Umjetna inteligencija u procjeni položaja tijela

Novaković, Lora

SVEUČILIŠTE U SPLITU
PRIRODOSLOVNO MATEMATIČKI FAKULTET

ZAVRŠNI RAD

UMJETNA INTELIGENCIJA U PROCJENI POLOŽAJA TIJELA

Lora Novaković

Split, srpanj 2019.

ZAHVALA

Ponajprije se zahvaljujem svom mentoru prof.dr.sc. Saši Mladenoviću koji me svojim stručnim savjetima vodio kroz ovaj završni rad. Puno hvala na strpljivosti, pristupačnosti i potpori tokom izrade ovog završnog rada.

Veliku zahvalu dugujem obitelji, osobito svojim roditeljima, ocu Mirku i majci Sanji, koji su mi omogućili studiranje i nisu sumnjali u mene u niti jednom trenutku. Također se želim posebno zahvaliti baki Jasminki i djedu Mladenu koji su mi izuzetno velika potpora u životu.

IZJAVA

kojom izjavljujem s punom materijalnom i moralnom odgovornošću da sam završni rad s naslovom *Umjetna inteligencija u procjeni položaja tijela* izradila samostalno pod voditeljstvom dr.sc. Saše Mladenovića. U radu sam primijenila metodologiju znanstvenoistraživačkog rada i koristila literaturu koja je navedena na kraju diplomskog rada. Tuđe spoznaje, stavove, zaključke, teorije i zakonitosti koje sam izravno ili parafrazirajući navela u diplomskom radu na uobičajen, standardan način citirala sam i povezala s fusnotama s korištenim bibliografskim jedinicama. Rad je pisan u duhu hrvatskog jezika.

Studentica

Lora Novaković

Temeljna dokumentacijska kartica

Završni rad

Sveučilište u Splitu

Prirodoslovno-matematički fakultet

Odjel za informatiku

Ruđera Boškovića 33, 21000 Split, Hrvatska

UMJETNA INTELIGENCIJA U PROCJENI POLOŽAJA TIJELA

Lora Novaković

SAŽETAK

U ovom radu se prikazuje način na koji računalo može detektirati naše pokrete putem web kamere. Spominje se koja gotova biblioteka je najjednostavnija za korištenje - *PoseNet*. Pobliže se objašnjava rad neuronskih mreža, s naglaskom na konvolucijske neuronske mreže, koje su se pokazale jako pouzdane kad je riječ o obradi slika. S obzirom da ne treniramo sami mrežu, nije objašnjen detaljno način učenja mreže.

Ključne riječi: *strojno učenje, neuronske mreže, konvolucijske neuronske mreže, PoseNet*

Rad je pohranjen u knjižnici Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu

Rad sadrži: 33 stranice, 24 grafičkih prikaza i 13 literaturnih navoda. Izvornik je na hrvatskom jeziku.

Mentor: **Dr. sc. Saša Mladenović,** *izvanredni profesor Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Neposredni voditelj:

Dr. sc. Goran Zaharija, *poslijedoktorand Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Ocjenjivači: **Dr. sc. Saša Mladenović,** *izvanredni profesor Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Dr. sc. Goran Zaharija, *poslijedoktorand Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Divna Krpan, predavač *Prirodoslovno-matematičkog fakulteta, Sveučilišta u Splitu*

Rad prihvaćen: 07, 2019.

Basic documentation card

Thesis

University of Split

Faculty of Science

Department of informatics

Ruđera Boškovića 33, 21000 Split, Croatia

ARTIFICIAL INTELLIGENCE IN POSE ESTIMATION

Lora Novaković

ABSTRACT

This paper shows how the computer can detect our movements via web camera. It mentions which library is the easiest to use. It is closely explained how neural networks work, with an emphasis on convolutional neural networks, which has proved to be very reliable when dealing with image processing. Since we are not training the network itself, we have not explained the detailed way of learning the network.

Key words: *machine learning, neural networks, convolutional neural networks, PoseNet*

Thesis deposited in library of Faculty of science, University of Split

Thesis consists of: 33 pages, 24 figures and 13 references.

