaznog niza Transformera. Dodatno, na početak se niza ugrađivanja isječaka ubaci [class] ugrađivanje (engl. *embedding*), koje služi kao globalna reprezentacija čitave slike pomoću koje se donosi odluka u klasifikacijskim zadacima. Kao i kod arhitekture Transformer, ovdje se također koriste jednodimenzionalna pozicijska ugrađivanja kako bi se očuvala informacija o položaju isječaka unutar slike.

4. Kontrastno predučenje uz nadzor iz prirodnog jezika

4.1. CLIP

Dugo je vremena učenje uz fiksan, prethodno poznat broj razreda bila pretpostavljena paradigma razvoja klasifikacijskih modela računalnog vida. No ovakav oblik nadzora ograničava općenitost takvih modela te zahtijeva dodatne označene podatke ukoliko se žele reprezentirati ikakvi drugi vizualni koncepti. Kako bi premostili taj problem, autori su članka [19] razvili model CLIP (engl. Contrastive Language-Image Pre-training), koji se uči pomoću nadziranja prirodnim jezikom (engl. natural language supervision) na velikim, nestrukturiranim skupovima podataka koji sadrže slike i njihove popratne jezične opise. Time je omogućeno učenje generaliziranih reprezentacija koje se mogu primijeniti na širok spektar zadataka, bez potrebe za dodatnim ugađanjem modela (engl. fine-tuning). Osim toga, motivacija je bila i razvoj modela koji može bolje razumjeti kontekstualne veze između jezika i slika, čime bi se unaprijedile mogućnosti modela u stvarnim primjenama poput pretraživanja slika, generiranja opisa i semantičkog razumijevanja.

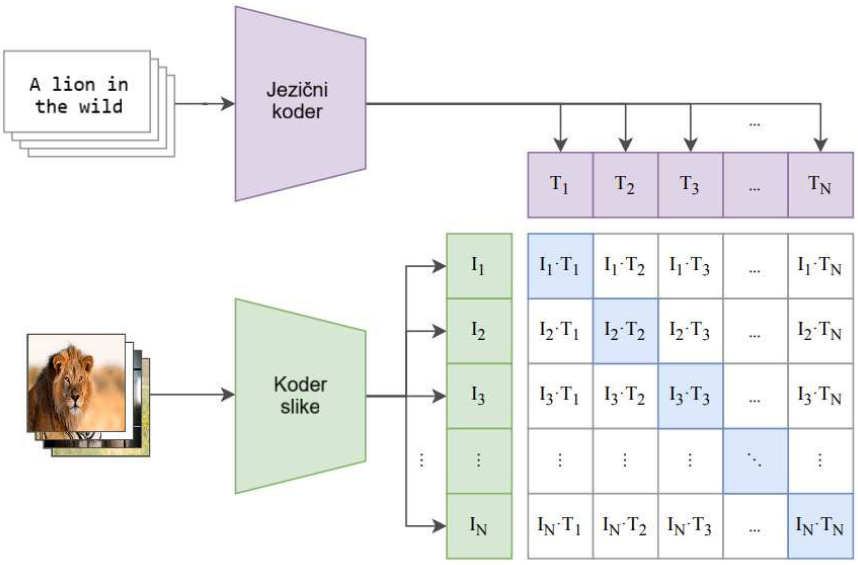
Preduvjet razvoju modela CLIP, bilo je konstruiranje odgovarajućeg skupa podataka. Autori su članka [19] u tu svrhu putem slobodno dostupnih izvora na internetu prikupili 400 milijuna parova slika i njihovih jezičnih opisa. Predviđanje točnih riječi za svaku od slika zahtjevan je zadatak zbog širokog raspona opisa, komentara i srodnih tekstova koji se pojavljuju uz te slike. Stoga su autori članka [19] relaksirali taj problem kontrastnim predučenjem kodera slike temeljenom na arhitekturi ViT te jezičnog kodera temeljenog na arhitekturi Transformer, nad parovima slika i čitavih jezičnih opisa.

Za mini-grupu od N parova slika i pripadajućih jezičnih opisa, CLIP se uči kako bi predvidio koja su se od $N \times N$ mogućih uparivanja zaista dogodila. Kako bi u tome uspio,

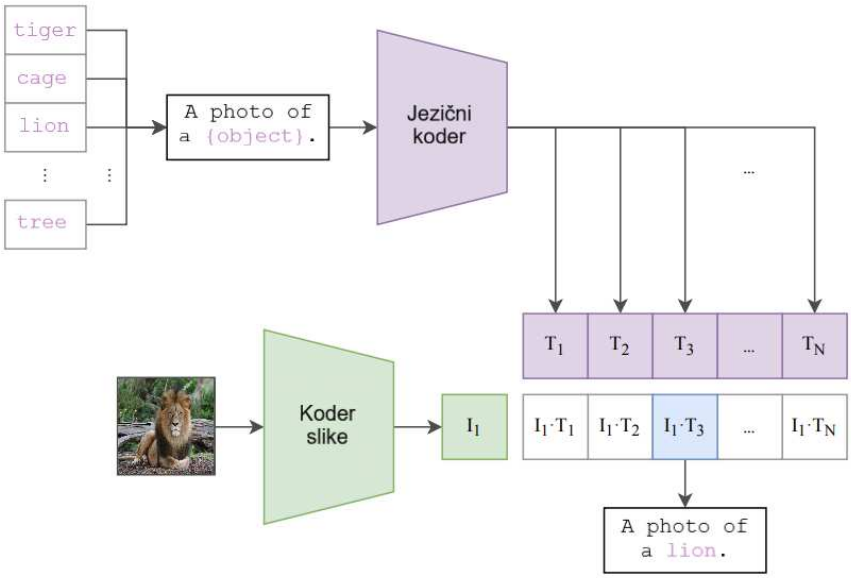
CLIP uči multimodalni prostor ugrađivanja tako što istovremeno podučava koder slike i jezični koder da maksimiziraju kosinusnu sličnost N izvornih odgovarajućih parova unutar mini-grupe te minimiziraju kosinusnu sličnost ugrađivanja $N^2 - N$ nepodudarnih uparivanja. Naposljetku se nad takvim ocjenama podudarnosti optimizira simetrični gubitak unakrsne entropije. Pseudokod ovog pristupa prikazan je slikom 4.1.

