

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. xxxx

SUSTAV ZA DETEKCIJU I PRAĆENJE DLANA

Vedran Ćutić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. xxxx

SUSTAV ZA DETEKCIJU I PRAĆENJE DLANA

Vedran Ćutić

Zagreb, lipanj 2024.

Zagreb, 4. ožujka 2024.

ZAVRŠNI ZADATAK br. 1482

Pristupnik: **Vedran Ćutić (0036541466)**
Studij: Elektrotehnika i informacijska tehnologija i Računarstvo
Modul: Računarstvo
Mentor: izv. prof. dr. sc. Zoran Kalafatić

Zadatak: **Sustav za detekciju i praćenje dlana**

Opis zadatka:

Detekcija i praćenje dlana u videu ima zanimljive primjene u izgradnji naprednih sučelja za interakciju između čovjeka i računala. U okviru završnog rada treba proučiti pristupe detekciji i praćenju objekata u slikama. Odabrati prikladan pristup te programski ostvariti sustav za detekciju i praćenje dlana. Pripremiti skup slika za učenje i testiranje oblikovanog sustava. Analizirati dobivene rezultate u pogledu točnosti detekcije te računske zahtjevnosti.

Rok za predaju rada: 14. lipnja 2024.

Sadržaj

1	Uvod.....	1
2	Umjetna inteligencija.....	2
2.1	Neuronske mreže	3
2.2	Strojno učenje.....	6
2.3	Duboko učenje.....	7
2.4	Treniranje	8
2.4.1	Nadzirano učenje	9
2.4.2	Nenadzirano učenje.....	9
3	Detekcija i praćenje objekata.....	11
3.1	Obrada slike i predprocesiranje.....	11
3.2	Ekstrakcija značajki i segmentacija i detekcija.....	11
3.3	Klasifikacija slika i videa.....	13
3.3.1	Vrste klasifikacije slika	14
4	Korišteni alati	15
4.1	Python	15
4.2	YOLOv8.....	16
4.2.1	Instalacija	17
4.2.2	Treniranje	17
4.2.3	Predviđanje.....	18
4.3	Flask	19
5	Implementacija.....	20
5.1	Hand Detector	20

6	Eksperimenti i rezultati	23
7	Zaključak	24
8	Literatura	25
9	Sažetak	26
10	Summary	27

1 Uvod

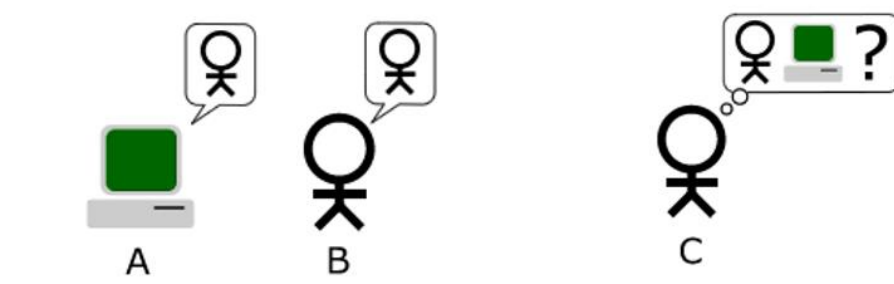
2 Umjetna inteligencija

Ciljevi računala su imitiranje čovjekove inteligencije i izvedba zadataka neizvedivih čovjeku. Eksponencijalnim razvojem umjetne inteligencije rješenja mnogih takvih problema postaje stvarnost i razvijaju se napredni algoritmi, softveri i modeli koji omogućavaju strojevima da percipiraju svoje okruženje i koriste učenje i inteligenciju kako bi to i čovjek i time povećavaju šanse za postizanje ciljeva. Takve strojeve nazivamo AI (Artificial Intelligence).

Tehnologije umjetne inteligencije imaju mnoge primjene u razvoju softvera i algoritama, u znanosti, politici. Neki primjeri su korištenje algoritama umjetne inteligencije za napredne web tražilice poput Google-a koji koristi umjetnu inteligenciju za računanje i prikazivanje najrelevantnijih rezultata i preporuka, interakcije čovjeka i računala, na primjer govorom, u autonomnim vozilima i daleko najaktualniji i najpopularniji primjer korištenja umjetne inteligencije danas su generativni modeli poput GPT-a i slično.

Alan Turing bio je prva osoba koja je provodila istraživanja pod nazivom strojne inteligencije i smatra se osnivačem umjetne inteligencije te se 1956. godine umjetna inteligencija osniva kao akademska disciplina. Poznati test inteligencije pod nazivom Turing-ov test, originalno zvan *igra imitacije*, test je koji provjerava inteligenciju računala i određuje smatra li se ono *intelligentnim*. Testira se sposobnost računala da pokaže inteligentno ponašanje slično, ili nerazlučivo čovjekovu, time se određuje je li računalo sposobno razmišljati poput čovjeka.

Test funkcionira tako da igrač **C** ispituje igrače **A** i **B** i pokušava odrediti koji je računalo, a koji čovjek. Postupak se jasno vidi na slici **Slika 2.1** Danas, više od 70 godina od Turing-ova prijedloga, nijedna tehnologija umjetne inteligencije nije uspjela proći test ispunjavajući specifične uvjete.



Slika 2.1 Opis postupka Turingova testa gdje igrač C pokušava odredit koji od igrača A i B je računalo, a koji čovjek

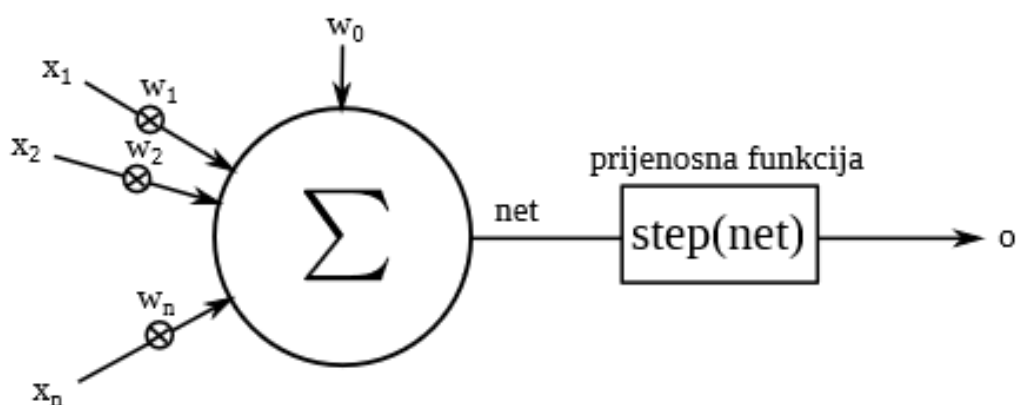
Umjetna inteligencija obuhvaća strojno učenje, duboko učenje kao i neuronske mreže. Ove discipline uključuju razvoj algoritama umjetne inteligencije, modeliranih prema procesima donošenja odluka u ljudskom mozgu, koji mogu učiti iz dostupnih podataka i svoje modele koristiti za rješavanje zahtjevnih problema i zadataka.

2.1 Neuronske mreže

Poznato je da se ljudski mozak sastoji od mnoštva međusobno povezani neurona koji rade istovremeno. Izgrađuju se sustavi *umjetnih neuronskih mreža* koji imitiraju rad ljudskog mozga. Umjetna neuronska mreža (ANN) replika je ljudskog mozga kojom se simuliraju postupci učenja i obrade podataka.

