UNIVERZITET U NIŠU
ELEKTRONSKI FAKULTET

SISTEM ZA PREPOZNAVANJE GESOVA ŠAKE ZASNOVAN NA DUBOKOM UČENJU

Tehički izveštaj

Predmet:
Duboko učenje

Studenti:

Ana Tonić, br. ind. 1566
Nikola Đorđević, br. ind. 1567

Mentor:

Prof. dr Aleksandar Milosavljević

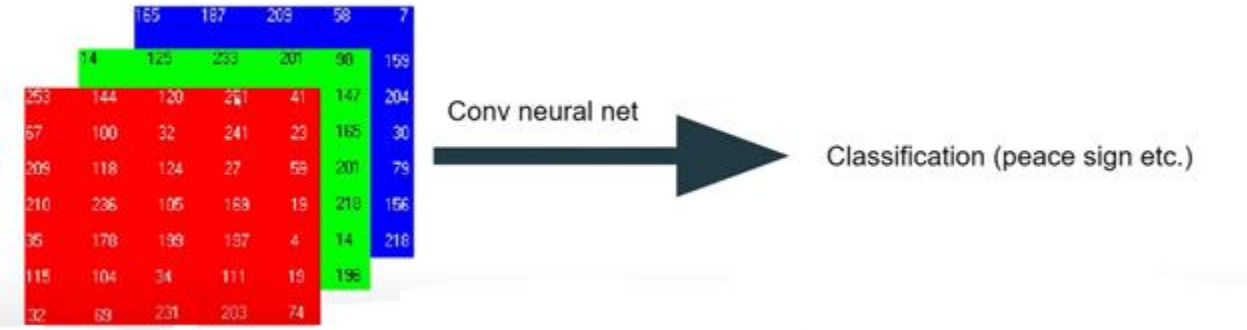
Niš, jun 2023. godine

Sadržaj

O aplikaciji

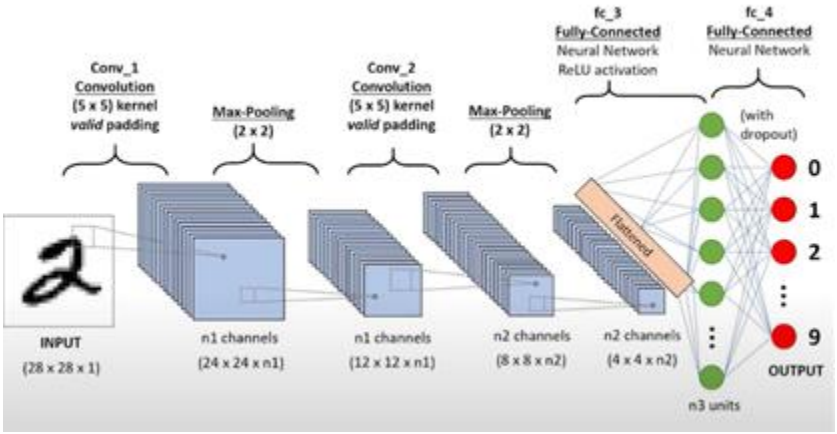
Problemi i izazovi prilikom izgradnje sistema za prepoznavanje gestova: šake mogu biti različitih oblika, gestovi mogu biti različitih oblika, pozadina može da se razlikuje, osvetljenje može da varira.

Postoji više različitih pristupa: imamo pixel first approach. U ovom pristupu se koriste RGB matrice. Sve slike u računaru su predstavljene kao RGB matrice.



Slika 1 Pixel first approach

Te matrice se prosleđuju kroz konvolucione neuronske mreže, koje predstavljaju tip neuronskih mreža optimizovan za rad sa slikama, pomoću njih se vrši klasifikacija.

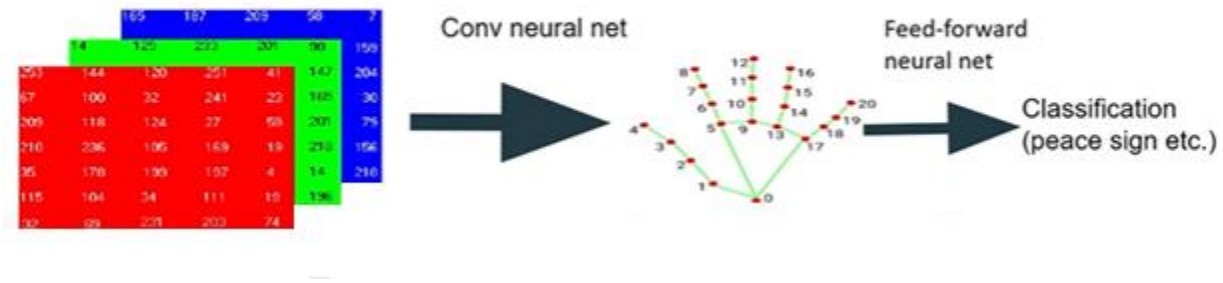


Slika 2 Arhitektura komplikovane konvolucione neuronske mreže

Za treniranje ovakvog sistema, potrebno je prikupiti mnogo podataka. Potrebno je prikupiti slike šaka koje se nalaze na različitim tipovima pozadina.