Original language: Croatian

Mentor: **Saša Mladenović, Ph.D.** *Associate Professor of Faculty of Science, University of Split*

Supervisor: **Goran Zaharija, Ph.D.** *Postdoctoral Researcher of Faculty of Science, University of Split*

Reviewers: **Saša Mladenović, Ph.D.** *Associate Professor of Faculty of Science, University of Split*

Goran Zaharija, Ph.D. *Postdoctoral Researcher of Faculty of Science, University of Split*

Divna Krpan, *Lecturer of Faculty of Science, University of Split*

Thesis accepted: 07, 2019.

Sadržaj

UVOD

Umjetna inteligencija je dio računalne znanosti koji se bavi razvojem inteligentnih neživih sustava. Inteligentnim sustavom smatramo svaki sustav koji pokazuje prilagodljivo ponašanje, može učiti nove koncepte, barata velikim podacima i sl. U današnje vrijeme UI se razvija rapidno imajući veliki utjecaj na svakodnevni život. Jedan od primjera koji većina koristi na dnevnoj bazi su glasovne naredbe na našim mobilnim uređajima. Također vrlo korisni primjer i za kojim je potražnja sve veća je prepoznavanje pokreta preko kamera na našim računalima, laptopima i mobilnim uređajima. Upravo o toj temi je riječ u ovom seminaru: što je prepoznavanje pokreta, kako kamera prepoznaje naše pokrete i položaje, kako možemo implementirati takav sustav, koje već uvježbane modele koristimo itd. Jedan od glavnih primjera gdje se takav sustav koristi su danas računalne i video igrice. Npr. prije su igrače konzole, kao što je *Nintendo Wii* sa svojim *Wii Remote*, za prepoznavanje pokreta morale koristiti dodatne uređaje sa markerima i različitim senzorima, tada bi se prikupljeni podaci obrađivali i kao rezultat vraćali lokacije dijelova tijela ili brzinu pokreta. Danas sve što je potrebno je web kamera čiji prikupljeni podaci se obrađuju u stvarnom vremenu. Neovisno o kojem tipu podatka je riječ, bili oni slike, videa, tekst i sl., neuronske mreže su se pokazale kao najbolji algoritam za učenje. Za obradu slika kao najpouzdaniji kandidat su se pokazale konvolucijske neuronske mreže o kojima će također biti riječ u ovom seminaru. Već postoje brojni uvježbani modeli neuronskih mreža koji se mogu jednostavno implementirati u aplikacije. S obzirom na jednostavnost implementacije gotovih modela i potražnjom za takvim sustavima gdje sučelje reagira kako se mi krećemo, odlučila sam napraviti jednostavnu web aplikaciju koja može biti primjer za interaktivno učenje. Radi se o primjeru jednostavnog kviza sa općim pitanjima na koja odgovaramo pokretima ruke.

1. PREPOZNAVANJE POLOŽAJA TIJELA

Procjena položaja tijela je složeni problem koji uključuje različite pod-probleme od scene do prepoznavanja dijelova tijela. Ljudi su različitih oblika i veličina: ima mnogo ključnih točaka (kao što su zglobovi) za praćenje, mogu se nalaziti u različitim pozicijama i smjerovima. Također ljudi su često u okruženju drugih ljudi ili predmeta što može dovesti do vizualne okluzije. Neki ljudi koriste različite pomoćne uređaje kao što su invalidska kolica ili štake, koji mogu blokirati dio tijela koji kamera snima. Drugi možda ni nemaju određene udove, ili imati različite proporcije i slično. Sve navedeno su problemi s kojima se susrećemo pri procjeni položaja tijela. Cilj je razviti i kasnije koristiti modele strojnog učenja koji mogu razumjeti i pametno interpretirati podatke o svim tim različitim vrstama tijela.

Kao što je već spomenuto, za prepoznavanje položaja nam treba uvježbani model strojnog učenja. Sigurno mnoge zanima: *Zašto već uvježban? Zašto ga ne uvježbavamo sami?* Prvi razlog je zato što treba značajno vrijeme za uvježbavanje modela, a i jednostavnije, ponekad i pouzdanije, je uzeti već gotov model. Drugi jako bitan razlog je taj što teško možemo nabaviti toliku količinu podataka kako bi naš model bio jako precizan. Potrebno je tisuće, ako ne i stotine tisuća slika, kako bi predstavili sve dijelove tijela u svim mogućim položajima.