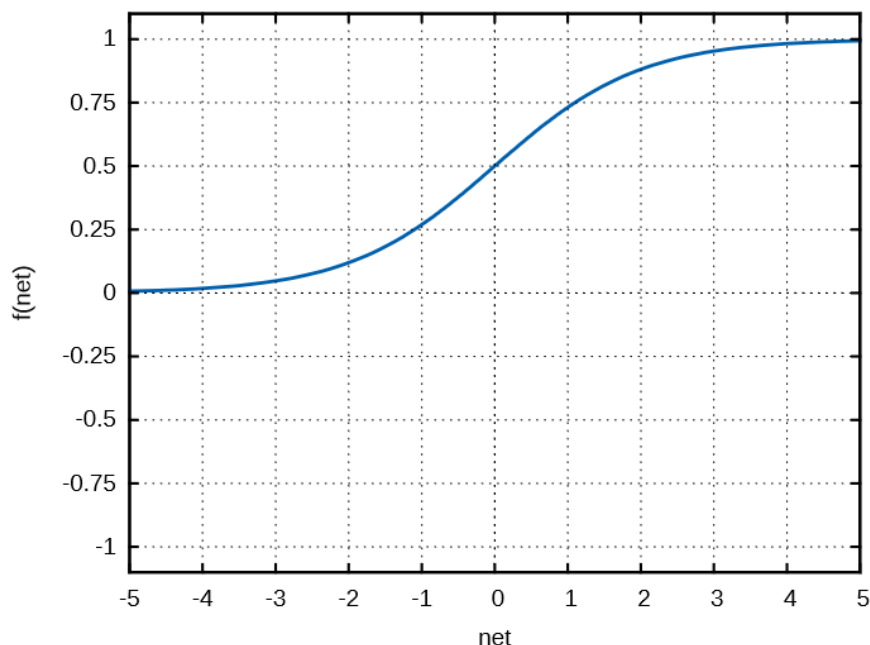
Neuroni su procesni elementi neuronske mreže inspirirani biološkim neuronima u mozgu. Biološki neuron se sastoji od tijela za obradu impulsa, *dendrite* za primanje impulsa i *akson* za prijenos impulsa do ostalih neurona. Arhitektura umjetnog neurona slična je biološkom, sadrži ulazni sloj neurona, skriveni sloj neurona koji predstavljaju tijelo biološkog neurona i izlazni sloj neurona koji računa izlazne vrijednosti.

Funkcionalnost umjetnog neurona temelji se na vrijednostima osjetljivosti i pristranosti svakog od međusobno povezanih neurona u mreži. Vrijednost sa svakog ulaza x_i u neuron se množi osjetljivošću tog ulaza w_i i akumulira u tijelu. Ukupnoj sumi dodaje se pomak w_0 , još poznat kao pristranost (eng. bias). Time je definirana akumulirana vrijednost net koja se propušta kroz prijenosnu funkciju čime nastaje izlazna vrijednost. Prikaz modela umjetnog neurona vidi se na slici (Slika 2.2).



Slika 2.2 Model umjetnog neurona

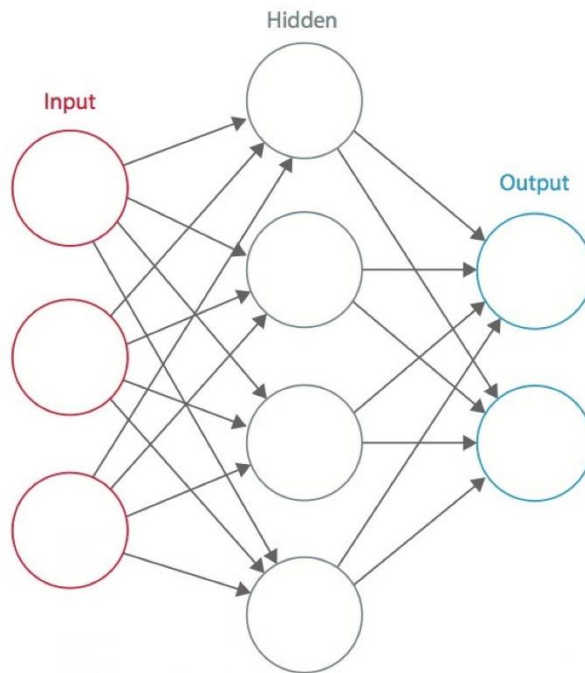
Prijenosna funkcija smanjuje linearnost i omogućuje rješavanje nelinearnih problema. Neke od često korištenih prijenosnih funkcija su funkcija identiteta, funkcija skoka, sigmoidalna funkcija(**Slika 2.3**), tangens hiperbolni.



Slika 2.3 Sigmoidalna funkcija, vraća vrijednosti između 0 i 1, simetrična je

Duboke neuronske mreže proširenje su umjetnih neuronskih mreža. Za razliku od konvencionalnih neuronskih mreža duboke neuronske mreže obično sadrže mnogo skrivenih slojeva i velik broj neurona čiji broj dostiže milijune u nekim kompleksnijim primjerima.

Duboke neuronske mreže su se pokazale da probleme računalnog vida podižu na još višu razinu točnosti i učinkovitosti, zahvaljujući konvolucijskim neuronskim mrežama (CNN), te zbog toga su najčešće rješenje problema klasifikacije slika. Konvolucijska neuronska mreža proširena je inačica umjetne neuronske mreže (ANN) koja se primarno koristi za procesuiranje podataka matričnog oblika, poput slika. CNN-ovi oponašaju neuronske mreže ljudskog uma u više slojeva: slojevi ulaznih podataka, konvolucijski slojevi, ReLU slojevi, slojevi sažimanja i sloj izlaznih podataka. Pojednostavljena arhitektura konvolucijske neuronske mreže prikazana je na slici **Slika 2.4**.



Slika 2.4 Arhitektura konvolucijske mreže sa ulaznim slojem sa 3 neurona, jednim međuslojem s 4 neurona i izlaznim slojem sa 2 neurona

Konvolucijski slojevi izvode linearno-transformacijske operacije među dvije matrice, gdje je prva matrica filter koji se može naučiti, a druga matrica je ograničeni dio ulazne matrice. Filter klizi po visini i širini slike stvarajući slikovnu reprezentaciju čime se dobiva aktivacijska mapa, dvodimenzionalni prikaz slike. Treniranjem na podacima mreža će naučiti sve filtre.

Sloj ReLU (Rectified Linear Unit) postao je popularan u zadnjih par godina. ReLU sloj se generalno nalazi iza konvolucijskog sloja i on svaku negativnu vrijednost pretvara u nulu.

Sloj sažimanja smanjuje veličinu slika što čini izračun bržim te smanjuje memorijsku i vremensku kompleksnost. Neke od vrsta sažimanja su maksimalno sažimanja i prosječno sažimanje.

Sposobnost neuronskih mreža da identificiraju obrasce, rješavaju kompleksne zadatke i prilagode se promjenjivom okruženju je ključna. Razvoj umjetne inteligencije uvelike ovisi o neuronskim mrežama, koje također pokreću inovacije i utječu na smjer razvoja tehnologije. Neuronske mreže osnovna su komponenta mnogih tipova učenja umjetne inteligencije poput strojnog i dubokog učenja.

2.2 Strojno učenje

Strojno učenje koristi podatke i algoritme kako bi omogućilo umjetnoj inteligenciji da imitira čovjekov način učenja, postepeno poboljšavajući točnost i uspješnost.

Algoritam algoritma strojnog učenja generalno se dijeli na tri glavna dijela:

1. **Proces odluke:** Na temelju ulaznih podataka algoritam proizvodi procjenu uzoraka u njima.
2. **Funkcija pogreške:** Procjenjuje se predviđanje modela, na temelju poznatih primjera može se odrediti točnost modela.
3. **Proces optimizacije modela:** Težine se prilagođavaju kako bi se smanjila pogreška. Algoritam će ponoviti ovaj iterativni proces "procjene i optimizacije", autonomno ažurirajući težine dok se ne postigne zadovoljavajući rezultat.