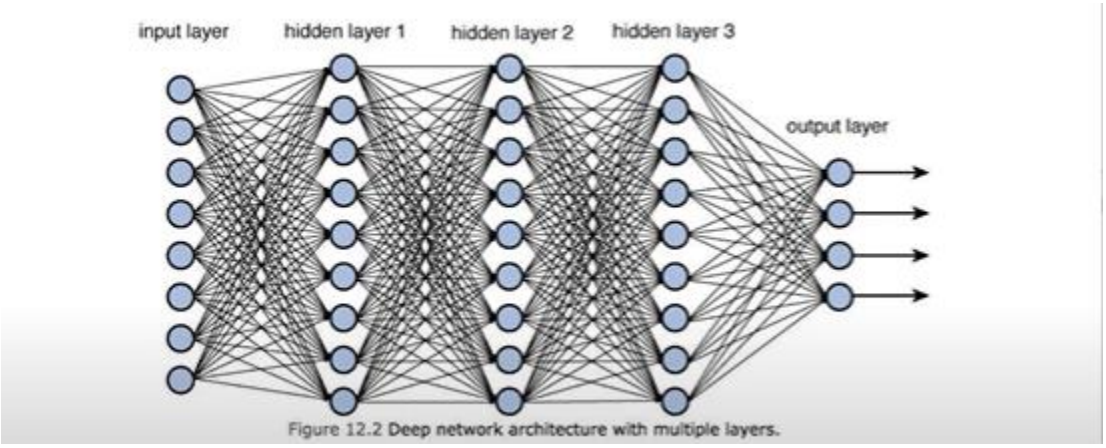
Drugi pristup, koji ima više slojeva nego prvi, sličan prvom pristupu:

Uzima se frame sa kamere i prosleđuje se kroz konvolucionu neuronsku mrežu, ali ova neuronska mreža je drugačija. Ova neuronska mreža ne daje odgovor na to koji gest je prikazan u frame-u, već izvlači ključne tačke.



Slika 3 Grafički prikaz pristupa sa korišćenjem ključnih tačaka

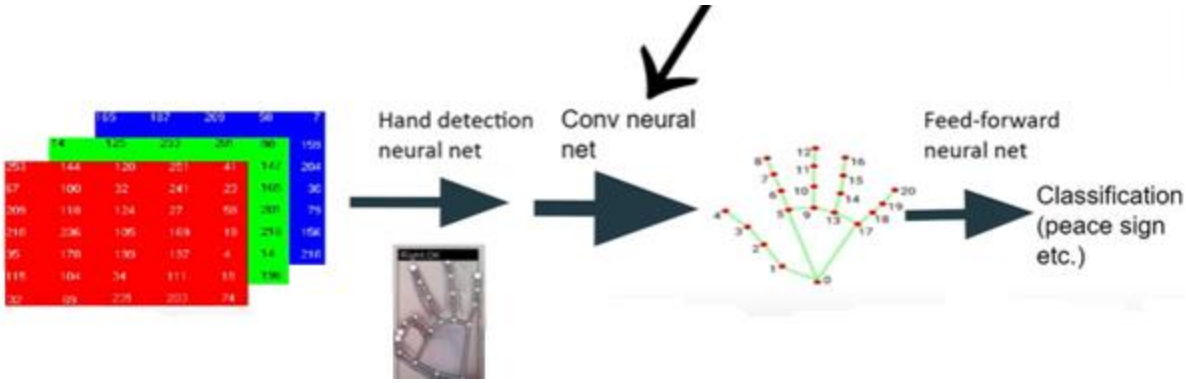
Onda se te ključne tačke prosleđuju kroz mnogo manju i mnogo jednostavniju neuronsku mrežu – feed forward arhitekture.



Slika 4 Arhitektura jednostavne konvolucione neuronske mreže

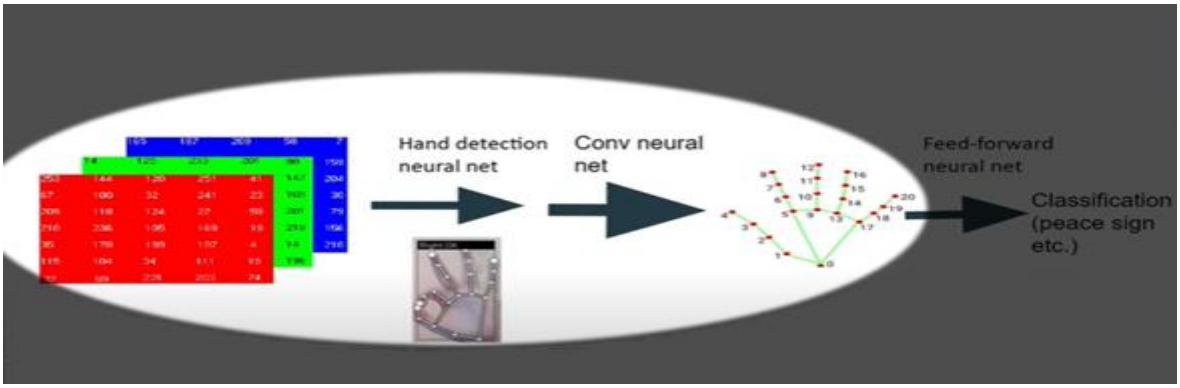
Za prvi deo zadatka, izdvajanje ključnih tačaka, koristićemo gotovu biblioteku. Drugi deo zadatka je treniranje jednostavne neuronske mreže, gde ne moramo da brinemo o dostavljanju slika sa različitom pozadinom. Dovoljno je da dostavimo slike na kojima je gest koji želimo da pokažemo, sa različitom orijentacijom.

Konvolucionoj neuronskoj mreži za ekstraktovanje ključnih tačaka nije potrebno dostaviti celu sliku, već samo kropovan deo slike na kojoj se vidi samo šaka.



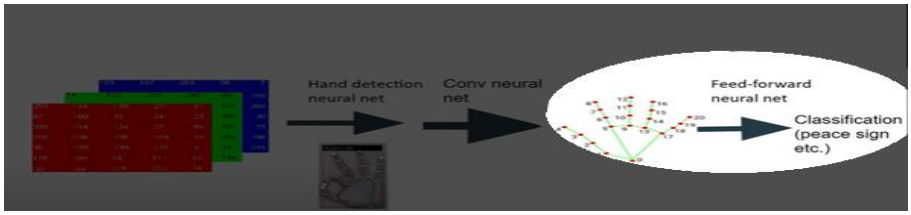
Slika 5 Detaljni prikaz pristupa sa korišćenjem ključnih tačaka

Na sledećoj slici je prikazan deo koji je zaista dobro handlovan google mediaPipe bibliotekom:



Slika 6 Deo arhitekture koji je pokriven MediaPipe bibliotekom

Na sledećoj slici je prikazan deo koji mi radimo u svom projektu:



Slika 7 Deo arhitekture koji mi radimo u svom projektu

Mediapipe

MediaPipe je besplatan frejmvork(okvir), otvorenog koda, koji nudi višeplatformska, prilagodljiva rešenja zasnovana na mašinskom učenju za obradu video snimaka i slika. Pruža različita rešenja za detekciju ključnih tačaka na telu ili detekciju objekata [1].