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t             - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss   = (loss_i + loss_t)/2
```

Slika 4.1. Ključni dio faze kontrastnog predučenja prikazan u obliku pseudokoda nalik Numpyju. Propuštanjem mini-grupa slika i pripadajućih jezičnih opisa kroz njihove odgovarajuće kodere, dobijemo slikovne i jezične reprezentacije koje se potom projiciraju u prostor ugrađivanja te normiraju. Nakon toga se izračunaju kosinusne sličnosti između svih parova slikovnih i jezičnih ugrađivanja. Nad tako dobivenom matricom sličnosti izračuna se simetrični gubitak unakrsne entropije. Slika je preuzeta od [19].



Slika 4.2. Faza kontrastnog predučenja. Mini-grupu slika propustimo kroz koder slike, a mini-grupu jezičnih opisa kroz jezični koder. Potom izračunamo kosinusne sličnosti za sve moguće parove slika i jezičnih opisa unutar mini-grupe veličine N. U konačnici optimiziramo simetrični gubitak unakrsne entropije, maksimizirajući skalarne umnoške normiranih ugrađivanja na glavnoj dijagonali matrice sličnosti, istovremeno minimizirajući ostale elemente van dijagonale. Ovo je modificirana inačica slike iz [19].



Slika 4.3. Faza zaključivanja klasifikatora. Iz ciljanih imena razreda stvorimo jezične opise nalik onima korištenim u fazi predučenja, koje potom predamo jezičnom koderu. Množenje tako dobivenih jezičnih ugrađivanja sa slikovnim ugrađivanjem promatranog primjera, rezultira vektorom kosinusnih sličnosti. Konačna odluka klasifikatora je element vektora u kojem je kosinusna sličnost između slikovnog i jezičnog ugrađivanja najveća. Ovo je modificirana inačica slike iz [19]

4.2. SigLIP

Pojavom i naknadnim uspjehom CLIP-a, predučenje temeljeno na slabom nadzoru pomoću parova slika i jezičnih opisa nađenih na internetu, postala je sve učestalija strategija za stjecanje općenitih okosnica (engl. backbone) modela računalnog vida, pritom istiskujući iz uporabe predučenje na velikim skupovima podataka s višerazrednim oznakama. Uobičajen način predučenja takvih modela oslanja se na optimizaciju kontrastnog cilja između parova slika i jezičnih opisa. Kako je opisano prethodnim odjeljkom 4.1., kontrastno predučenje podrazumijeva poravnavanje valjanih (pozitivnih) slikovnih i jezičnih parova u prostoru ugrađivanja (engl. *embedding space*) te međusobno udaljavanje pogrešnih (negativnih) parova. To se postiže dvostrukom primjenom softmax gubitka nad čitavom mini-grupom, jednom za normalizaciju kosinusnih sličnosti preko svih slika te drugi puta za normalizaciju preko svih jezičnih opisa. Kako bi se izbjegla brojčana nestabilnost naivne implementacije softmaxa, od svakog člana promatranog vektora oduzima se njegov element najvećeg iznosa prije primjene samoga softmaxa [20], što zahtijeva još dva dodatna prolaza po čitavoj mini-grupi. Kako bi doskočili tom problemu, autori članka [21] predložili su jednostavniju alternativu u vidu sigmoidnog gubitka, koji ne zahtijeva provođenje računskih operacija preko cijelih mini-grupa te stoga značajno pojednostavljuje implementaciju distribuiranog gubitka i povećava efikasnost. Također konceptualno odvaja veličinu mini-grupe od definicije samoga zadatka.

U nastavku ćemo objasniti predučenje parova slika i jezičnih opisa sigmoidnim gubitkom (engl. *Sigmoid Loss for Languag-Image Pretraining*, SigLIP) te kako se ono razlikuje od klasičnog predučenja pomoću softmax gubitka. Za mini-grupu parova slika i njihovih jezičnih opisa $\beta = \{(I_1, T_1), (I_2, T_2), ..., (I_N, T_N)\}$, kontrastnim učenjem potičemo poravnavanje podudarnih parova (I_i, T_i) u prostoru ugrađivanja te udaljavanje ugrađivanja različitih parova $(I_i, T_{j \neq i})$. To se pomoću softmax gubitka formalizira na način da se koder slike $f(\cdot)$ i jezični koder $g(\cdot)$ uče s ciljem minimizacije izraza:

$$-\frac{1}{2|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\left(\overbrace{\log\frac{e^{t\mathbf{x}_i\cdot\mathbf{y}_i}}{\sum_{j=1}^{|\mathcal{B}|}e^{t\mathbf{x}_i\cdot\mathbf{y}_j}}}^{\text{slika}\rightarrow\text{jezični opis softmax}} + \overbrace{\log\frac{e^{t\mathbf{x}_i\cdot\mathbf{y}_i}}{\sum_{j=1}^{|\mathcal{B}|}e^{t\mathbf{x}_j\cdot\mathbf{y}_i}}}^{\text{jezični opis}\rightarrow\text{slika softmax}}\right). \tag{4.1}$$

Pritom su x_i i y_i jednaki izrazima:

$$\mathbf{x}_i = \frac{f(I_i)}{\|f(I_i)\|_2}, \tag{4.2}$$

$$\mathbf{y}_i = \frac{g(T_i)}{\|g(T_i)\|_2}. \tag{4.3}$$

Koder slike $f(\cdot)$ ostvaren je arhitekturom ViT, a jezični koder $g(\cdot)$ arhitekturom Transformer. Skalar t parametriziran je izrazom $\exp(t')$ pri čemu je t' globalni parametar koji se također uči. Umjesto globalne normalizacije po mini-grupi kao izrazu 4.1, sigmoidni gubitak nezavisno obrađuje svaki par slike i jezičnog opisa. Time se problem učenja pretvara u uobičajenu binarnu klasifikaciju na skupu podataka svih mogućih parova slika i jezičnih opisa, pritom dodjeljujući pozitivne oznake pravim parovima (I_i, T_i) te negativne oznake nepodudarnim parovima $(I_i, T_{j\neq i})$. Sigmoidni gubitak definiran je na sljedeći način:

$$-\frac{1}{|\mathcal{B}|}\sum_{i=1}^{|\mathcal{B}|}\sum_{j=1}^{|\mathcal{B}|}\log\underbrace{\frac{1}{1+e^{z_{ij}(-t\mathbf{x}_i\cdot\mathbf{y}_j+b)}}}_{\mathcal{L}_{ij}}, \tag{4.4}$$

gdje oznaka z_{ij} za promatrani par slike i jezičnog opisa iznosi 1 ako se oni podudaraju ili -1 ako se ne podudaraju. Autori su članka [21] dodali još jedan parametar, b, koji smanjuje početnu dominaciju negativnih članova prilikom računanja funkcije gubitka te posljedično sprječava velike optimizacijske korake u početku učenja modela. Pseudokod algoritma predučenja sigmoidnim gubitkom prikazan je na slici 4.4.

```
# img_emb       : image model embedding [n, dim]
# txt_emb       : text model embedding [n, dim]
# t_prime, b    : learnable temperature and bias
# n             : mini-batch size