Klasično strojno učenje često ovisi o ljudskoj intervenciji u učenju. Ljudski stručnjaci određuju skup značajki kako bi razumjeli razlike između podataka. Postoje mnogi algoritmi i načini strojnog učenja koji se ovisno o problemu, dubini i kompleksnosti odabiru za rješenje:

- **Neuronske mreže**
- **Linearna regresija:** Postupak predviđanja vrijednosti na temelju linearne zavisnosti između varijabli.
- **Logistička regresija:** Predviđanja kategorijskih varijabli poput da/ne
- **Grupiranje:** Identificiranje uzoraka u podacima i njihovo grupiranje u razrede.
- **Stablo odluke:** Koriste se za predviđanje vrijednosti i za razvrstavanje podataka u razrede. Koristi niz grananja povezanih odluka koje se mogu prikazati dijagramom stabla.
- **Nasumične šume:** Kombinacijom izlaza više stabala odluke postiže se jedan rezultat.

Mnoge su prednosti strojnog učenja jer je moguće uz malu ljudsku intervenciju uočiti uzorke i trendove u ogromnim skupovima podataka. Algoritmi se neprestano poboljšavaju s više unosa podataka, što znači da su potrebne velike količine podataka za obuku, što povećava vjerojatnost za greškom.

2.3 Duboko učenje

Strojno učenje i duboko učenje koriste *neuronske mreže* za učenje iz ogromnih količina podataka, razlikuju se u vrsti neuronske mreže koji koriste i u količini ljudske intervencije.

Klasično strojno učenje se najčešće uče nadziranim učenjem, što znači da koristi podatke koji su strukturirani ili označeni, dok algoritmi dubokog učenja koriste duboke neuronske mreže koje se sadrže od više skrivenih slojeva koje omogućavaju nenadzirano učenje koje automatizira ekstrakciju značajki iz nestrukturiranih podataka. Duboke neuronske mreže se obučavaju na velikim količinama podataka kako bi identificirale i klasificirale fenomene, prepoznale obrasce i odnose, procijenile vjerojatnosti te donosile predviđanja i odluke.

Svaki od međusobno povezanih čvorova se nadograđuje na prethodni sloj kako bi se poboljšalo i optimiziralo predviđanje ili kategorizacija. Napredovanje i učenje kroz mrežu poznato je pod nazivom *forward propagation*. Drugi proces učenja, *backpropagation*, mreže povratno se širi kroz mrežu izračunavajući grešku zatim prilagođava težine i pristranosti mreže pomicanjem unatrag kroz slojeve u nastojanju da se uvrjeba model. Širenje unaprijed i natrag omogućuje neuronskoj mreži da napravi predviđanja i ispravi što više pogrešaka i na taj način algoritam postupno postaje precizniji.

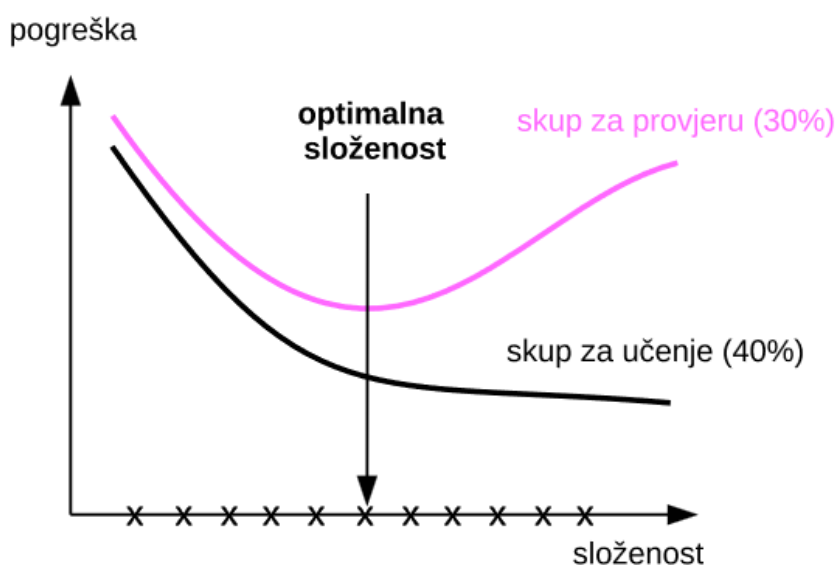
Duboko učenje zahtijeva ogromnu količinu računalne snage. Grafičke procesorske jedinice (GPU) visokih performansi idealne su za učenje modela dubokim učenjem jer mogu podnijeti veliku količinu izračuna u više jezgri s velikom dostupnom memorijom. Grafičke kartice su optimizirane za matrične operacije, a same neuronske mreže i njihove težine i pristranosti mogu se prikazati kao matrice.

Generativni modeli aktualan su primjer primjene dubokog učenja. To su modeli umjetne inteligencije koji se odnose na modele dubokog učenja koji mogu uzeti neobrađene podatke i naučiti statistički vjerojatne rezultate. Porast dubokog učenja omogućio je njihovo proširenje i jednostavno korištenje dostupno svima.

2.4 Treniranje

Postoje različite metode treniranja koje ovise o dostupnim podacima i zadatku kojeg pokušavamo odraditi. Osim metoda treniranja bitno je kvalitetno odrediti skup podataka jer će kvaliteta mreže ovisiti o njegovoj kvaliteti. Loše reprezentativni ili mali skupovi podataka rezultirat će mrežama koje neprecizno generaliziraj naučeno na neviđene ulazne podatke.

Čest je problem prenaučivosti (engl. *overfitting*), što predstavlja model koji se previše naučio nad podacima za treniranje i njih savršeno generalizira i klasificira, ali loše i s niskom preciznošću prepoznaje neviđene ulaze, model postaje *štreber*. Generalno se savjetuje podjela skupa podataka na podskup za učenje (engl. *train*), podskup za provjeru (engl. *valid*) kojim se provjerava uspješnost i podskup za testiranje (engl. *test*). Omjer podjele obično je 40% na podskup za učenje, 30% na podskup za testiranje i 30% na podskup za provjeru. Ovaj postupak naziva se unakrsna provjera i njime se određuje optimalna složenost modela, nju je moguće odrediti iz grafa (**Slika 2.5**), naime optimalna složenost je u točki gdje pogreška skupa za provjeru počinje rasti, a pogreška skupa za učenje i dalje se smanjuje.

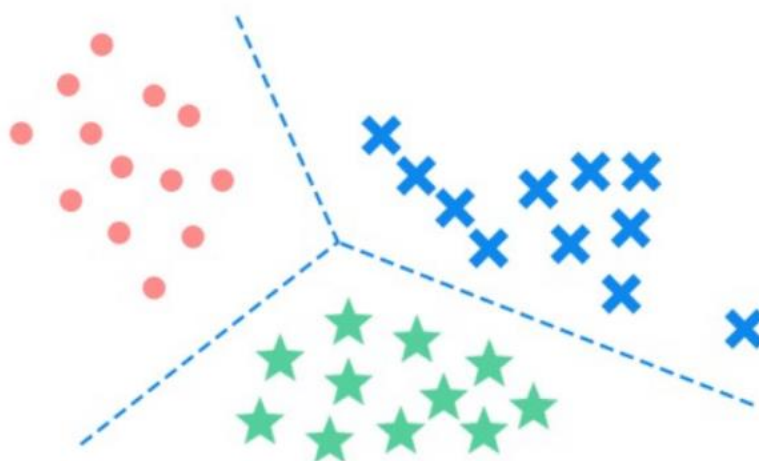


Slika 2.5 Graf postotka pogreške na skupovima za provjeru i učenje, određuje se optimalna složenost unakrsnom provjerom

2.4.1 Nadzirano učenje

Nadzirano strojno učenje algoritam je učenja na označenim skupovima podataka, sa već određenim odnosom ulaza i ispravnih izlaza. Skup podataka slika je već označen i klasificiran. Na primjer ulazni skup podataka može se sastojati od slika životinja koje su označene ovisno o vrsti, slike pasa pripadaju klasi pas, slike mačaka pripadaju klasi mačka i slično. Dovođenjem učenja, analizom novih ulaza algoritam na temelju podataka za učenje zaključuje i predviđa razrede novih slika.