MediaPipe je u stanju da postigne svoju brzinu zahvaljujući korišćenju ubrzanja na grafičkom procesoru i multi tredingu. Takve tehnike razvoja su generalno teške za izvođenje, ali MediaPipe to odrađuje za nas. Multitreding i GPU akceleracija dozvoljava novim mobilnim telefonima da koriste MediaPipe. Podrška za više različitih platforma može biti veliki zadatak za male development razvojne timove. To se odnosi za podršku za Vindovs, Mac i različite distribucije linuks-a. MediaPipe omogućava lak diplojment na različite platforme.

U okviru MediaPipe-a postoji rešenje za detekciju lica na slici (Face Detection), detekciju 468 3D ključnih tačaka lica u Face Mesh rešenju. Zatim detekciju zenica (Iris), detekciju šaka (Hands), detekciju delova tela (Pose), koja uključuje i neke ključne tačke na licu i rukama. Rešenje koje obuhvata detekciju ključnih tačaka lica, šaka i tela istovremeno se naziva MediaPipeHolistic. Pored ovih rešenja za ljudsko telo, MediaPipe ima i rešenja za detekciju objekata, kao što su Object Detection, Box Tracking, Instant Motion Tracking, Objectron i KNIFT.

Sposobnost opažanja oblika i pokreta ruku može biti neophodna komponenta u poboljšanju korisničkog iskustva u različitim tehnološkim domenima i platformama. Na primer, može da predstavlja osnovu za razumevanje znakovnog jezika i za kontrolu pomoću gestova ruke. Takođe, može da se koristi za integrisanje digitalnog sadržaja i informacija na fizički svet u proširenoj realnosti. Percepcija gestova šake je prirodna za ljude, ali zbog toga što određeni gestovi zaklanjaju svoje delove(pesnica) ili šake međusobno zaklanjaju jedna drugu(rukovanje) i zbog postojanja malog broja jedinstvenih gestova, precizna percepcija gestova šake u realnom vremenu je izazovan zadatak računarskog vida.

MediaPipe hands je rešenje visoke preciznosti za praćenje dlana i prstiju. Koristi mašinsko učenje da zaključi 21 3D ključnu tačku šake iz samo jednog frame-a. Dok se trenutni najsavremeniji pristupi oslanjaju na desktop računare kako bi došli do zaključivanja, ovaj sistem postiže performanse u realnom vremenu na mobilnom telefonu, pa čak i za više od jedne ruke. Cilj pravljenja ovog sistema jeste da posluži kao osnova za razvoj aplikacija računarskog vida.

Nakon detekcije dlana iz cele slike, model za detekciju ključnih tačaka šake vrši preciznu lokalizaciju koordinata svih 21 3D ključnih tačaka šake unutar otkrivenog regiona šake, putem regresije, odnosno direktnog predviđanja koordinata. Model uči konzistentnu unutrašnju reprezentaciju poze ruke i nije sklon greškama čak i u slučaju delimično vidljive šake i okluzije.

Da bi dobili osnovni skup podataka na kome su tačke tačno određene, ručno su označili približno 30 hiljada slika iz stvarnog sveta sa 21 3D koordinatom, kao što je prikazano na slici. Da bi pokrili sve moguće poze ruku i obezbedili dodatno nadgledanje prilikom učenja, koristili su renderovane slike visokog kvaliteta sintetičkog modela šake preko puno različitih pozadina i te sintetičke slike su obeležili 3D koordinatama.

MediaPipe biblioteka radi efikasno detektovanje ključnih tačaka ako joj se kao ulaz proslede slike ali i ako joj se kao ulaz prosledi video. U slučaju da se kao ulaz prosledi video najbolje je STATIC_IMAGE_MODE ostaviti na podrazumevanu vrednost koja je false, zbog toga što se u tom slučaju ne pokreće pretraga lokacije šake nakon što se po prvi put uspešno detektuje. Jednom kada se detektuje maksimalni broj šaka koji je određen podešavanjem i kada se lokalizuju sve ključne tačke, jednostavno se prate te utvrđene ključne tačke, bez pozivanja nove detekcije, sve dok se ne izgube ključne tačke za bar jednu od šaka. Na ovaj način se omogućava redukcija kašnjenja i sistem je pogodan za procesiranje video frejmova. Drugi način rada jeste da se detekcija šake radi na svakoj slici, što je idealno za procesiranje skupa, potencijalno nepovezanih slika. Sistem podrazumevano radi tako što ne pokreće sistem za detektovanje šake sem kad je to nužno, kao što je opisano.

Izlaz

Ključne tačke koje predstavljaju izlaz iz protočne obrade ove biblioteke su predstavljene u vidu kolekcije gde je svaka šaka predstavljena listom od 21-og elementa. Svaki element je ključna tačka koja se sastoji od x, y i z koordinate. X i y koordinata su normalizovane na opseg od 0.0 do 1.0 u skladu sa širinom i visinom prozora. Z predstavlja dubinu ključne tačke, s tim što je tačka koja predstavlja zglob osnova dubine. Što je vrednost manja, to je ključna tačka bliža kameri, a što je veća, to je ključna tačka dalja od kamere.

Ovaj sistem kao izlaz daje i listu 21 tačke u svetskim koordinatama. Svaka ključna tačka se sastoji od x, y, i z koordinata u metrima sa osnovom u približnoj geometrijskoj sredini šake.

Za svaku šaku koja je detektovana dobijamo informaciju da li je detektovana šaka leva ili desna i procenat sigurnosti sistema u tu detekciju. Sistem radi sa pretpostavkom da je slika izokrenuta.