t = exp(t_prime)
zimg = l2_normalize(img_emb)
ztxt = l2_normalize(txt_emb)
logits = dot(zimg, ztxt.T) * t + b
labels = 2 * eye(n) - ones(n) # -1 with diagonal 1
l = -sum(log_sigmoid(labels * logits)) / n
```

Slika 4.4. Pseudokod predučenja pomoću sigmoidnog gubitka. Prvo eksponiramo temperaturni parametar. Nakon toga normaliziramo slikovna i jezična ugrađivanja unutar mini-grupe. Zatim skalarno pomnožimo normalizirana ugrađivanja kako bi dobili kosinusne sličnosti između slikovnih i jezičnih ugrađivanja. Konačno, primjenom logaritmirane sigmoide na umnožak oznaka i kosinusnih sličnosti dobijemo gubitak promatrane mini-grupe. Slika je preuzeta od [21].

5. Samonadzirano učenje vizualnih transformera

5.1. DINO

Nedugo nakon svoje pojave, vizualni transformeri (ViT) upareni s predučenjem na velikim skupovima podataka i ugađanjem (engl. *fine-tuning*) na ciljanom skupu podataka, u domeni raspoznavanja slika postali su prikladna alternativa konvolucijskim neuronskim mrežama. Unatoč tome, vizualni transformeri nisu pokazali jasne prednosti nad konvolucijskim modelima, od kojih su računski zahtjevniji, zahtijevaju više podataka i njihove značajke ne prikazuju unikatna svojstva. Potaknuti uspjehom samonadziranog predučenja transformera u domeni obrade prirodnog jezika (engl. *natural language processing*), autori članka [3] htjeli su ispitati bi li samonadzirano predučenje pokazalo slično poboljšanje performansi kod vizualnih transformera. Pri obradi prirodnog jezika, samonadzirane strategije predučenja koriste riječi unutar rečenice za stvaranje pomoćnih zadataka (engl. *pretext tasks*), koji pružaju bolji signal za učenje u usporedbi s nadziranim učenjem, koje se temelji na predviđanju jedne oznake po rečenici. Analogno tome, nadzor na razini čitave slike sklon je reduciranju bogate vizualne informacije prisutne u slici na jedan koncept, odabran iz unaprijed definiranog skupa razreda. Prilikom ispitivanja samonadzirnog predučenja vizualnih transformera, autori članka [3] došli su do sljedećih ključnih spoznaja:

- značajke dobivene samonadziranim modelima ViT-a sadrže eksplicitnu informaciju o rasporedu scena i granicama objekata, što se može izravno očitati iz slojeva samopažnje posljednjeg bloka.

- Pojava segmentacijskih maski u značajkama model ViT pokazala se kao uobičajena osobina samonadziranih metoda, no dobre performanse k-NN klasifikatora jav-

ljaju se samo ukoliko se istovremeno koriste određene metode poput momentnog kodera [22] te isječaka različitih rezolucija (engl. *multi-crop augmentation*) [23].

- Slikovni isječci (engl. *patches*) manjih dimenzija rezultiraju kvalitetnijim značajkama.

Na temelju ovih otkrića, autori članka [3] predložili su jednostavan samonadzirani pristup nazvan DINO, koji se može tumačiti kao oblik destilacije znanja bez oznaka (engl. *distillation with no labels*). DINO pojednostavljuje samonadzirano predučenje, izravnim predviđanjem izlaza mreže učitelja (momentni koder), pritom koristeći standardan gubitak unakrsne entropije. Ovaj pristup djeluje čak i ako se samo izlazi učiteljske mreže izoštre i centriraju kako bi se izbjegla trivijalna rješenja u vidu kolapsa moda (engl. *mode collapse*), bez potrebe za naprednim normalizacijama ili kontrastnim gubitkom. U nastavku ćemo objasniti princip samonadziranog učenja uz pomoć destilacije bez oznaka.

Destilacija znanja (engl. *knowledge distillation*) paradigma je učenja kojom se mreža studenta g_{θ_s} parametrizirana s θ_s, uči tako da se njeni izlazi podudaraju s izlazima mreže učitelja g_{θ_t}, parametrizirane s θ_t. Za ulaz x, obje mreže daju K-dimenzionalni vektore razdiobe vjerojatnosti P_s i P_t. Pritom K označava dimenziju izlazne reprezentacije svake mreže. Vjerojatnost P_s dobijemo normalizacijom izlaza mreže g softmaxom:

$$P_s(x)^{(i)} = \frac{\exp\left(g_{\theta_s}(x)^{(i)}/\tau_s\right)}{\sum_{k=1}^{K} \exp\left(g_{\theta_s}(x)^{(k)}/\tau_s\right)}. \tag{5.1}$$

Pritom je $\tau_s > 0$ temperaturni parametar kojim kontroliramo oštrinu distribucije. Na identičan način računa se i P_t. Fiksiranjem mreže učitelja g_{θ_t}, učimo podudaranje distribucija P_s i P_t, minimizacijom gubitka unakrsne entropije s obzirom na parametre mreže studenta θ_s:

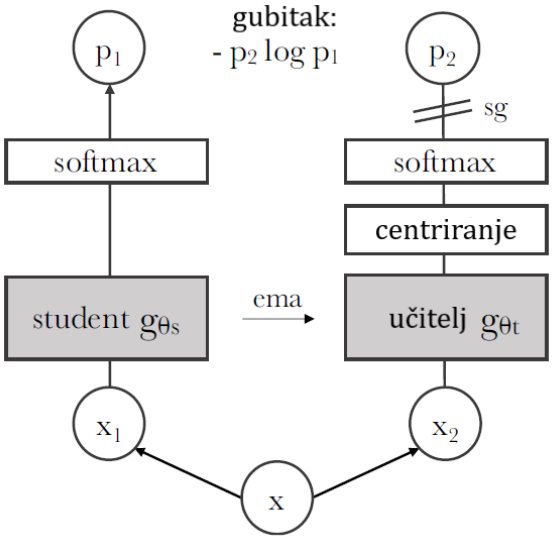
$$\min_{\theta_s} H(P_t(x), P_s(x)), \tag{5.2}$$

pri čemu je $\mathrm{H}(a, b) = -a\log b$. Kako bi navedeni optimizacijski problem prilagodili paradigmi samonadziranog učenja, moramo izraditi različite poglede na sliku (engl. *views*), odnosno njene različite isječke. Iz dane slike x konstruiramo skup različitih pogleda V. Ovaj skup sadrži dva globalna pogleda, x_1^g i x_2^g te nekoliko lokalnih pogleda manje rezolu-

cije. Svi su pogledi iz skupa propuštaju kroz mrežu studenta, dok mreža učitelja dobiva samo globalne poglede. Time se postiže stvaranje korespondencije između lokalnih i globalnih pogleda. Sada minimiziramo sljedeći gubitak:

$$\min_{\theta_s} \sum_{x \in \{x_1^g, x_2^g\}} \sum_{\substack{x' \in V \\ x' \neq x}} H(P_t(x), P_s(x')). \tag{5.3}$$

Ovakav oblik gubitka može se koristiti nad dvama ili više pogleda. Autori članka [3] odlučili su se koristiti standardnu postavku isječaka različitih rezolucija, koja se sastoji od dva globalna isječka rezolucije 224×224 piksela, koji prekrivaju barem 50% površine izvorne slike te više lokalnih isječaka rezolucije 96×96 piksela koji prekrivaju manje od 50% površine izvorne slike. Mreže studenta i učitelja dijele istu arhitekturu g, pri čemu svaka ima vlastite parametre θ_s i θ_t. Parametri mreže studenta θ_s uče se minimizacijom izraza 5.3 pomoću stohastičkog gradijentnog spusta. Postupak DINO, ilustriran je slikom 5.1.



Slika 5.1. Samodestilacija bez oznaka. Slikom je prikazan postupak DINO za jedan par pogleda (engl. *views*) na izvornu sliku. Model prvo vrši dvije različite transformacije ulazne slike te ih predaje mrežama studenta i učitelja. Obje mreže imaju jednaku arhitekturu, ali različite parametre. Izlaz mreže učitelja centrira se parametrom centra koji se računa eksponencijalnim pomičnim prosjekom (engl. *exponential moving average*) preko mini-grupe izlaza mreže učitelja. Svaka mreža daje K-dimenzionalan vektor značajki koji je normaliziran temperaturnim softmaxom preko dimenzije značajki. Potom se pomoću gubitka unakrsne entropije izračuna njihova sličnost. Zatim se na mrežu učitelja primjeni operator zaustavljanja gradijenta (engl. *stop-gradient*, sg) kako bi se gradijenti propagirali isključivo kroz mrežu studenta. Parametri mreže učitelja osvježe se eksponencijalnim pomičnim prosjekom parametara mreže studenta. Ovo je modificirana inačica slike iz [3].

Mreža učitelja nema apriorno znanje pomoću kojeg bi se učila, stoga u tu svrhu koristi prethodne iteracije mreže studenta, primjenom eksponencijalnog pomičnog prosjeka (engl. *exponential moving average*) nad težinama mreže studenta. Kako bi prilikom učenja izbjegli situaciju u kojoj model za različite ulaze uvijek rezultira istim izlazima, odnosno kolaps moda (engl. *mode collapse*) , autori članka [3] otkrili su da je dovoljno napraviti centriranje te izoštravanje izlaza momentnog kodera mreže učitelja. Centriranje sprječava dominaciju jedne dimenzije vektora značajki tako što sve njegove vrijednosti približava k uniformnoj razdiobi, što opet može imati neželjenu posljedicu kolapsa moda k uniformnoj razdiobi. S druge strane, izoštravanje ima suprotan efekt kroz stvaranje izlazne razdiobe koja ima naglašeniji vrh. Stoga se zajedničkom primjenom ovih operacija nad izlazima mreže učitelja balansiraju njihovi utjecaji te sprječava kolaps moda. Izoštravanje izlaza postiže se korištenjem male vrijednosti temperaturnog parametra τ_t prilikom normalizacije mreže učitelja softmaxom. Centriranje se u suštini može interpretirati kao dodavanje vektora pristranosti c izlazima mreže učitelja: $g_t(x) \leftarrow g_t(x) + c$. Vrijednost centra c osvježava se eksponencijalnim pomičnim prosjekom pomoću izraza 5.4, gdje m parametar stope momentnog kodera, a B veličina mini-grupe.

$$c \leftarrow mc + (1 - m)\frac{1}{B}\sum_{i=1}^{B} g_{\theta_t}(x_i). \tag{5.4}$$

Pseudokod ovako opisanog algoritma samonadzirane destilacije znanja prikazan je slikom 5.2.