Algoritmi nadziranog učenja dijele se na klasifikaciju s jednim razredom koji sliči dodjeljuju samo jedan razred poput dlana ruke i klasifikaciju s više razreda (**Slika 2.6**). Mnogi algoritmi implementiraju neku vrstu nadziranog strojnog učenja. Jedan od takvih je stablo odluke koja se lako implementira i široko je primjenjiva. Stablo odluke funkcionira poput dijagrama toka gdje se na svakom čvoru određuje u kojem će se smjeru dalje kretati sve dok se ne dođe do lista koji predstavlja ciljnu vrijednost.



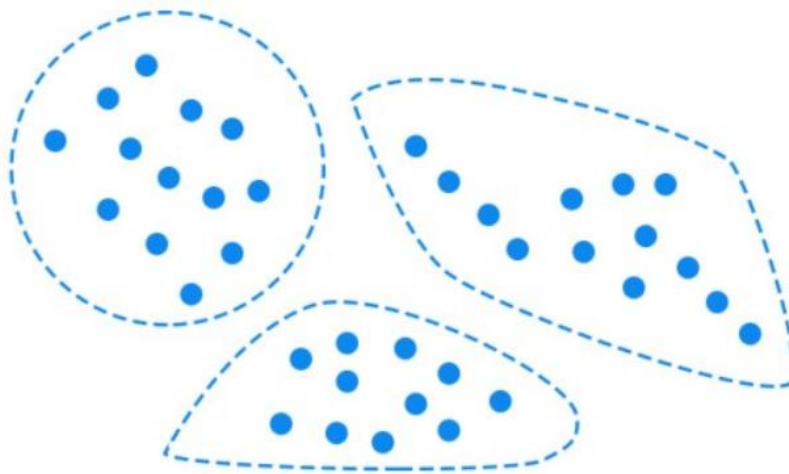
Slika 2.6 Primjer klasifikacije u 3 razreda ulaznog skupa podataka uz nadzirano učenje

2.4.2 Nenadzirano učenje

Za razliku od nadziranog učenja, skupovi podataka u algoritmima nenadziranog učenja nisu označeni niti unaprijed klasificirani, ovi algoritmi pronalaze pravilnosti i sličnosti u podacima grupirajući, otkrivajući strukturne vrijednosti i smanjenjem dimenzionalnosti. Skup podataka mogu biti slike životinja, ali bez klasifikacija, algoritam grupira podatke na temelju nekih

sličnosti pa će na primjer psi biti grupirani u jednu grupu na temelju sličnih karakteristika, dok će mačke biti grupirane u drugu grupu. Primjer grupiranja ulaznih podataka vidimo na slici **Slika 2.7**.

Cilj nenadziranog učenja je istražiti i razumjeti prirodu danih podataka. Algoritam tumači podatke prema vlastitim uvjetima, prepoznaje uzorke i izvlači zaključke. Najčešći koncept koji koristi nenadzirano učenje je grupiranje podataka na temelju nekih sličnosti i uzoraka kojim dobijemo grupe podataka čiji će se razredi kasnije odrediti.



Slika 2.7 Primjer grupiranja podataka uz nenadzirano učenje

3 Detekcija i praćenje objekata

Detekcija i praćenje objekata je proces pronalaženja, klasificiranja i praćenja objekata na slikama i videima. Postupak se izvodi u nekoliko slijednih koraka:

1. **Obrada slike i predprocesiranje**
2. **Ekstrakcija značajki i segmentacija i detekcija**
3. **Klasifikacija**
4. **Postprocesiranje**
5. **Sinteza**

3.1 Obrada slike i predprocesiranje

Svaka slika sačinjena je od tisuće piksela koji se u tehnikama klasifikacije tretiraju kao liste matrica, veličine koja ovisi o rezoluciji same slike. Svaki algoritam implementira neke metode koje će poboljšati kvalitete slike i pripremiti ih za daljnju analizu. Slikama se često mijenjaju dimenzije (visina i širina) ili izrezuju samo dijelovi slike radi uklanjanja nebitnih dijelova slike ili radi smanjenja veličine i kompleksnosti.

Koristi se metoda normalizacije slike gdje se vrijednost svakog piksela podešava na normalnu distribuciju. Svakom pikselu se oduzima srednja vrijednost i dijeli se sa standardnom devijacijom vrijednosti piksela. Mnoge slike sadrže šum koji može negativno utjecati na efikasnost modela pa je potrebno koristiti neke od tehnika filtriranja slike poput Gaussovog filtriranja ili srednjeg filtriranja. Ove metode pospješuju uspješnost i kvalitetu modela. Nakon prve obrade slike ona je spremna za daljnju analizu.

3.2 Ekstrakcija značajki i segmentacija i detekcija

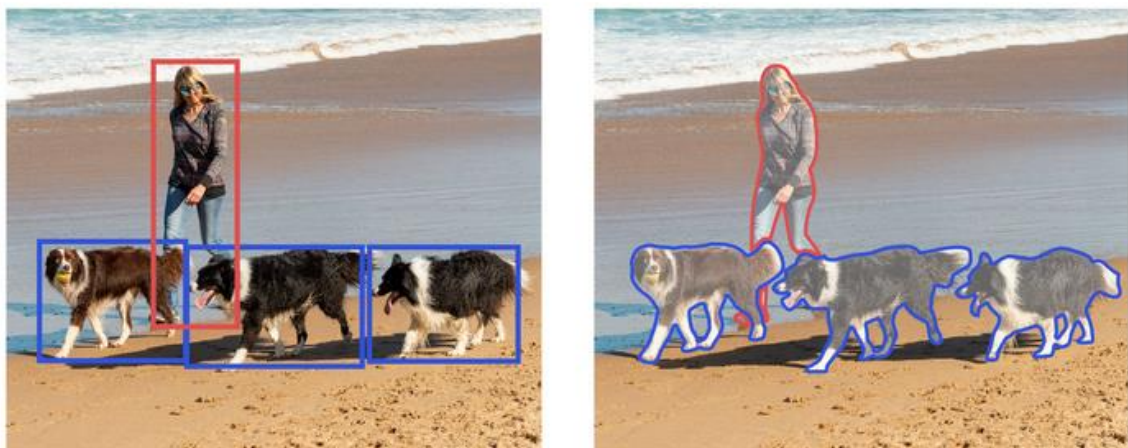
Ekstrakcija značajki je bitan proces prepoznavanja uzoraka unutar slike koji će se koristiti za razlikovanje jednog objekta od drugog. Uzorci su obično specifični za svaki razred što rezultira jasnom razlikom razreda. U slučaju klasifikacije jabuka i banana postoje značajke poput oblika i boje koje će se koristiti za razlikovanje ta dva razreda.

Ekstrakcija značajki uvelike poboljšava performanse modela fokusiranjem na najrelevantnije dijelove svake slike, u suprotnom bi se svaki put trebala analizirati cijela slika

što utječe na kompleksnost i točnost modela. Otkrivanje rubova na slici korisna je i bitna metoda jer pomoću nje uočavamo granice između područja na slici. Najčešće metoda otkrivanja rubova na slici je korištenje operatora gradijenta koji određuje di dolazi do prijelaza između elemenata na slici. Analiza teksture analizira sliku i pronalazi ponavljajuće uzorke što se koristi za prepoznavanje različitih materijala ili površina objekata, ovime se prepoznaju tumori na medicinskim slikama jer se tekstura zdravog tkiva razlikuje od teksture kancerogenog tkiva.