OpenCV

OpenCV (Open Source Computer Vision Library) je otvorena biblioteka računarskog vida koja pruža širok spektar algoritama i funkcija za obradu slika i video zapisa. Ova biblioteka je napisana u programskom jeziku C++ i podržava različite platforme kao što su Windows, Linux, Mac OS, Android i iOS. OpenCV je postao standardni alat za razvoj aplikacija u oblasti računarskog vida zbog svoje efikasnosti, fleksibilnosti i jednostavnosti upotrebe [4].

OpenCV pruža mnoge mogućnosti za obradu slika, uključujući osnovne operacije poput učitavanja i čuvanja slika, promene veličine, rotacije i izrezivanja slika. Takođe podržava napredne tehnike obrade slika kao što su detekcija ivica, filtriranje, segmentacija i prepoznavanje oblika. OpenCV takođe pruža podršku za rad sa video zapisima, omogućavajući snimanje, reprodukciju i obradu video sadržaja.

Osim toga, OpenCV je popularan zbog svoje sposobnosti za detekciju i praćenje objekata u stvarnom vremenu. Može se koristiti za detekciju lica, prepoznavanje gestova, pratiti objekte u videu i još mnogo toga. Pored toga, OpenCV ima i podršku za mašinsko učenje, što omogućava primenu tehnika dubokog učenja za različite zadatke računarskog vida.

Ukratko, OpenCV je moćna i fleksibilna biblioteka za obradu slika i video zapisa, koja pruža razne algoritme i funkcije za različite zadatke računarskog vida.

Keypoint generator

Aplikacija za generisanje ključnih tačaka je implementirana uz pomoć biblioteka OpenCV i MediaPipe.

Biblioteka OpenCV (Open Source Computer Vision Library) se koristi za rad sa slikama i video zapisima. U aplikaciji, biblioteka OpenCV se koristi za snimanje videa sa kamere, prikazivanje slike i manipulaciju slikama. Takođe se koristi za konverziju boja slika (BGR u RGB format) i procesiranje pritisnutih tastera.

Biblioteka MediaPipe se koristi za obradu slika i video zapisa, tj. izvlačenje ključnih tačaka. U aplikaciji, biblioteka MediaPipe se koristi za prepoznavanje ključnih tačaka (landmark) na rukama. Pomoću mp.solutions.hands modula iz biblioteke MediaPipe inicijalizuje se model za prepoznavanje ruku i vrši se detekcija šake na svakom frejmu kao i detekcija ključnih tačaka u okviru detektovane šake.

Što se same aplikacije tiče, nakon uključivanja potrebnih modula, vrši se parsiranje agrumenata komandne linije i tako dobijene vrednosti se dodeljuju željenim promenljivama (agrumenti poput identifikatora kamera, širine i visine slike, režim slike itd). Priprema se kamera za snimanje video zapisa koristeći cv.VideoCapture funkcije i postavljaju se širina i visina snimka koristeći cap.set(). Nakon toga se učitava se model za prepoznavanje ruku iz biblioteke MediaPipe i konfigurišu se parametri, kao što su static_image_mode, max_num_hands, min_detection_confidence i min_tracking_confidence. Nakon toga se u beskonačnoj while petlji proverava pritisnut taster i izvršava se odgovarajuća akcija. Snima se slika sa kamere koristeći cap.read() metode. Snimljena slika se zatim prebacuje u RGB format i prosleđuje se modelu detekciju šake. Detekcija ključnih tačaka ruke se vrši pomoću metode hand_landmarks. Ova metoda vraća normalizovane koordinate ključnih tačaka ruke, gde (0, 0) predstavlja gornji levi ugao slike, a (1, 1) predstavlja donji desni ugao slike. Funkcija calc_landmark_list() uzima rezultat detekcije šake i konvertuje ključne tačke u piksel koordinate slike. To se postiže množenjem normalizovanih koordinata sa širinom i visinom slike. Na taj način se dobijaju tačne koordinate ključnih tačaka u pikselima. Nakon toga se obavlja konverzija u relativne i normalizovane koordinate pomoću funkcije pre_process_landmark(). Ova funkcija vrši tranfsormisanje ključnih tačaka tako da što centar prve detektovane tačke postavi na koordinatni početak (0, 0), zatim se koorditane svih detektovanih tačaka računaju u ondosu na taj novi koordinatni početak, i na kraju se vrši normalizacija istih.

Treniranje modela

Podešvanje optimizer-a, broja epoha, slojeva i broja neurona

Isprobani su različiti optimizer-i ('sgd', 'rmsprop', 'adam', 'adagrad', 'adadelta', 'nadam'). Vrednosti za tačnost koje su dobijene su:

- 'sgd' – 0.8625
- 'rmsprop' – 0.9125
- 'adam' – 0.9125
- 'adagrad' – 0.8
- 'adadelta' – 0.35
- 'nadam' – 0.9125

Najbolje rezultate su dali optimizeri adam, rmsprop i nadam. Kao optimizer u daljim istraživanjima je uzet optimizer adam, zbog toga što daje najbolje rezultate i zbog toga što se ovaj optimizer najčešće koristi.