```
# gs, gt: student and teacher networks
# C: center (K)
# tps, tpt: student and teacher temperatures
# l, m: network and center momentum rates
gt.params = gs.params
for x in loader: # load a minibatch x with n samples
    x1, x2 = augment(x), augment(x) # random views

    s1, s2 = gs(x1), gs(x2) # student output n-by-K
    t1, t2 = gt(x1), gt(x2) # teacher output n-by-K

    loss = H(t1, s2)/2 + H(t2, s1)/2
    loss.backward() # back-propagate

    # student, teacher and center updates
    update(gs) # SGD
    gt.params = l*gt.params + (1-l)*gs.params
    C = m*C + (1-m)*cat([t1, t2]).mean(dim=0)

def H(t, s):
    t = t.detach() # stop gradient
    s = softmax(s / tps, dim=1)
    t = softmax((t - C) / tpt, dim=1) # center + sharpen
    return - (t * log(s)).sum(dim=1).mean()
```

Slika 5.2. Algoritam DINO bez korištenja više isječaka različitih rezolucija (engl. *multi-crop*), prikazan u obliku pseudokoda nalik PyTorchu. Na početku su parametri mreže učitelja i mreže studenta jednaki. Proces učenja započinje primjenom dvaju različitih transformacija na sliku x, kako bi dobili dva pogleda (engl. *views*) na tu sliku. Te poglede, odnosno isječke predamo koderima mreže studenta i mreže učitelja. Zatim centriramo te izoštrimo vektore značajki dobivenih mrežama studenta i učitelja te izračunamo gubitak. U konačnici se još osvježi parametar centriranja c. Slika je preuzeta od [3].

5.2. DINOv2

Predučenje na velikim skupovima pokazalo je obećavajuće rezultate u domeni obrade prirodnog jezika. Ponukani tom činjenicom, autori su članka [24] odlučili sličan princip primijeniti za učenje modela računalnog vida, s ciljem stvaranja fundacijskih modela (engl. *fundation models*) koji bi proizvodili značajke za općenitu uporabu (engl. *general purpose features*), odnosno značajke koje bi bolje radile na zadacima van-distribucijske generalizacije bez potrebe za dodatnim ugađanjem. Iako se DINO pokazao kao dobar ekstraktor značajki, temeljio se na malim okosnicama ViT-S s 21 milijuna (21M) parametara i ViT-B s 85M parametara. Također, učen je na relativno ograničenom skupu podataka ImageNet [25]. Stoga su poboljšanjem postupka samonadziranog učenja te skaliranjem veličina okosnica ViT-a i skupa podataka, autori članka [24] stvorili obitelj modela nazvanu DINOv2. U nastavku ćemo ukratko proći kroz osnovne ideje kojima DINOv2 unaprjeđuje svog prethodnika DINO.

Iz perspektive podataka, autori članka [24] osmislili su automatizirani cjevovod za izgradnju uređenog, raznovrsnog i specijaliziranog skupa podataka za samonadzirano učenje vizualnih značajki. Polazišnu točku u tom cjevovodu čine dvije vrste podataka. Prva su vrsta uređeni skupovi podataka poput ImageNeta [25] i Google Landmarksa [26]. Drugi skup sačinjen je od nefiltriranih slika iz javno dostupnih repozitorija podataka prikupljenih s weba (engl. *crawled web data*). Kako bi se izbjegla redundancija i povećala raznovrsnost slika, na neuređenom skupu izvršen je proces deduplikacije pomoću cjevovoda za detekciju kopija. Uređeni skup za predučenje izgrađen je odabirom slika iz neuređenog skupa, na temelju bliskosti sa slikama iz uređenog skupa. Bliskost se računala kosinusnom sličnošću slikovnih ugrađivanja slika iz dvaju skupova, dobivenih ViT-H/16 okosnicom predučenom na ImageNet-22k skupu podataka. Nakon izračuna sličnosti, za svaku je sliku upita (engl. *query image*) pomoću algoritma k-sredina (engl. *k-means*) izabrano N (uobičajeno 4) najbližih susjeda.

Sa strane implementacije i postupka predučenja, DINOv2 između ostalog uvodi sljedeće ključne izmjene:

- nasumično se maskira dio ulaznih slikovnih isječaka predanih mreži studenta, ali ne i mreži učitelja. Potom se primjeni iBOT[27] glava mreže studenta nad ma-

skiranim slikovnim isječcima. iBOT glava se primjeni i nad vidljivim isječcima mreže učitelja koji odgovaraju skrivenim isječcima mreže studenta. Tako dobiveni vektori značajki se normaliziraju softmaxom i centriraju. Konačno iBOT gubitak izračunamo sljedećim izrazom:

$$\mathcal{L}_{iBOT} = -\sum_i p_{ti} \log p_{si}, \tag{5.5}$$

pri čemu je i indeks isječka maskiranih tokena. Slično kao i kod klasičnog DINO gubitka, učimo parametre mreže studenta, a glava mreže učitelja uči se eksponencijalnim pomičnim prosjekom parametara mreže studenta.

- Korištenje odvojenih parametara za DINO te iBOT projekcijske glave.
- Primjena Sinkhorn-Knopp centriranja nad mrežom učitelja, kao što je prikazano na slici 5.3. [28].

```
# x is n-by-K
# tau is Sinkhorn regularization param
x = exp(x / tau)
for _ in range(num_iters): # 1 iter of Sinkhorn
  # total weight per dimension (or cluster)
  c = sum(x, dim=0, keepdim=True)
  x /= c