Slike se sastoje od informacija bitnih za analizu i onih nebitnih. Koriste se mnogi načini podijele slike na bitne dijelove za analizu čime se rezultati postupka klasificiranja objekata na slici značajno poboljšava. Segmentacija je postupak podijele slike ili videa na područja interesa za identificiranje i razlikovanje objekata ili zanimljivih dijelova. Područja od interesa mogu biti bilo kakvi objekti koji se ističu na slici – ljudi, automobili, zrakoplovi, životinje. U kontekstu detekcije i praćenja dlana dlan i ruka su područja interesa. Najčešći oblik je semantička segmentacija koja svakom pikselu na slici daje oznaku.

Detekcija lokalizira određeni objekt unutar slike često pružajući granične okvire njegove lokacije, što olakšava i ubrzava proces klasifikacije objekta unutar slike. Segmentacija slike daje detaljnu analizu gdje je izlazna slika podijeljena piksel po piksel dok nam detekcija daje granične okvire objekata. Primjer razlike segmentacije i detekcije vidi se na slici **Slika 3.1**.



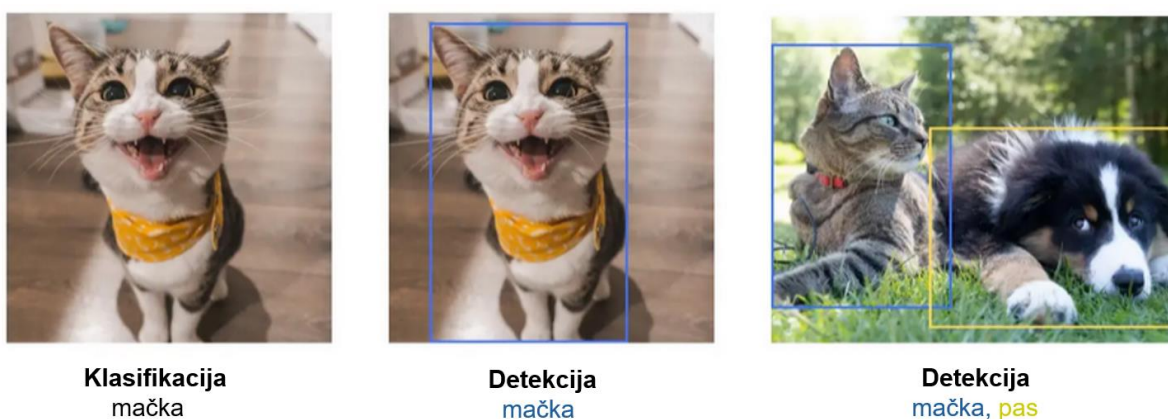
Slika 3.1 Razlika između detekcije(lijevo) i segmentacije(desno)

3.3 Klasifikacija slika i videa

Kako bi model mogao odrediti je li se na slici nalazi dlan, potrebno je odrediti razrede segmentiranih i detektiranih objekata, to se izvodi postupkom *klasifikacije*. Klasifikacija slika i videa zadatak je strojnog učenja da identificira i odredi što slika ili video predstavljaju. Model se trenira na skupu podataka za učenje koji sadrži različite razrede i oznake. Budući da je video skup slika, mnoga se jednaka pravila koriste za klasifikaciju slika kao i za klasifikaciju videa. Slici se dodjeljuje jedna ili više oznaka ili razreda na temelju postojećih podataka za treniranje već označenih slika. Dobivamo rezultate koji nam govore nalaze li se na slici određeni objekti, atributi ili uzorci. Neki od mnogih slučaja upotrebe klasifikacije slika i videa su: Automatska inspekcija i kontrola kvalitete artikala, Prepoznavanje objekata u autonomnim vozilima, Prepoznavanje lica u sigurnosnim snimkama.

Napretkom strojnog učenja u zadnjih nekoliko godina brzo i napreduju modeli klasifikacije i detekcije objekata. Danas, za razliku od prije samo deset godina klasifikacija objekata je brzo izvediva, ne zahtjeva toliko resurse i jednostavna je za implementaciju.

Potrebno je razlikovati klasifikaciju, i detekciju objekata. Klasifikacija dodjeljuje razred slici ili objektu dok detekcija objekta specificira područje fotografije ili videa na kojem se objekt određenog razreda nalazi i najčešće ga stavlja unutar graničnog okvira umjesto da klasificira cijelu sliku. Razlika je jasno vidljiva na slici **Slika 3.2**.

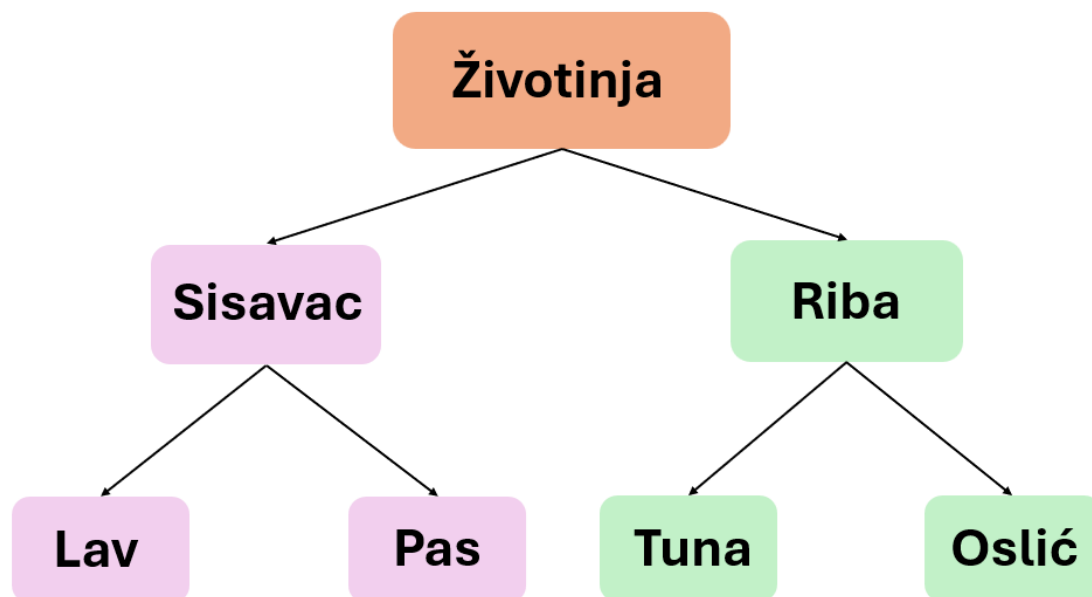


Slika 3.2 Primjer klasifikacije i detekcije mačke i psa na 3 slike gdje se jasno vidi razlika između klasifikacije i detekcije

3.3.1 Vrste klasifikacije slika

Postoje različite metodologije klasifikacije slike koje se mogu koristiti u rješavanju širokog opsega zadataka i problema.