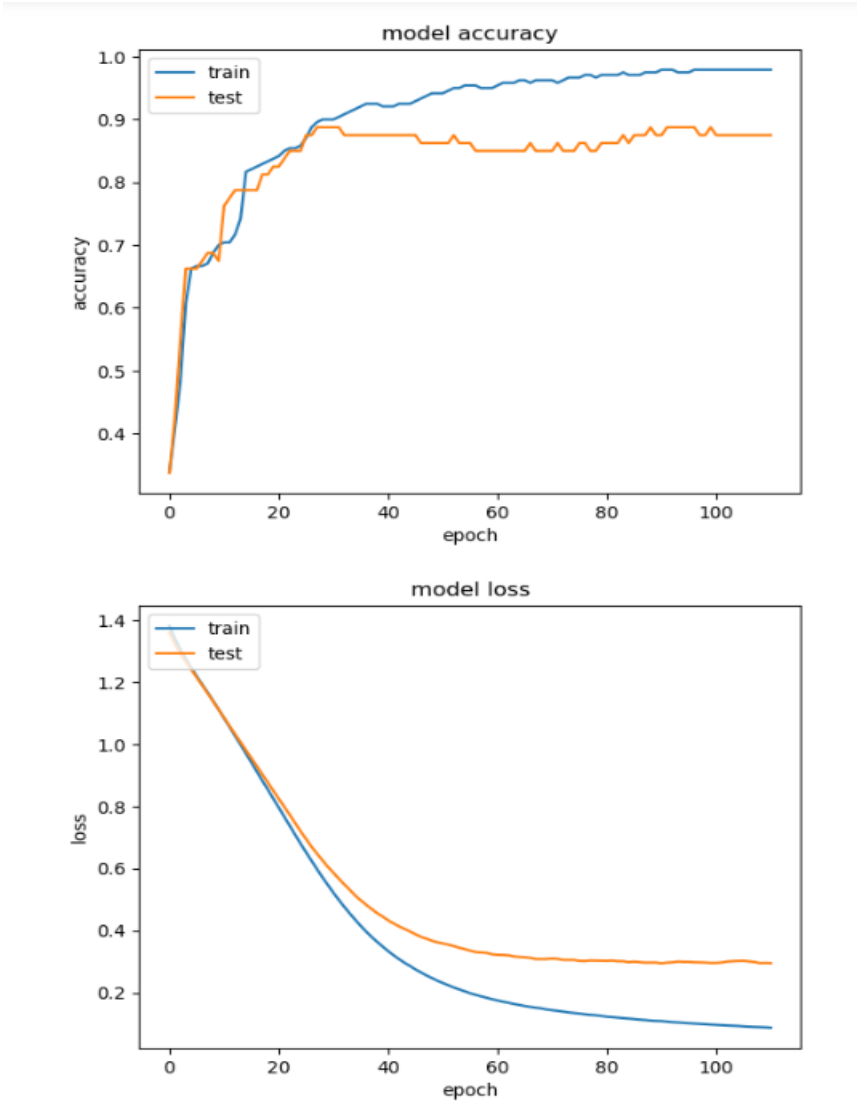
Isprobane su različite vrednosti vrednosti za broj epoha. Očekivano ponašanje modela je bilo poboljšanje i povećanje tačnosti sa povećanjem broja epoha. Vrednosti za tačnost, koje su dobijene za različih broj epoha, su:

- 100 – 0.9
- 200 – 0.9125
- 250 – 0.9125
- 500 – 0.9125
- 1000 – 0.9125
- 5000 – 0.9125

Može se uočiti da se povećanjem broja epoha sa 100 na 200 tačnost povećava, dok se daljim povećavanjem broja epoha ne postiže poboljšanje modela. Zbog toga je kao optimalan broj epoha za dalja istraživanja uzet broj 200.

Dalja ispitivanja su vršena dodavanjem različitog broja slojeva. Očekivano ponašanje je da se dodavanjem novih slojeva model poboljša. Najpre je dodat još jedan potpuno pozvezani sloj, sa 150 neurona i ReLU aktivationom funkcijom. Tačnost do koje se došlo primenom ovakvog modela je 0.9375. Zatim je dodat još jedan ovakav sloj i u tom slučaju je dobijena tačnost 0.925. Na kraju je isproban model sa 4 ovakva sloja i tačnost je i u tom slučaju ostala na 0.925. Kao osnova za dalja istraživanja je zbog postignute tačnosti uzet model koji ima 2 potpuno povezana skrivena sloja, sa 150 neurona i ReLU aktivacionom funkcijom.

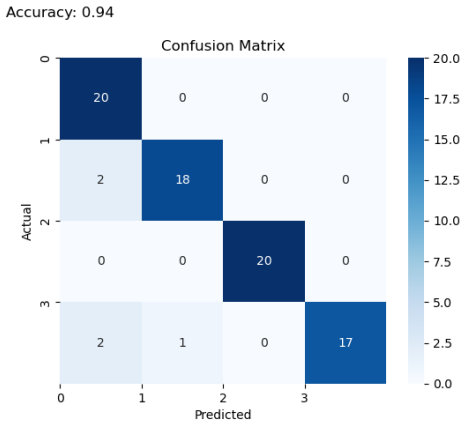
Sledeća istraživanja su vršena manipulacijom broja neurona. Što se broja neurona tiče, praćene su preporuke iz knjige „Learn Keras for Deep Neural Networks“ autora Jojo Moolayil-a. Isprobani su sledeći brojevi neruona u svakom od slojeva 8, 16, 32, 64, 128, 256, 512 [2]. Cilj manipulacije brojem neurona je pokušaj da se sa manjim brojem neurona dostigne ista tačnost, ili da se sa većim brojem neurona dostigne još bolje tačnost. Očekivano je da najbolje rezultate daje broj neruona koji je najpribližniji broju ulaznih vrednosti pomnoženom sa 2, a da pritom bude stepen broja 2 (uovom slučaju je to broj 64). Očekivan rezultat je i dobijen eksperimentisanjem sa brojem neruona, i najbolju tačnost od 0.95 su dali 64 neurona u prvom i isto toliko neurona u drugom skrivenom sloju. Dakle, nakon podešavanja optimizer-a, broja epoha, slojeva i broja neurona najbolji rezultat je dao model sa dva skrivena sloja, pri čemu je broj neurona u oba sloja 64 i oba sloja imaju ReLU aktivacionu funkciju. Korišćenjem ovog modela je tačnost 0.95 i grafici koji su dobijeni (tačnost i loss funkcija) su prikazani na sledećoj slici.