  # total weight per sample
  n = sum(x, dim=1, keepdim=True)
  # x sums to 1 for each sample (assignment)
  x /= n
```

Slika 5.3. Algoritam za centriranje Sinkhorn-Knopp, prikazan kodom nalik na PyTorch. Slika je preuzeta od [3].

- Primjena KoLeo regularizatora koji potiče uniformnost značajki unutar mini-grupe. Za skup od n vektora $(x_1, \dots, x_n)$, ovaj regularizator definiran je izrazom:

$$\mathcal{L}_{koleo} = -\frac{1}{n}\sum_{i=1}^{n} \log(d_{n,i}), \tag{5.6}$$

pri čemu je $d_{n,i} = \min_{j \neq i} ||x_i - x_j||$ minimalna udaljenost između vektora x_i te bilo koje druge točke u mini-grupi. Valja napomenuti da se prije računanja ovog regularizacijskog izraza provede ℓ_2 normalizacija značajki.

- Krajem faze predučenja rezolucija slika se poveća na 518×518 piksela kako bi se

sprječilo nestajanje malih objekata na niskim rezolucijama.

Umjesto zasebnog učenja manjih modela, oni su destilirani iz najvećeg DINOv2 modela, temeljenog na ViT-g okosnici s jednom milijardom (1B) parametara. Pritom je korištena izmijenjena inačica postupka destiliranja znanja opisanog u ovom poglavlju, gdje je ViT-g korišten kao zamrznuti učitelj, a manji modeli kao mreže studenta. Autori su članka [24] ovakvim pristupom postigli bolje performanse nego zasebnim učenjem manjih modela od početka.

6. Skupovi podataka

6.1. TACO

TACO (engl. *Trash Annotations in Context*) javno je dostupan skup podataka za raspoznavanje i segmentaciju slika otpada [29]. TACO se sastoji od 4784 anotacija sadržanih unutar 1500 slika visoke rezolucije, većinom fotografiranih mobilnim uređajima. Anotacije slika sadrže oznake pozadine (engl. *scene tags*) koje predstavljaju vrstu okoline na slikama kao što su vegetacija, voda, pločnik itd. Same instance otpada segmentirane su i označene hijerarhijskom taksonomijom koja se sastoji od 60 razreda, svrstanih u 28 nadrazreda, uključujući posebnu kategoriju "Unlabeled litter", koja predstavlja objekte koji nisu pokriveni ostalim razredima ili se ne mogu jednoznačno označiti. Sve instance mogu se svrstati u samo jedan razred pod nazivom "Litter", što je najveća razlika u odnosu na neke druge skupove podataka poput COCO-a [30], kod kojih je međusobna isključivost razreda ključna.Primjeri instanci skupa podataka TACO prikazani su na slici 6.1. U nastavku ćemo poglavlja prikazati dvije konfiguracije ovog skupa, nad kojima će biti provedeni eksperimenti u 8. poglavlju.

6.1.1. TACO-10

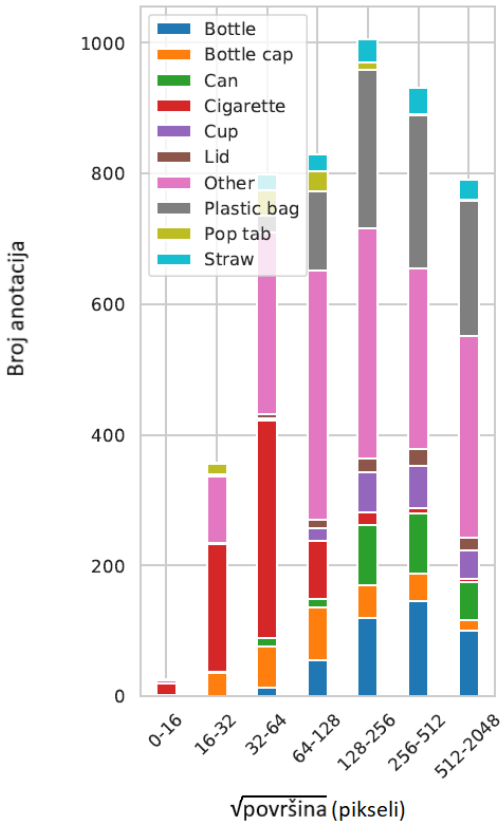
Ova je konfiguracija osmišljena od strane autora članka [29]. Originalni razredi preslikani su u devet nadrazreda odabranih na osnovu svoje brojnosti unutar skupa te dodatnog desetog razreda po imenu "Other", u koji su smješteni primjeri svih preostalih razreda. Ovakva postavka oznaka mogla bi se koristiti u nekom stvarnom sustavu za razvrstavanje otpada, u kojem bi bilo bitnije od kojeg je materijala otpad sačinjen, nego kojeg je on oblika te izvorne namijene. Veličine opisujućih kvadrata (engl. *bounding box*) instanci razreda prikazane su slikom 6.2.

Slika 6.1. Primjer nasumično odabranih instanci iz testnog podskupa skupa podataka TACO.

6.1.2. TACO-56

Kako bismo ispitali mogućnost naših modela da raspoznaju slike na skupu gdje je nekada teško odvojiti primjere različitih razreda ili kada je broj primjeraka pojedinih razreda izrazito malen, eksperimente ćemo također provesti na originalnoj podjeli po razredima. No kako bismo pomoću metode "train_test_split", kako je implementirana u biblioteci scikit-learn[1], mogli napraviti podjelu na podskupove za učenje, validaciju i testiranje, izbacit ćemo razrede sa svega par primjera, što će rezultirati skupom koji se sastoji od 56 oznaka razreda.

[1] `https://scikit-learn.org/stable/`



Slika 6.2. Stupčasti dijagram broja anotacija pojedinog razreda po veličini opisujućih regija (engl. *bounding box*) instanci razreda skupa podataka TACO-10. Ovo je preinačena verzija slike iz [29].

7. Implementacija

Modeli koje ćemo razmatrati u nastavku rada, implementirani su pomoću programskog jezika Python i njegove biblioteke otvorenog koda (engl. *open-source library*) za automatsku diferencijaciju te rukovanje tenzorima - PyTorch[1]. PyTorch je razvijen od strane istraživačkog tima MetaAI te omogućuje jednostavno definiranje, treniranje i testiranje neuronskih mreža. Automatska diferencijacija omogućena je modulom Autograd, što olakšava računanje gradijenata i optimizaciju modela. PyTorch se često koristi u istraživačkim i industrijskim projektima zbog svoje fleksibilnosti, jednostavnosti te snažne podrške za ubrzanje pomoću grafičkih procesorskih jedinica odnosno GPU-a (engl. *graphical processing unit*), što ga čini jednim od najpopularnijih alata u području umjetne inteligencije.

S obzirom na to da je skup podataka na kojem je naučen izvorni CLIP [19] privatan, autori su članka [31] za učenje vlastite implementacije modela CLIP (nazvane OpenCLIP), iskoristili javno dostupne skupove podataka LAION-400M[32] i LAION-5B[33]. Izvorni programski kod modela, upute za korištenje i rezultati vrednovanja nalaze se u repozitoriju[2] OpenCLIP-a, kojeg ćemo i mi koristiti za naše eksperimente. Konkretno, korištena je okosnica ViT-B/16 prednaučena na skupu podataka datacomp_xl_s13b_b90k. Za klasifikaciju bez ugađanja pomoću SigLIP-a, koristili smo model google/siglip-base-patch16, koji se može učitati putem koda dostupnog u repozitoriju[3]. Za ekstrakciju vizualnih značajki pomoću modela DINOv2, upotrijebili smo ViT-B/14[4] okosnicu destiliranu iz ViT-g/14 okosnice, predučene na skupu podataka LVD-142M[24]. Svi su eksperimenti provedeni preko platforme Google Colab[5], koristeći GPU T4 sa 16 GB GDDR6 memorije.

[1] https://PyTorch.org/
[2] https://github.com/mlfoundations/open_clip
[3] https://github.com/huggingface/transformers/blob/main/docs/source/en/model_doc/siglip.md
[4] https://github.com/facebookresearch/dinov2?tab=readme-ov-file
[5] https://colab.google/

Sposobnost raspoznavanja slika pomoću prednaučenih multimodalnih i samonadziranih modela ispitali smo kroz dvije glavne postavke: raspoznavanje, odnosno klasifikacija slika na neviđenim razredima (engl. *zero-shot classification*) te klasifikacija linearnom projekcijom (engl. *linear probe*). Dodatno, uspješnost linearne projekcije uspoređena je kroz više različitih konfiguracija:

1. Značajke dobivene koderom slike OpenCLIP-a ili DINOv2 značajke projicirane na jedan klasifikacijski potpuno povezani sloj, čija je dimenzija jednaka broju razreda u skupu podataka.

2. Značajke dobivene koderom slike OpenCLIP-a ili DINOv2 značajke projicirane na skriveni potpuno povezani sloj dimenzije 512, aktiviran funkcijom GELU te ponovno projiciran na klasifikacijski potpuno povezani sloj, čija je dimenzija jednaka broju razreda u skupu podataka.

3. Ansambliranje modela konkateniranjem slikovnih značajki dobivenih OpenCLIP-om sa slikovnim značajkama dobivenih modelom DINOv2 te njihovim naknadnim propuštanjem kroz klasifikacijsku glavu (engl. *classification head*) koja se sastoji od jednog potpuno povezanog sloja ili dva potpuno povezana sloja odvojena funkcijom GELU.