- **Binarna klasifikacija** klasificira sliku u jednu od dva razreda. Slika će pripadati ili jednom ili drugom razredu, što ima mnoge i široke primjene, poput prepoznavanja tumora, uočavanje nedostataka određenih objekata ili nešto poput klasifikacije životinje u razrede pas ili mačka.
- **Višerazredna klasifikacija** slična je binarnoj, kategorizira sliku u tri ili više razreda i ima širu primjenu jer se njome slike mogu detaljnije i kompleksnije analizirati i kategorizirati, moguće je napraviti model koji će na temelju slike prepoznati koju to životinju predstavlja, a ne poput binarne samo jednu od dvije moguće klasifikacije.
- **Hijerarhijska klasifikacija** za zadatak ima organizaciju klasa u hijerarhijsku strukturu na temelju njihove sličnosti, moguće je klasificirati životinje na sisavce i ribe te zatim sisavce i ribe dodatno podijeliti na specifičnije članove tog skupa poput lav, tuna, i slično. Primjer hijerarhijske klasifikacije slike vidimo na **Slika 3.3**.



Slika 3.3 Primjer hijerarhijske klasifikacije slike

4 Korišteni alati

Problem detekcije i praćenja objekata na slici i videu nije jednostavan proces. Primjenjivost i efikasnost metoda ovisi o mnogim čimbenicima, na primjer kvaliteta slike, broj sličica po sekundi na videu, šumovi na podacima i slično. Složenost najvećim dijelom proizlazi iz samih uvjeta u kojima se vrši detekcija. Kretnja dlana i ruke nepredvidiva je, brza i često promjenjiva, osim tog i pozadina sama po sebi može biti nepredvidiva. Količina vanjskog svjetla, sjene i slično drastično utječu na efikasnost modela. Spoj ili kombinacija spomenutih svojstava definira uvjete u kojima se vrši detekcija, koje je nužno uzeti u obzir pri odabiru metoda.

Računalni vid danas je glavni alat za složenu analizu slike, prvenstveno u onim slučajevima kada je potrebno postići seciranje na onoj razini na kojoj to radi ljudski mozak. Računalni vid podvrsta je umjetne inteligencije gdje se promatraju vizualni podaci. Raznovrsnim se metodama pokušava postići viša razina razumijevanja vizualnih podataka kako bi se povećala primjenjivost u što većem broju slučajeva. Jedan od najčešćih zadataka računalnog vida je raspoznavanje objekata na slici ili videu.

Danas dolazi do ubrzanog razvoja ovog područja zbog novih specifičnih potreba za raspoznavanjem predmeta, stalnim napretkom, poboljšanjem starih metoda i stalni razvoji novih. Postoje mnogi sustavi temeljeni na odabranim algoritmima koji posjeduju ugrađene vrijednosti čije stanje ovisi o nizu ulaznih podataka u procesu treniranja modela. Ti sustavi su prilagodljivi različitim zadacima i problemima. Neki od poznatijih konkretnih primjera su **YOLO** (You Only Look Once), **TensorFlow** i **EfficientDet**.

U sklopu ovog rada u svrhu rješavanja problema detekcije i praćenja dlana korišten je **YOLOv8** model koji je implementiran unutar **Flask** aplikacije koristeći **python**.

4.1 Python

Python je objektno orijentirani programski jezik visoke razine s dinamičkom semantikom. Sadrži mnoge ugrađene podatkovne strukture, jednostavnu sintaksu koju je lako naučiti, čitljiv je i stoga smanjuje troškove održavanja programa što ga čini vrlo atraktivnim za brzi razvoj aplikacija, kao i za korištenje kao jezik za povezivanje postojećih komponenti. Python podržava module i pakete, što potiče modularnost programa i ponovnu upotrebu koda. Python-

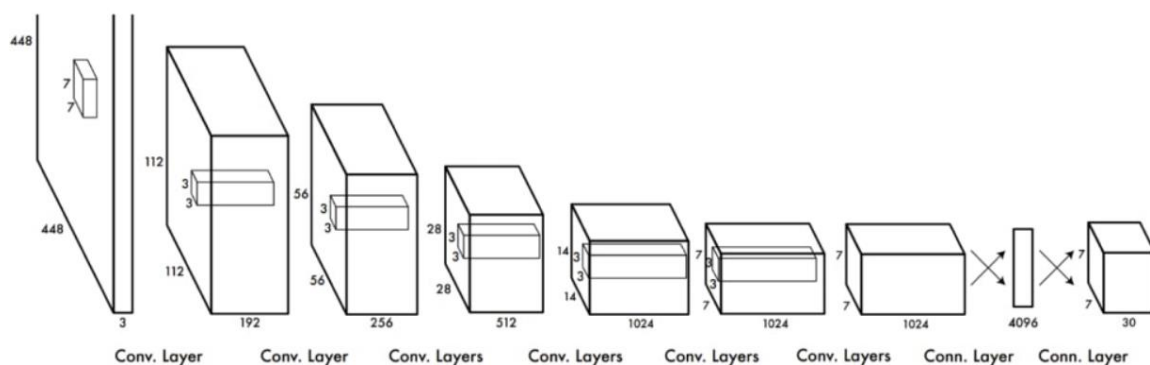
ov *interpreter* i opsežna standardna biblioteka javno su dostupni i mogu se besplatno distribuirati.

Python-ova jednostavnost i brzina razlog su njegove rasprostranjenosti, jednostavan je i intuitivan za početnike i jednostavne programe, a implementira kompleksne i složene metode i algoritme za razvoj naprednijih programa.

4.2 YOLOv8

YOLO (You Only Look Once) vrhunski je model računalnog vida kojeg razvija *Ultralytics*, platforma umjetne inteligencije za stvaranje, treniranje i implementiranje modela strojnog učenja sa sučeljima bez izvornog koda za jednostavnost i praktičnost uporabe. Ultralytics pojednostavljuje proces i pruža rješenja koja se lako implementiraju.

Arhitektura dostiže veliki ugled i raširenost zahvaljujući brzini raspoznavanja s relativno visokom točnosti. Ova svojstva omogućavaju primjenu u pravom vremenu i kvalitetnu analizu videozapisa. YOLO arhitektura se razlikuje od ostalih jer na prepoznavanje predmeta gleda kao na problem regresije do prostorno odvojenih poznatih okvira i pridruženih vjerojatnosti mogućih klasa. Tijekom analize se promatra čitava slika što pospješuje preciznost jer se predviđanja temelje na globalnom kontekstu. Ovo su posljedice upotrebe samo jedne konvolucijske neuronske mreže koja dijeli sliku u niz podmreža kojima je raspodijeljen zadatak određivanja mogućih okvira, pridružene sigurnosti i klasificiranja. Dodatnim metodama i vektorizacijom odstranjuju se nevjerojatni okviri te utvrđuju konačna predviđanja. Prikaz jedne neuronske mreže vidimo na slici **Slika 4.1**.



Slika 4.1 Arhitektura konvolucijske neuronske mreže YOLOv8 modela

YOLOv8 ažurirana je verzija popularnog YOLO algoritma za raspoznavanje objekata, koji je predstavljen 2016. Model sadrži brojne arhitektonske i razvojne promjene i poboljšanja u odnosu na prijašnje verzije. Utvrđeno je kako uspijeva pronaći i klasificirati objekte različitih razreda na slikama s visokom točnošću.

Novo razvijena arhitektura YOLOv8 koristi princip raspoznavanja bez sidra, što znači da sve objekte predviđa direktno u njihovu centru umjesto koristeći granični okvir poznat kao *anchor box*. Granični okviri bili su nepredvidiv dio ranijih YOLO modela, budući da mogu predstavljati distribuciju okvira ciljne referentne vrijednosti, ali ne i distribuciju prilagođenog skupa podataka. Prepoznavanje bez sidra smanjuje kompleksnost i ubrzava proces prepoznavanja.

4.2.1 Instalacija

YOLOv8 primarno koristi programski jezik *python* za implementaciju i pokretanje, stoga je prije same instalacije potrebno na vlastito računalo instalirati programski jezik python sa službene stranice. Potreban je i *pip*, alat koji nam omogućava i olakšava instalaciju python paketa. Kada su te dva zahtjeva ispunjena moguće je instalirati paket *ultralytics* i sve njegove zahtjeve pozivanjem sljedeće komande unutar naredbenog retka:

```
pip install ultralytics
```

Yolov8 može se koristiti direktno u sučelju naredbenog retka (**CLI**) komandom `yolo`. Može se koristiti na razne načine i za mnoge zadatke, prima mnoge argumente i sadrži različite načine rada.