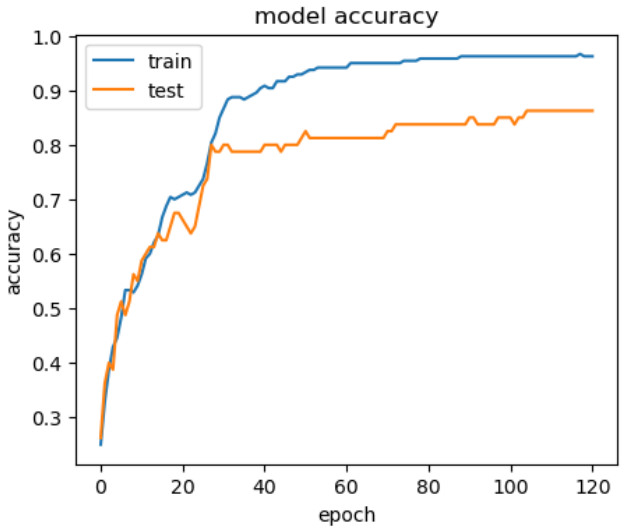


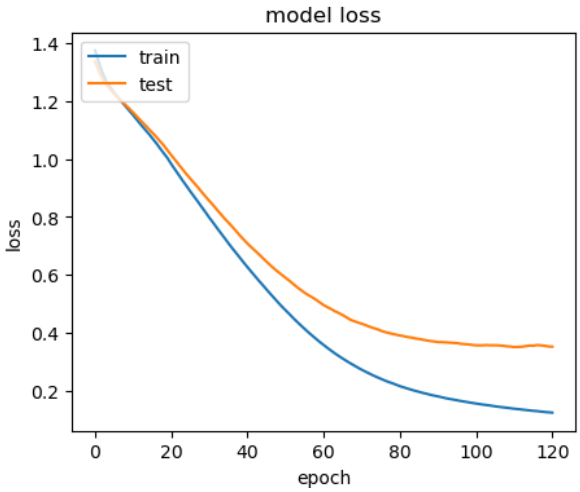
Aktivacione funkcije

Za skrivene slojeve u višeklasnoj klasifikaciji često se preporučuju aktivacione funkcije poput ReLU (Rectified Linear Unit), Leaky ReLU, ili njihovih varijanti. Ove aktivacione funkcije su popularne zbog svoje sposobnosti da modeluju nelinearne veze između ulaza i izlaza, kao i zbog svoje jednostavnosti i efikasnosti u računanju.

Tokom izrade projekta, vršili smo eksperimentisanje koristeći više različitih aktivacionih funkcija za skrivene slojeve tanh, sigmoid, softmax, relu, pritom očekujući da će najbolji model biti onaj koji koristi relu funkciju za skrivene slojeve. Rezultati se slažu sa očekivanjima. Funkcije su testirane nad mrežom sa dva skrivena sloja od po 100 i 19 neurona, sa 200 epoha i primenjenom early stopping metodom za sprečavanje overfitting-a. Rezultati su sledeći:





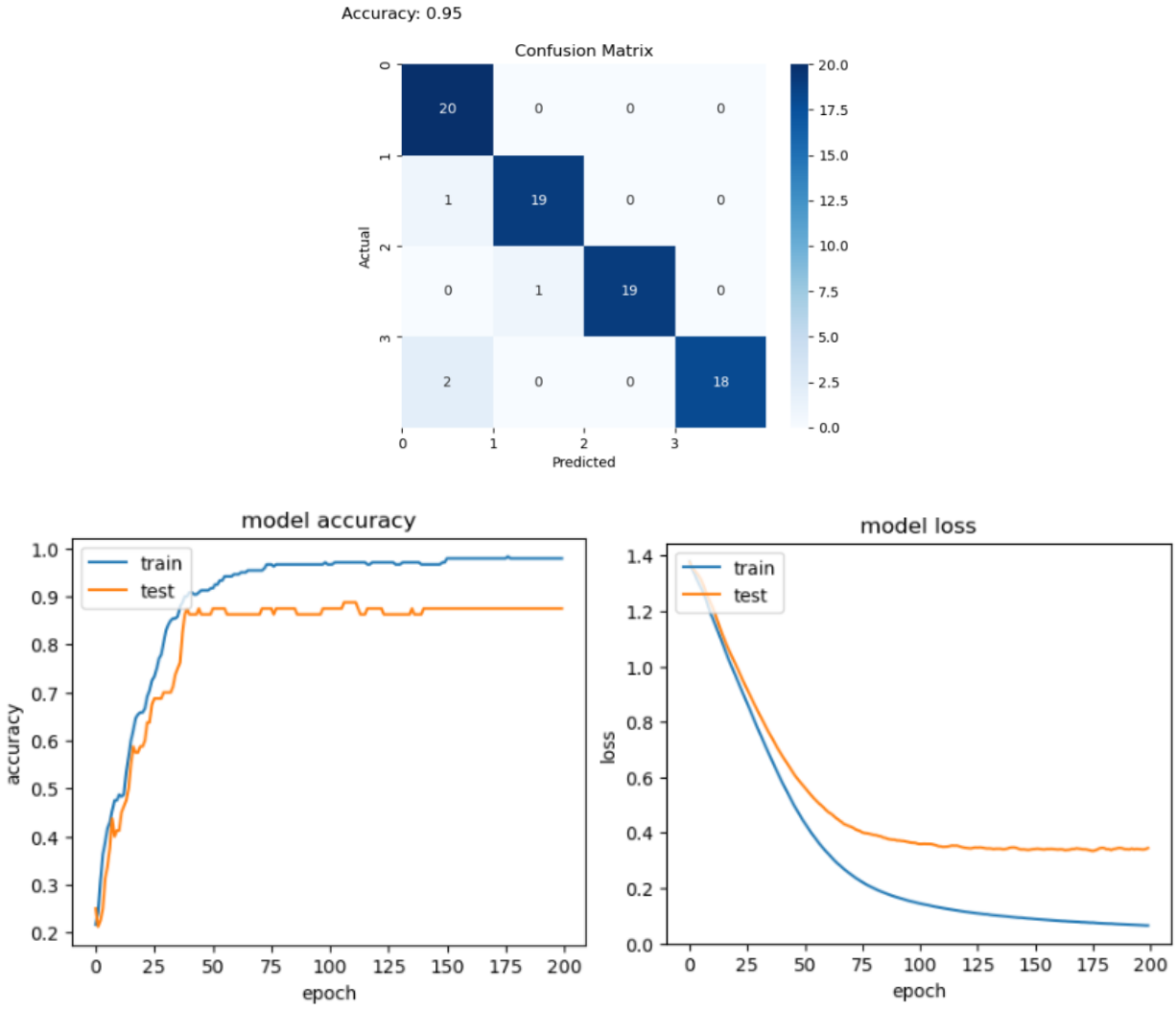


Za izlazni sloj u višeklasnoj klasifikaciji, često se koristi aktivaciona funkcija Softmax. Softmax funkcija se primenjuje na izlazu neuronske mreže i transformiše rezultate u vrednosti koje predstavljaju verovatnoće za svaku klasu S obzirom na to da se u našem slučaju radi o

višeklasnoj klasifikaciji, u svakom test modelu koristili smo ovu aktivacionu funkciju za izlazni sloj.

Podešavanje stope učenja

U dubokom učenju, tehnike za podešavanje stope učenja se koriste kako bi se optimizovalo obučavanje neuronskih mreža. Stope učenja određuju koliko brzo se parametri mreže ažuriraju tokom procesa obučavanja. U našem projektu smo isprobali sve tehnike za podešavanje stope učenja, a to su: Reduce Learning Rate on Plateau i Exponential learning rate scheduler. Exponential learning rate scheduler daje vrlo nisku tačnost, čak goru nego za nasumičan klasifikator. Reduce Learning Rate on Plateau je tehnika koja je dala dobre rezultate i poboljšala tačnost modela. Upotrebom ove tehnike uz korišćenje mreže sa dva sloja sa 100 i 10 neurona respektivno, treniranjem modela kroz 200 iteracija postigli smo tačnost od 95%.



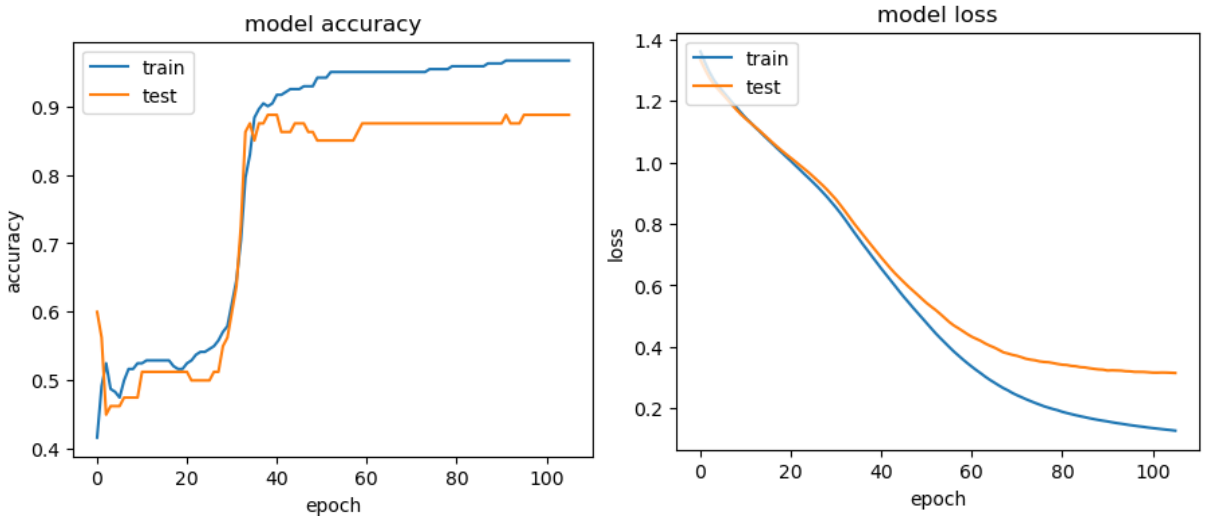
Rešavanje overfitting-a

U procesu razvoja i treniranja modela mašinskog i dubokog učenja, često nailazimo na scenario gde trenirani model daje dobre performanse nad trening podacima ali ponaša se lošije nad test skupom podataka. U nauci o podacima, ova pojava se zove „overfitting“. U bukvalnom smislu, naš model se uklapa u podatke.

Proces treniranja modela se naziva „uklapanje podataka“(„fitting the data“). Neuronska mreža uči skrivene obrasce među podacima i matematički poboljšava težine i strukturu modela da bi se model uklopio u obrasce koje je mreža otkrila u toku procesa učenja. Ukratko, obukom modela se podešava njegova struktura (težine) da bi se uklopio u podatke (obrasce) i na taj način se poboljšavaju njegove performanse. Ovaj proces postaje komplikovan u trenutku kada se uspostavi da je obrazac koji je model pronašao zapravo samo šum u podacima. Na žalost, matematička jednačina ne može uvek da razlikuje signal od šuma (pod šumom se podrazumeva uzorak koji ne predstavlja uzorak za obuku, već nastaje slučajno.) Kada ne uspe, model takođe nauči šum i podesi svoju težinu tako da se prilagodi novom signalu, koji je zapravo šum [4].

Kako se može videti na prethodnoj slici na kojoj su prikazane krive učenja za model koji nam naje najbolje rezultate, tačnost počinje da stagnira posle 100-te iteracije iako se vrednost loss funkcije i dalje blago smanjuje. To ukazuje da je došlo do overfitting-a i da je potrebno upotrebiti neku od tehnika za rešavanje overfitting-a kao što su Early stopping i tehnike regularizacije L1 i L2.

Upotrebom tehnike early stopping, treniranje se zaustavlja ubrzo posle 100-te iteracije ali se tačnost klasifikatora smanjuje na 91%.