4. Ansambliranje modela aritmetičkom sredinom pojedinih elemenata vektora logita iz zasebno naučenih klasifikacijskih glava modela OpenCLIP i DINOv2.

Svi eksperimenti linearnih projekcija provedeni su smrzavanjem okosnica modela OpenCLIP i DINOv2 te učenjem isključivo klasifikacijskih glava. Učenje svih modela provodilo se nad skupom za učenje kroz 20 epoha uz implementirano rano zaustavljanje ukoliko se gubitak na validacijskom skupu nije smanjio nakon 5 uzastopnih epoha. Veličina mini-grupe iznosila je 32. Za učenje klasifikacijske glave OpenCLIP-a korištena je stopa učenja $1e^{-3}$, a za DINOv2 $1e^{-4}$. Za raspoređivač stope učenja (engl. *learning rate scheduler*) odabrano je kosinusno kaljenje (engl. *cosine annealing*) s minimalnom vrijednošću parametra stope učenja postavljenom na $1e^{-6}$ [34].

8. Eksperimenti

U ovom poglavlju predstavit ćemo rezultate provedenih eksperimenata. Odjeljkom 8.1. predstavljeni su rezultati klasifikacije slika bez dodatnog ugađanja, dok su u odjeljku 8.2. prikazani rezultati klasifikacije slika pomoću linearnih projekcija.

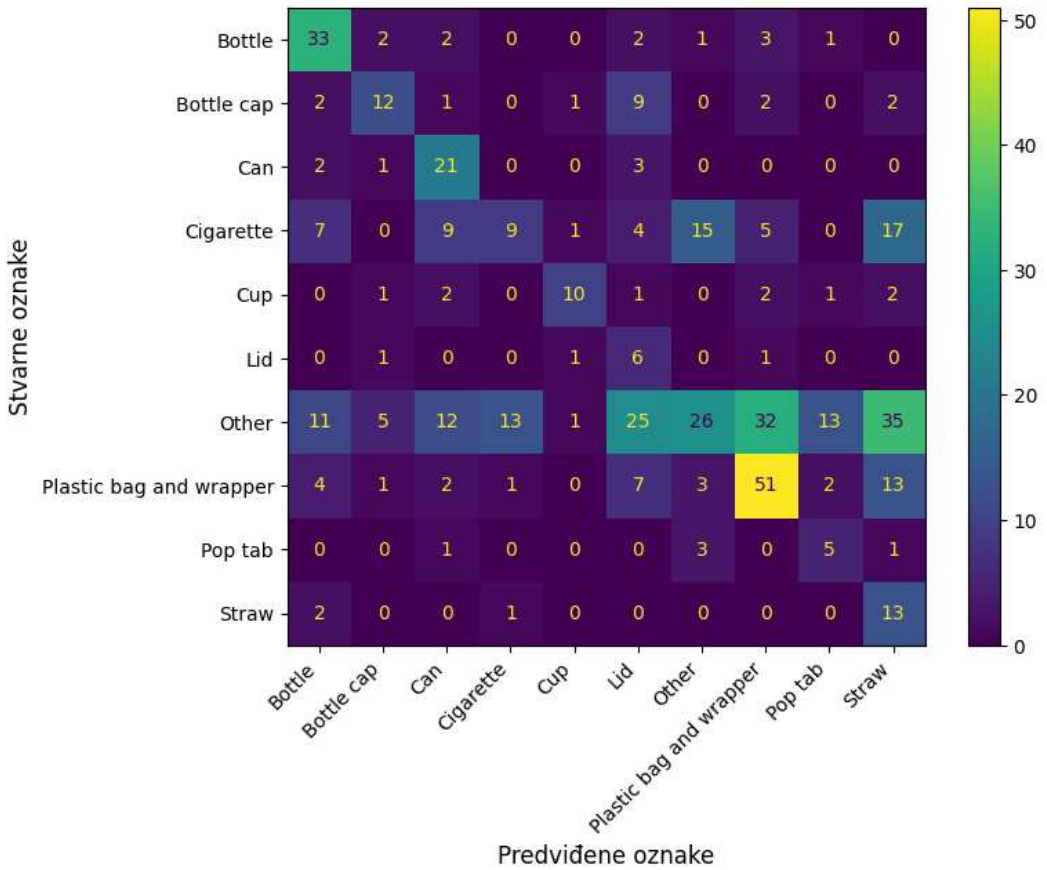
8.1. Klasifikacija bez ugađanja

Jedan od ključnih ciljeva ovoga rada bio je ispitati kvalitetu raspoznavanja slika na neviđenim razredima bez ugađanja (engl. *zero-shot*) pomoću kontrastno prednaučenog multimodalnog modela CLIP. U tablici 8.1. vidimo top-1, top-2 i top-3 točnosti klasifikacije pomoću modela OpenCLIP, koji koristi standardni softmax gubitak te SigLIP koji koristi sigmoidni gubitak. Na oba skupa podataka, SigLIP nadmašuje top-1 točnost OpenCLIP-a, s izraženijom razlikom na skupu TACO-10. No, u slučaju top-2 i top-3 točnosti, OpenCLIP na oba skupa podataka pokazuje bolju točnost nego SigLIP. Iz ovih rezultata možemo iščitati da SigLIP donosi bolji zaključak kada dani primjer treba označiti jednim, najvjerojatnijim razredom. Za razliku od toga, OpenCLIP općenito daje veću vjerojatnost semantički bližim razredima, no nije toliko siguran u jednu, najbolju predikciju.

Skup podataka		TACO-10			TACO-56		
Metoda	Okosnica	Top-1	Top-2	Top-3	Top-1	Top-2	Top-3
OpenCLIP	ViT-B/16	35.36	**48.95**	**57.74**	22.38	**36.82**	**46.44**
SigLIP	ViT-B/16	**38.91**	48.54	53.77	**23.22**	35.98	43.31

Tablica 8.1. Rezultati klasifikacije slika bez ugađanja (engl. *zero-shot*) pomoću metoda OpenCLIP i SigLIP sa ViT-B/16 okosnicama. Prikazane su top-1, top-2 i top-3 točnosti klasificiranja nad skupovima podataka TACO-10 i TACO-56.

Također vidimo da razmatrani modeli pokazuju značajno lošije performanse na skupu podataka TACO-56. Ovo je vjerojatno posljedica većeg broja razreda skupa TACO-56,



Slika 8.1. Matrica zabune koja prikazuje ishod klasifikacije nad skupom podataka TACO-10, koristeći model SigLIP bez ugađanja (engl. *zero-shot*).

gdje bi i čovjeku zbog semantičke bliskosti nekih razreda bilo teško donijeti odluku o ispravnoj oznaci. Nadalje, slikom 8.1. prikazana je matrica zabune (engl. *confusion matrix*) top-1 točnosti klasifikacije modela SigLIP na skupu podataka TACO-10. Iz nje je vidljivo da model najviše točnosti gubi na primjerima razreda "Cigarette", "Other" te "Plastic bag and wrapper". Sličan ishod imali su i autori članka [29]. Manjak pouzdanosti prilikom raspoznavanja primjera razreda "Cigarette" može se objasniti malom izvornom rezolucijom njegovih slika, kao što je vidljivo na slici 6.2. Rasipanje točnosti na razredu "Other" vrlo je vjerojatno posljedica njegovog širokog semantičkog opsega jer je u njega preslikano više od 40 originalnih oznaka razreda. Iz navedenog razloga je modelima teško poravnati slikovna i jezična ugrađivanja navedenog razreda. Iako se primjeri razreda "Plastic bag and wrapper" vrlo dobro klasificiraju, dio gubitka točnosti na njima moguća je posljedica loše kvalitete i velikog broja oblika u kojem se ti objekti mogu pronaći.

8.2. Klasifikacija linearnom projekcijom

Kao što smo vidjeli u prethodnom odjeljku, raspoznavanje slika na neviđenim razredima pomoću modela OpenCLIP i SigLIP pokazuje općenito loše performanse. Stoga ćemo u nastavku ovoga odjeljka vidjeti kako se modeli OpenCLIP i DINOv2 ponašaju ukoliko nad njihovim ugrađivanjima slikovnih značajki izvršimo linearnu projekciju (engl. *linear probe*) pomoću klasifikacijske glave sačinjene od jednog (LP) ili dva potpuno povezana sloja odvojena aktivacijskom funkcijom GELU (MLP).

Pogledom na tablicu 8.2. prvo možemo primijetiti da model koji koristi DINOv2 za ekstrakciju vizualnih značajki, na oba skupa podataka puno bolje klasificira primjere nego model temeljen na OpenCLIP-u, što je najizraženije u slučaju top-1 točnosti, koja se u korist DINOv2-a razlikuje za otprilike 7% (LP) i 10% (MLP). Iz toga možemo zaključiti da DINOv2 stvara bolje vizualne značajke za klasifikaciju nego OpenCLIP. Također, vidimo da klasifikacija pomoću složene klasifikacijske glave postiže u pravilu veću točnost od samo jednog sloja linearne projekcije, što upućuje na to da značajke dobivene iz obiju okosnica nisu u potpunosti linearno odvojive. Na temelju tih spoznaja možemo zaključiti, što je i potkrijepljeno rezultatima u tablici 8.2., da najbolju mogućnost raspoznavanja slika ima model DINOv2 sa složenom klasifikacijskom glavom.

Skup podataka		TACO-10			TACO-56		
Metoda	Okosnica	Top-1	Top-2	Top-3	Top-1	Top-2	Top-3
OpenCLIP LP	ViT-B/16	70.50	90.59	**96.65**	58.37	73.85	81.38
OpenCLIP MLP	ViT-B/16	71.76	91.21	96.03	58.37	74.69	81.80
DINOv2 LP	ViT-B/14	77.41	91.00	95.61	65.48	77.41	**85.15**
DINOv2 MLP	ViT-B/14	**80.54**	**92.89**	95.61	**67.78**	**79.29**	**85.15**

Tablica 8.2. Rezultati klasifikacije slika linearnom projekcijom (engl. *linear probe*) nad metodama OpenCLIP s okosnicom ViT-B/16 i DINOv2 s okosnicom ViT-B/14. Kratica "LP" označava linearnu projekciju s jednim potpuno povezanim slojem u klasifikacijskoj glavi, dok "MLP" označava klasifikacijsku glavu s dva potpuno povezana sloja odvojena aktivacijskom funkcijom GELU. Prikazane su top-1, top-2 i top-3 točnosti klasificiranja nad skupovima podataka TACO-10 i TACO-56.

Dodatno smo proveli eksperimente kojima smo htjeli provjeriti pomaže li ansambliranje modela DINOv2 i OpenCLIP pri raspoznavanju slika. Rezultati ovih eksperimenata prikazani su tablicom 8.3. Najbolju je top-1 točnost na oba skupa podataka postigao model kojem su konkatenirane vizualne značajke dobivene modelima DINOv2 i Open-

CLIP te potom propuštene kroz MLP klasifikacijsku glavu, dok je najbolju top-2 i top-3 točnost pokazao model jednostavne aritmetičke sredine logita zasebno naučenih modela DINOv2 MLP i OpenCLIP MLP. Usporedbom tablica 8.2. i 8.3., vidimo da kombiniranje OpenCLIP-a i DINOv2-a kao ekstraktora značajki, vodi k poboljšanju performansi što upućuje na to da ove okosnice iz primjera izvlače komplementarne značajke. Također možemo vidjeti da dodatan potpuno povezani sloj, odvojen aktivacijskom funkcijom povećava točnost modela i time opet potvrđuje da značajke dobivene pojedinačnim okosnicama modela nisu linearno odvojive. Kao što je to bio slučaj i u prethodnim eksperimentima, tako i prilikom ansambliranja modela, točnost na skupu podataka TACO-56 znatno je lošija nego točnost na skupu TACO-10 navodeći nas na zaključak da sama kompleksnost semantičkog odvajanja razreda skupa TACO-56 predstavlja problem za modele DINOv2 i OpenCLIP.