4.2.2 Treniranje

Glavna funkcionalnost YOLOv8 modela je treniranje i učenje na skupu podataka. Poziva se funkcija treniranja koja kao argumente prima skup podataka (*data*), veličinu slike (*imgsz*) i broj epoha (*epoch*). Epoha predstavlja jedan prolaz algoritma kroz sve podatke na kojima se model trenira. Postoje mnogi argumenti koji se mogu postaviti, ali ovo su najbitniji i oni koji su se koristili u sklopu ovog rada. Metoda treniranja može se pozvati iz naredbenog retka:

```
yolo detect train data=coco8.yaml model=yolov8n.pt epochs=100  
imgsz=640
```

Način treniranja u YOLOv8 osmišljen je za djelotvorno treniranje modela raspoznavanja objekata, uz potpuno korištenje modernih hardverskih mogućnosti. Prije samog treniranja potrebno je pripremiti skup podataka za učenje. Potrebno je prikupiti slike, označiti ih i izvesti u odgovarajućem formatu. Koristan alat je **Roboflow** koji se koristi za označavanje i predprocesiranje slika i podataka te sadrži javno dostupne gotove skupove podataka za sve korisnike, nudi formate za treniranje YOLOv8 modela. Važno je napomenuti da svaki skup podataka sadrži skup slika za učenje, skup slika za provjeru i skup slika za testiranje koji se koriste za optimizaciju i pronalaženje najpreciznijeg rješenja i pomaže u smanjenju prenaučivosti.

Potrebno je uzeti u obzir brzinu treniranja, mnoga današnja računala nisu optimizirana za zadatke matričnog množenja što može uvelike usporiti treniranje podataka. Jedna epoha treniranja podataka sa više od 1000 slika na računalu bez ugrađene GPU može trajati do 15 minuta, dok se sa ugrađenom GPU testiranje od 100 epoha na jednakom skupu podataka može izvesti u sat vremena.

Nakon završetka treniranja modela, dobiva se skup istreniranih težina *best.pt* koji se dalje može koristiti za raspoznavanje objekata. Dobiju se i mnogi rezultati, analize i statistike treniranja modela. Svi rezultati spremaju se u datoteku *runs*.

Modele je moguće izvesti funkcijom *export* te postoji mogućnost validacije modela kojim se procjenjuje kvaliteta obučanih modela. *Val* je funkcija koja pruža širok skup alata i metrika za procjenu izvedbe vlastitih modela prepoznavanja objekata.

4.2.3 Predviđanje

YOLOv8 nudi snažnu značajku koja obavlja funkcionalnost predviđanja i raspoznavanja objekata koja koristi gotove modele ili vlastito trenirane. Moguće je donositi zaključke nad slikama, videima i prijenosu uživo u stvarnom vremenu.

Predviđanje se izvodi pozivom modela i prijenosom argumenata izvora na kojem će se izvoditi predviđanje s mogućnošću spremanja novonastale slike ili videa na kojem je izvedeno prepoznavanje objekata:

```
model = YOLO("yolov8n.pt")
```



```
model.predict("slika.jpg", save=True, imgsz=320)
```

Rezultat predviđanja biti će slika ili video na kojem su unutar graničnih okvira označeni objekti za koje je model treniran da prepoznaje. Moguće je manipulirati elementima *Boxes* koji predstavljaju granične okvire, može im se mijenjati format, oblik, indeks i dodatne informacije koje se ispisuju poput vjerojatnosti raspoznavanja točnog razreda i sami nazivi razreda.

Predviđanje je moguće samostalno pokrenuti na slikama i videima ili je moguće ih implementirati u sklopu aplikacija da se dobiveni rezultati koriste za daljnje funkcionalnosti.

4.3 Flask

Flask je okvir za izradu web aplikacija, to je python paket koji omogućava jednostavan razvoj aplikacija. Brz je i jednostavan, osmišljen je kako bi se brzo izradile jednostavne aplikacije, uz mogućnost skaliranja do složenih aplikacija. Ima malu jezgru koju je lako proširiti. Jednostavnost i brzina razvoja Flask aplikacije čini ga savršenim za izbor za razvoj jednostavne aplikacije koja koristi trenirane modele za detekciju i praćenje dlana. Instalacija Flask paketa izvršava se pozivanjem sljedeće komande unutar naredbenog retka:

```
pip install -U Flask
```

Instalacijom može se početi sa izradom aplikacije ili se može koristiti za pokretanje gotovih aplikacija lokalno na računalu.

5 Implementacija

5.1 Hand Detector

Razvijena je web aplikacija Hand Detector u sklopu ovog rada koja omogućuje detekciju i praćenje dlana na slici i videu. Pokretanje aplikacije moguće je unutar naredbenog retka unutar datoteke unutar koje se aplikacija nalazi komandom:

```
flask run
```

Implementacija je podijeljena u više datoteka sa nekoliko *python* modula i skripti te nekoliko *html* i *css* datoteka.

routes.py - središnji modul koji postavlja rute web aplikacije, to je tipični princip pisanja Flask aplikacije gdje su sve moguće rute upisane unutar jednog modula

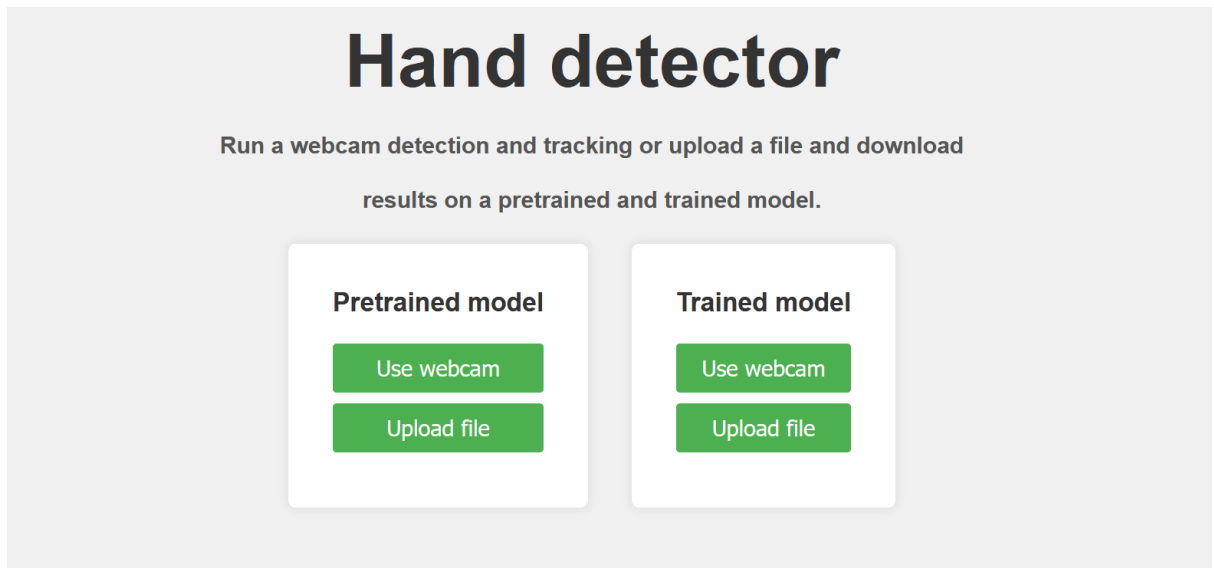
templates – datoteka koja sadrži *html* datoteke poput početne stranice, stranice za učitavanje vlastitih datoteka i stranice za pokretanje web kamere

upload.py – *python* modul koji se pokreće učitavanjem datoteke i njenim predviđanjem, unutar modula stvara se YOLOv8 model nad kojim se zatim izvodi predviđanje, detekcija i praćenje dlana, nakon uspješno provedenog predviđanja korisniku se preusmjerava na novu stranicu na kojoj može preuzeti novonastalu datoteku

webcam.py - *python* modul koji se pokreće prilikom pokretanja web kamere, modul je zadužen za njeno pokretanje i za izvođenje predviđanja, detekcije i praćenja dlana u stvarnom vremenu na web kameri korištenjem YOLOv8 modela koji se stvara unutar modula, web kamera se otvara kao skočni prozor te je moguće zaustaviti proces pritiskom tipke **ESC**