Upotrebom dropout tehnike i early stopping zajedno dovodi do iste tačnosti od 91%. Nakon toga smo eksperimentisali sa tehnikama za regularizaciju.

U L1 regularizaciji, apsolutne težine se dodaju loss funkciji. Da bi se model učinio generalnijim, vrednosti težina su smanjene na 0, i stoga je ovaj model veoma poželjan kada pokušavamo da komprimujemo model radi bržeg izračunavanja. Ovaj model je veoma pogodan kada pokušavamo da smanjimo složenost modela radi bržeg izračunavanja [5]. Jednačina se može predstaviti kao (Slika 8):

$$\text{Cost Function} = \text{Loss (as defined)} + \frac{\lambda}{2m} * \sum \|Weights\|$$

Slika 8 Formula za L1 regularizaciju

Regularizacija L1 značajno smanjuje tačnost klasifikacije, za oko 10%, tako da nismo nastavili dalja eksperimentisanja korišćenjem ove tehnike.

U L2 regularizaciji, kvadratne vrednosti težina se dodaju loss funkciji. Da bi se model učinio generalnijim, vrednosti težina su smanjene na vrednosti blizu 0 (ali ne zapravo 0), pa se ovo naziva i metodom opadanja težine. U većini slučajeva, L2 se preporučuje pre nego L1, u cilju smanjenja overfitting-a [5]. Jednačina može biti predstavljena na sledeći način (Slika 9):

$$\text{Cost Function} = \text{Loss (as defined)} + \frac{\lambda}{2m} * \|Weights\|^2$$

Slika 9 Jednačina L1 regularizacije

Korišćenjem regularizacije L2, samostalno i u kombinaciji sa mrežom koja koristi dinamičku promenu learning rate-a, kao i sa mrežom u koju su dodati dropout slojevi postigli smo tačnost istreniranog modela iznad 90%, ali to je ipak niže u odnosu na model bez korišćenja tehnika za rešavanje overfitting-a, koji ima preciznost od 95%.

Aplikacija za prepoznavanje gestova

Dati kod implementira sistem za prepoznavanje gestova šake koristeći MediaPipe biblioteku i TensorFlow. Sistem snima video kadrove sa kamere, obrađuje ih koristeći MediaPipe Hands modul radi detekcije i praćenja ključnih tačaka šake, i vrši klasifikaciju gestova šake koristeći unapred obučen model TensorFlow.

Kod počinje sa uvozom neophodnih biblioteka i definisanjem argumenata komandne linije korišćenjem modula "argparse". Argumenti definišu indeks uređaja, širinu i visinu kamere, pragove pouzdanosti za detekciju i praćenje, kao i druge podešavanja. Funkcija "get_args()" parsira argumente komandne linije i vraća ih kao objekat. Zatim, kod učitava unapred obučen TensorFlow model za klasifikaciju gestova ruke korišćenjem funkcije "tf.keras.models.load_model()". Putanja do modela se navodi u promenljivoj "model_save_path".

Funkcija "main()" je ulazna tačka programa. Ona počinje inicijalizacijom kamere korišćenjem OpenCV biblioteke i podešavanjem svojstava kamere prema navedenim argumentima. MediaPipe Hands modul se instancira kao "mp_hands. Hands" sa navedenom konfiguracijom, uključujući da li se koristi režim statičke slike, maksimalni broj detektovanih ruku i pragove pouzdanosti za detekciju i praćenje. Kod zatim učitava oznake za klasifikaciju gestova ruke iz CSV fajla koristeći funkciju "csv.reader()" i čuva ih u listi "keypoint_classifier_labels". Unutar glavne petlje, program obrađuje unos korisnika i prekida petlju ako je pritisnuto dugme ESC.

Trenutni kadar se snima sa kamere, i ako snimanje bude uspešno, kadar se horizontalno okreće kako bi se dobio ogledalo prikaza. Poziva se funkcija "hands.process()" za detekciju i praćenje ključnih tačaka ruke na snimku, a rezultati se čuvaju u promenljivoj "results". Ako su ključne tačke ruke detektovane na snimku, kod prolazi kroz detektovane tačke i njihove odgovarajuće ruke (leva ili desna). Poziva se funkcija "calc_landmark_list()" za izračunavanje koordinata tačaka relativno u odnosu na veličinu slike, a funkcija "pre_process_landmark()" se koristi za predobradu tačaka normalizacijom njihovih koordinata. Predobrađene tačke se zatim prosleđuju učitanom TensorFlow modelu za klasifikaciju gestova ruke. Predviđeni ID gesta se dobija pomoću funkcije "np.argmax()", a odgovarajuća oznaka se izvlači iz liste "keypoint_classifier_labels". Oznaka gesta se prikazuje na slici pomoću funkcije "cv.putText()" iz OpenCV biblioteke. Na kraju, obrađeni kadar se prikazuje u prozoru nazvanom "Prepoznavanje gestova ruke". Program nastavlja sa snimanjem i obradom kadrova sve dok se ne pritisne dugme ESC. Nakon toga, snimanje sa kamere se završava, a svi prozori iz OpenCV se zatvaraju.

Dati kod demonstrira kompletan tok za prepoznavanje gestova ruke, od snimanja kadrova do izvršavanja klasifikacije pomoću unapred obučenog modela. Demonstracija rada aplikacije je data u nastavku.

Literatura

[1] MediaPipe library, [Posetili ste ovu stranicu 1.6.23], Dostupno na: https://developers.google.com/mediapipe/

[2] Learn Keras for Deep Neural Networks, Jojo Moolayil

[3] L1 i L2 regularizacija, [Na mreži], Dostupno na: https://neptune.ai/blog/fighting-overfitting-with-l1-or-l2-regularization

[4] OpenCV library, [Posetili ste ovu stranicu 6.6.23], Dostupno na: https://docs.opencv.org/4.x/

[5] Tehnike za prevenciju overfitting-a, [Na mreži], Dostupno na: https://towardsdatascience.com/8-simple-techniques-to-prevent-overfitting-4d443da2ef7d