Skup podataka	TACO-10			TACO-56		
Model	Top-1	Top-2	Top-3	Top-1	Top-2	Top-3
DINOv2 + OpenCLIP (concat) LP	77.41	92.23	97.07	66.53	79.29	85.77
DINOv2 + OpenCLIP (concat) MLP	**80.75**	93.31	97.70	**67.78**	80.33	85.56
DINOv2 + OpenCLIP logits LP	79.08	93.31	97.07	66.53	80.33	86.19
DINOv2 + OpenCLIP logits MLP	79.29	**95.40**	**97.49**	67.36	**81.38**	**87.45**

Tablica 8.3. Rezultati klasifikacije slika linearnom projekcijom (engl. *linear probe*) nad mješavinom modela OpenCLIP i DINOv2. Za OpenCLIP je korištena okosnica ViT-B/16, a za DINOv2 okosnica ViT-B/14. "DINOv2+CLIP(concat)" predstavlja model kod kojeg se ugrađivanja slikovnih značajki (engl. *image feature embeddings*) modela DINOv2 konkateniraju na ugrađivanja slikovnih značajki modela OpenCLIP te naknadno povezuju s klasifikacijskom glavom. "DINOv2 + OpenCLIP logits" predstavlja aritmetičke sredine po elementima vektora logita zasebno naučenih klasifikacijskih glava modela OpenCLIP i DINOv2. Kratica "LP" označava linearnu projekciju s jednim potpuno povezanim slojem u klasifikacijskoj glavi, dok kratica "MLP" označava klasifikacijsku glavu koja sadrži dva potpuno povezana sloja odvojena aktivacijskom funkcijom GELU. Prikazane su top-1,top-2 i top-3 točnosti klasificiranja nad skupovima podataka TACO-10 i TACO-56.

9. Zaključak

Cilj ovog rada bio je istražiti te ispitati suvremene metode raspoznavanja slika, s posebnim naglaskom na samonadzirane i multimodalne modele dubokog učenja. Kako bi u tome uspjeli, u 3. poglavlju analizirali smo duboke modele koji se koriste u računalnom vidu, s dodatnim fokusom na modele temeljene na mehanizmu pažnje, odnosno arhitekture Transformer i Visual Transformer (ViT). U narednom poglavlju, detaljno smo opisali postupak kontrastnog predučenja uz nadzor iz prirodnog jezika, pomoću multimodalnog modela CLIP. Poglavljem 5. je kroz model DINO predstavljena samonadzirana paradigma učenja, koja umjesto eksplicitnih oznaka, za nadzor koristi pseudooznake generirane postupkom destilacije znanja iz samih podataka.

Kroz proces implementacije odabran je otvoren skup podataka TACO te su oblikovani podskupovi za učenje, validaciju i testiranje, kako bi osigurali dosljednu evaluaciju modela. Primjenom prednaučenih modela OpenCLIP, SigLIP i DINOv2, nad dvjema konfiguracijama TACO-a, proveli smo eksperimente koji su omogućili detaljnu analizu njihove generalizacijske sposobnosti. Prvo smo ispitali sposobnost prednaučenih modela OpenCLIP i SigLIP da klasificiraju slike bez dodatnog ugađanja svojih okosnica. SigLIP je pokazao veću sigurnost prilikom predviđanja jedne, najvjerojatnije oznake razreda, dok je standardan OpenCLIP pokazao bolju top-2 i top-3 točnost, što znači da je pridijelio veću vjerojatnost razredima koji su semantički bliži ciljanom razredu. No, oba su pristupa pokazala relativno slabe performanse na neviđenim razredima, ne prešavši ni prag top-1 točnosti od 39%. Nadalje, također smo ispitali utjecaj dodavanja klasifikacijske glave u obliku jednoslojne ili dvoslojne linearne projekcije, na prednaučene okosnice modela OpenCLIP i DINOv2. Dodavanjem klasifikacijske glave na predučeni OpenCLIP model ostvarili smo skoro duplo bolju točnost nego što je to bio slučaj prilikom klasifikacije nad neviđenim razredima. Eksperimentalni rezultati upućuju na to da DINOv2

izvlači semantički bogatije značajke nego OpenCLIP. Dodavanjem nelinearne aktivacije i još jednog potpuno povezanog sloja u klasifikacijsku glavu, također se povećala točnost klasifikacije, što upućuje na činjenicu da značajke koje navedeni modeli generiraju nisu u potpunosti linearno odvojive. Općenito najbolje performanse postigli smo konkatenacijom značajki iz okosnica modela OpenCLIP i DINOv2, time pokazujući da oni generiraju donekle komplementarne značajke. Valja naglasti da su u svim eksperimentalnim postavkama, performanse osjetno lošije na skupu podataka TACO-56, koji sadrži desetke više razreda od skupa TACO-10. Stoga smo zaključili da navedene okosnice u ovakvim postavkama imaju poteškoće pri klasifikaciji skupova podataka s većim brojem semantički bliskih razreda. Uočili smo i uzorak u rasipanju točnosti na pojedinim razredima skupa podataka TACO.

Početni pravac za budući rad bio bi ispitati performanse modela korištenih u ovom radu, na nekom većem skupu podataka poput ImageNeta. Također bi valjalo provjeriti bismo li korištenjem većih okosnica ViT-L i ViT-G uspjeli poboljšati sposobnost raspoznavanja slika na skupu podataka TACO.

Literatura

[1] J. Šnajder, *Strojno učenje 1*, 2022., pog. Osnovni koncepti v2.2.

[2] ——, *Strojno učenje 1*, 2022., pog. Grupiranje v2.2.

[3] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, i A. Joulin, "Emerging properties in self-supervised vision transformers", u *2021 IEEE/CVF International Conference on Computer Vision, ICCV 2021, Montreal, QC, Canada, October 10-17, 2021*. IEEE, 2021., str. 9630–9640. https://doi.org/10.1109/ICCV48922.2021.00951

[4] J. Šnajder, *Strojno učenje 1*, 2022., pog. Logistička regresija v2.3.

[5] R. Szeliski, *Computer Vision: Algorithms and Applications 2nd Edition*. Springer, 2022.

[6] J. Devlin, M. Chang, K. Lee, i K. Toutanova, "BERT: pre-training of deep bidirectional transformers for language understanding", u *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, J. Burstein, C. Doran, i T. Solorio, Ur. Association for Computational Linguistics, 2019., str. 4171–4186. https://doi.org/10.18653/V1/N19-1423

[7] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever *et al.*, "Improving language understanding by generative pre-training", 2018.

[8] D. Hendrycks i K. Gimpel, "Gaussian error linear units (gelus)", 2023. [Mrežno]. Adresa: https://arxiv.org/abs/1606.08415

[9] S. Ioffe i C. Szegedy, “Batch normalization: Accelerating deep network training by reducing internal covariate shift”, u *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, ser. JMLR Workshop and Conference Proceedings, F. R. Bach i D. M. Blei, Ur., sv. 37. JMLR.org, 2015., str. 448–456. [Mrežno]. Adresa: http://proceedings.mlr.press/v37/ioffe15.html

[10] Y. Lecun, L. Bottou, Y. Bengio, i P. Haffner, “Gradient-based learning applied to document recognition”, *Proceedings of the IEEE*, sv. 86, br. 11, str. 2278–2324, 1998. https://doi.org/10.1109/5.726791

[11] K. He, X. Zhang, S. Ren, i J. Sun, “Deep residual learning for image recognition”, u *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016., str. 770–778. https://doi.org/10.1109/CVPR.2016.90

[12] ——, “Identity mappings in deep residual networks”, u *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, ser. Lecture Notes in Computer Science, B. Leibe, J. Matas, N. Sebe, i M. Welling, Ur., sv. 9908. Springer, 2016., str. 630–645. https://doi.org/10.1007/978-3-319-46493-0_38

[13] K. Cho, B. van Merrienboer, Ç. Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, i Y. Bengio, “Learning phrase representations using RNN encoder-decoder for statistical machine translation”, str. 1724–1734, 2014. https://doi.org/10.3115/V1/D14-1179