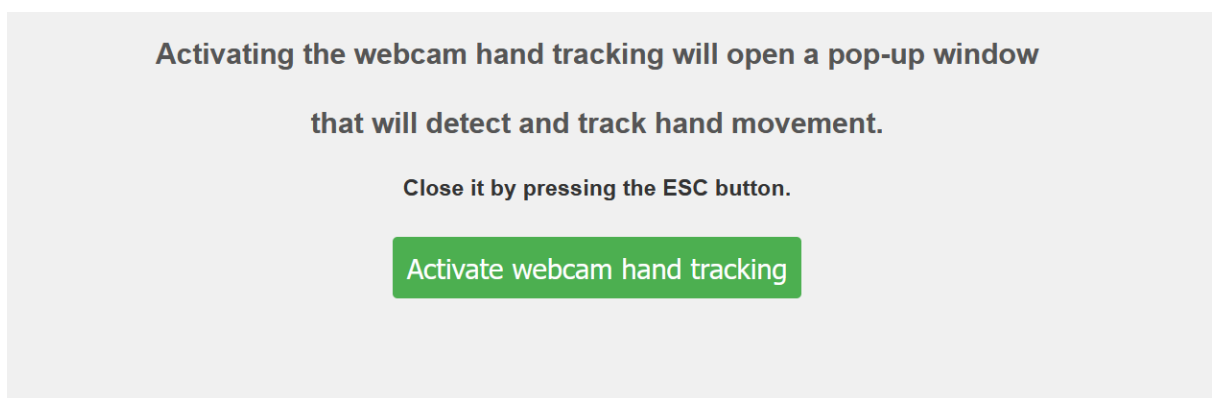
best.pt – datoteka koja predstavlja vlastito istrenirani YOLOv8 *PyTorch* model

Pokretanjem poslužitelja aplikacije možemo ju otvoriti lokalno na web-pregledniku. Aplikacija omogućuje dvije mogućnosti: upaliti web kameru te detektirati dlan u stvarnom vremenu i učitavanje videa ili slike sa računala. Moguće je pokrenuti detekciju korištenjem već istreniranog modela ili korištenjem modela koji je istreniran u svrhu ovog rada. Početna stranica aplikacije prikazana je na slici **Slika 5.1**.



Slika 5.1 Početna stranica web aplikacije *Hand Detector* na kojoj je moguć odabir detekcije dlana web kamerom ili učitavanjem datoteke

Odabirom detekcije u stvarnom vremenu dobivamo mogućnost upaliti web kameru i pokrenuti proces detekcije. U pozadini se pokreće *python* skripta *webcam.py* koja će pokrenuti web kameru i u stvarnom vremenu detektirati i pratiti dlan. Stavlja se granični okvir oko dlana i ispisuje razred kojem prepoznati objekt pripada sa kojom vjerojatnošću. Sučelje stranice za praćenje i detekciju dlana u stvarnom vremenu korištenjem web kamere prikazano je na slici **Slika 5.2**.

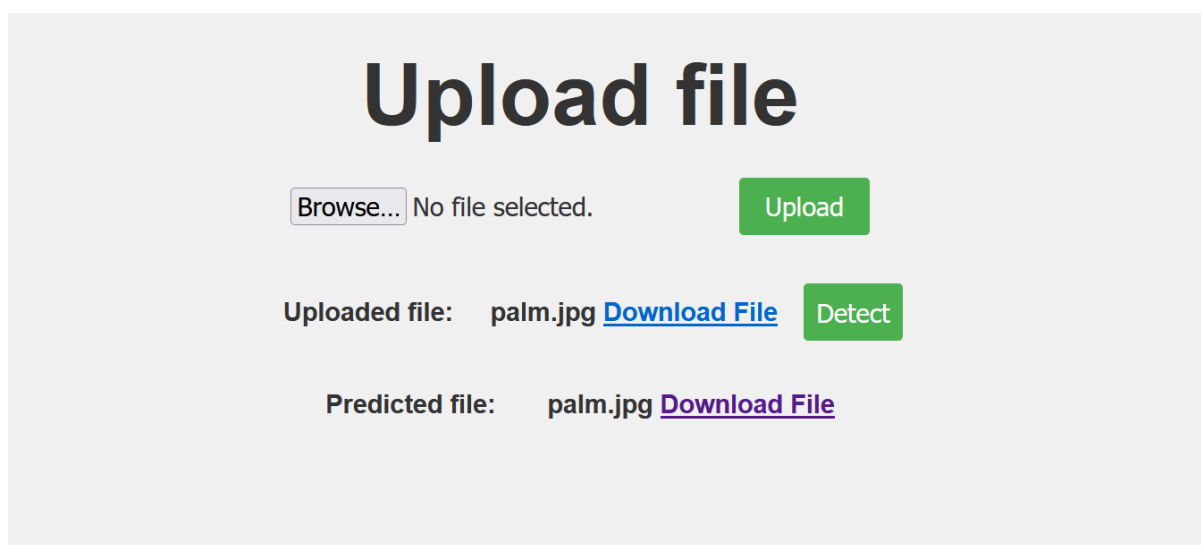


Slika 5.2 Sučelje za pokretanje web kamere kao izvor za detekciju dlana

Učitavanje vlastitih slika i videa funkcionira tako da se korisniku da mogućnost odabira datoteke na kojoj želi obaviti procjenu, točnije nad kojom želi provest proces detekcije i praćenja dlana. Formati datoteke na kojom se može provesti procjena su .jpg, .png i .mp4.

Nakon što korisnik učitava datoteku, on ima mogućnost tu istu preuzeti na vlastito računalo kako bi se moglo provjeriti je li točna datoteka učitana. Zatim korisnik pritiskom na gumb za predviđanje pokreće *python* skriptu *upload.py* koja se pokreće u pozadini i za zadanu datoteku izvodi proces predviđanja, prepoznavanja i praćenja dlana.

Pri završetku korisnik ima mogućnost preuzeti novu datoteku nad kojom je izvedeno predviđanje. Slika ili video bit će jednaka onoj učitanoj gdje će se detektirani dlan staviti unutar graničnog okvira i ispisati razred kojem prepoznati objekt pripada sa kojom vjerojatnošću. Izgled stranice nakon što je korisnik učitao datoteku i proveo predviđanje prikazan je na slici **Slika 5.3**.



The screenshot displays a web interface titled "Upload file". It features a file selection area with a "Browse..." button and the text "No file selected.", alongside a green "Upload" button. Below this, the "Uploaded file:" section shows "palm.jpg" with a blue "Download File" link and a green "Detect" button. The "Predicted file:" section also shows "palm.jpg" with a purple "Download File" link.

Slika 5.3 Stranica za učitavanje lokalne datoteke nakon što je provedena detekcija

6 Eksperimenti i rezultati

7 Zaključak

8 Literatura

9 Sažetak

10 Summary