[14] I. Sutskever, O. Vinyals, i Q. V. Le, “Sequence to sequence learning with neural networks”, u *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, i K. Q. Weinberger, Ur., 2014., str. 3104–3112. [Mrežno]. Adresa: https://proceedings.neurips.cc/paper/2014/hash/a14ac55a4f27472c5d894ec1c3c743d2-Abstract.html

[15] D. Bahdanau, K. Cho, i Y. Bengio, “Neural machine translation by jointly learning to align and translate”, 2016. [Mrežno]. Adresa: https://arxiv.org/abs/1409.0473

[16] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, i I. Polosukhin, "Attention is all you need", *Advances in neural information processing systems*, sv. 30, 2017.

[17] J. L. Ba, J. R. Kiros, i G. E. Hinton, "Layer normalization", 2016. [Mrežno]. Adresa: https://arxiv.org/abs/1607.06450

[18] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, i N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale", u *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021. [Mrežno]. Adresa: https://openreview.net/forum?id=YicbFdNTTy

[19] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark *et al.*, "Learning transferable visual models from natural language supervision", u *International conference on machine learning*. PmLR, 2021., str. 8748–8763.

[20] I. Goodfellow, Y. Bengio, i A. Courville, *Deep Learning*. MIT Press, 2016., http://www.deeplearningbook.org.

[21] X. Zhai, B. Mustafa, A. Kolesnikov, i L. Beyer, "Sigmoid loss for language image pre-training", u *Proceedings of the IEEE/CVF international conference on computer vision*, 2023., str. 11 975–11 986.

[22] K. He, H. Fan, Y. Wu, S. Xie, i R. Girshick, "Momentum contrast for unsupervised visual representation learning", u *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020., str. 9729–9738.

[23] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, i A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments", *Advances in neural information processing systems*, sv. 33, str. 9912–9924, 2020.

[24] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba,

R. Howes, P. Huang, S. Li, I. Misra, M. G. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jégou, J. Mairal, P. Labatut, A. Joulin, i P. Bojanowski, "Dinov2: Learning robust visual features without supervision", *CoRR*, sv. abs/2304.07193, 2023. https://doi.org/10.48550/ARXIV.2304.07193

[25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, i L. Fei-Fei, "Imagenet: A large-scale hierarchical image database", u *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009., str. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

[26] T. Weyand, A. Araújo, B. Cao, i J. Sim, "Google landmarks dataset v2 - A large-scale benchmark for instance-level recognition and retrieval", u *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, June 13-19, 2020*. Computer Vision Foundation / IEEE, 2020., str. 2572–2581. https://doi.org/10.1109/CVPR42600.2020.00265

[27] J. Zhou, C. Wei, H. Wang, W. Shen, C. Xie, A. Yuille, i T. Kong, "ibot: Image bert pre-training with online tokenizer", 2022. [Mrežno]. Adresa: https://arxiv.org/abs/2111.07832

[28] M. Caron, I. Misra, J. Mairal, P. Goyal, P. Bojanowski, i A. Joulin, "Unsupervised learning of visual features by contrasting cluster assignments", *Advances in neural information processing systems*, sv. 33, str. 9912–9924, 2020.

[29] P. F. Proença i P. Simões, "Taco: Trash annotations in context for litter detection", 2020. [Mrežno]. Adresa: https://arxiv.org/abs/2003.06975

[30] T. Lin, M. Maire, S. J. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, i C. L. Zitnick, "Microsoft COCO: common objects in context", u *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, ser. Lecture Notes in Computer Science, D. J. Fleet, T. Pajdla, B. Schiele, i T. Tuytelaars, Ur., sv. 8693. Springer, 2014., str. 740–755. https://doi.org/10.1007/978-3-319-10602-1_48

[31] M. Cherti, R. Beaumont, R. Wightman, M. Wortsman, G. Ilharco, C. Gordon, C. Schuhmann, L. Schmidt, i J. Jitsev, "Reproducible scaling laws for

contrastive language-image learning", u *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, lipanj 2023., str. 2818–2829. https://doi.org/10.1109/cvpr52729.2023.00276

[32] C. Schuhmann, R. Vencu, R. Beaumont, R. Kaczmarczyk, C. Mullis, A. Katta, T. Coombes, J. Jitsev, i A. Komatsuzaki, "Laion-400m: Open dataset of clip-filtered 400 million image-text pairs", 2021. [Mrežno]. Adresa: https://arxiv.org/abs/2111.02114

[33] C. Schuhmann, R. Beaumont, R. Vencu, C. Gordon, R. Wightman, M. Cherti, T. Coombes, A. Katta, C. Mullis, M. Wortsman *et al.*, "Laion-5b: An open large-scale dataset for training next generation image-text models", *Advances in neural information processing systems*, sv. 35, str. 25 278–25 294, 2022.

[34] I. Loshchilov i F. Hutter, "SGDR: Stochastic gradient descent with warm restarts", u *International Conference on Learning Representations*, 2017. [Mrežno]. Adresa: https://openreview.net/forum?id=Skq89Scxx

Sažetak

Raspoznavanje slika predstavlja važan izazov u području računalnog vida, s mnogim praktičnim primjenama. Na početku ovog rada dan je uvid u postojeće duboke arhitekture za raspoznavanje slika, s posebnim naglaskom na modele temeljene na mehanizmu pažnje. Središnja tema rada bila je raspoznavanje slika pomoću prednaučenih samonadziranih i multimodalnih modela. U tu smo svrhu pomoću prednaučenih okosnica modela OpenCLIP i DINOv2 proveli klasifikaciju nad skupom podataka TACO. Došli smo do zaključka da korištenjem navedenih modela za klasifikaciju bez dodatnog ugađanja okosnice, postižemo znatno lošije rezultate nego ako povrh njih dodamo povezane slojeve. Nadalje, uočili smo da znatnu ulogu u gubitku točnosti igra i sama priroda skupa podataka TACO. Također smo pokazali da ansambliranjem okosnica ovih dvaju modela postižemo najbolje performanse, što nas je navelo na zaključak da OpenCLIP i DINOv2 ekstrahiraju komplementarne značajke.

Ključne riječi: duboko učenje; računalni vid; raspoznavanje slika; predučenje; samonadzirano učenje; multimodalni modeli

Abstract

Image recognition represents a significant challenge in the field of computer vision, with numerous practical applications. At the beginning of this work, an overview of existing deep architectures for image recognition is provided, with a particular emphasis on attention-based models. The central theme of this research is image recognition using pretrained self-supervised and multimodal models. To this end, we performed classification on the TACO dataset using the pretrained backbones of the OpenCLIP and DINOv2 models. We concluded that using these models in a zero-shot setting results in significantly poorer performance compared to the event of an addition of even a single fully connected layer on top of the model backbones. Furthermore, we observed that the nature of the TACO dataset itself plays a significant role in the observed loss of accuracy. Additionally, we demonstrated that ensemble learning—combining the backbones of both models—yields the best performance, leading us to conclude that OpenCLIP and DINOv2 extract complementary features.

Keywords: deep learning; computer vision; image recognition;pretraining; self-supervised learning; multimodal models