2. POGODNE PLATFORME I BIBLIOTEKE

Kako bi započeli s realizacijom konkretnog programa koji će služiti za prepoznavanje položaja tijela, potrebno je razmotriti postojeća rješenja. Danas postoji cijeli niz razvojnih okruženja, programskih jezika i biblioteka koje stoje na raspolaganju programeru početniku. Ideja korištenja biblioteka je ubrzati izradu prototipa uz smanjivanje generiranja pogrešaka koje su neminovne pri izradi programske podrške. Strojno učenje, koje je potrebno za realizaciju programa u sebi uključuje i znanje iz područja statistike, ali i znanje o neuronskim mrežama. Iz navedenog razloga odlučila sam se odabrati biblioteku koja mi može poslužiti za brzu izradu prototipa, ali i koja će omogućiti kasniju industrijsku primjenu izrađenog rješenja. Kao odabir nametnula se biblioteka *Tensorflow*.

2.1. TENSORFLOW

Danas najjednostavnija i najpraktičnija *open source* platforma koja se koristi za strojno učenje je *Tensorflow*, osnovana od strane *Google Brain Team-a*. Ima sveobuhvatan, fleksibilan ekosustav alata, biblioteka i resursa zajednice koji istraživačima omogućuju korištenje najnovije tehnologije strojnog učenja, te da razvojni inženjeri lako izrade i implementiraju aplikacije koje koriste strojno učenje. *Tensorflow* je poslužio za razvoj cijelog niza biblioteka koje služe za rješavanje konkretnih problema temeljenih na neuronskim mrežama. Jedna od njih je i biblioteka *PoseNet*.

2.1.1. PoseNet

PoseNet je biblioteka temeljena na *Tensorflow-u* koja može prepoznati ili jednu pozu ili više poza. To znači da postoji verzija algoritma koja može otkriti pozu samo jedne osobe na slici i koja može otkriti više osoba na slici te također poze i pokrete svih njih. Autori su se odlučili za dvije verzije, prva je puno brža i jednostavnija, dok druga je sporija ali obuhvaća veći broj ljudi. Također je bitno da algoritam ne prepoznaje tko je na slici. Prednost ove biblioteke je što se pokreće na *TensorFlow.js-u* i bilo tko sa pristojnom web-kamerom na računalu ili mobilnom uređaju može koristiti ovu vrstu tehnologije putem browser-a. *PoseNet* je zapravo model strojnog učenja temeljen na neuronskim mrežama, točnije dubokim konvolucijskim neuronskim mrežama.

2.2. OpenCV

OpenCV je *open source* za računalni vid i biblioteka za softvere koji koriste strojno učenje. *OpenCV* je osmišljen kako bi pružio zajednički infrastrukturu za aplikacije računalnog vida i ubrzao korištenje strojne percepcije u komercijalnim proizvodima, te je također besplatan za komercijalne potrebe. Proizvod je licenciran BSD-om tako ta olakšava tvrtkama korištenje i modificiranje koda. Biblioteka ima više od 2500 optimiziranih algoritama, te je glavni fokus na aplikacijama u stvarnom vremenu. Ovi algoritmi mogu se koristiti za prepoznavanje lica, identifikaciju objekata, praćenje pokreta preko kamere, izdvajanje 3D modela itd. Trenutno broji oko 47 000 korisnika i broj preuzimanja je veći od 18 milijuna.

2.2.1. OpenPose

OpenPose je *open source* biblioteka napisana u C++ koja koristi *OpenCV* i *Caffe* za procjenu položaja jedne ili više osoba. *OpenPose* je prva biblioteka koja u stvarnom vremenu prepoznaje ljudsko tijelo, ruku i ključne točke na licu, koristeći jednu sliku. Ukupno prepoznaje do 130 ključnih točaka na tijelu. Neke od funkcija

- prepoznavanje 15-18 ključnih točaka kod prepoznavanja položaja s više osoba; vrijeme izvršavanja varira o broju osoba na slici
- prepoznavanje 2x21 ključnih točaka na šakama
- prepoznavanje 70 ključnih točaka na licu kod prepoznavanja položaja s više osoba, itd.

Temeljena je na neuronskim mrežama. Problem kod ove biblioteke je što zahtijeva da imamo instalirano *OpenCV, Caffe, CUDA* i *cuDNN*, te zato nije baš praktična.

3. STROJNO UČENJE I NEURONSKE MREŽE

Strojno učenje grana je umjetne inteligencije koja se bavi oblikovanjem algoritama koji svoju učinkovitost poboljšavaju na temelju empirijskih podataka. Računala su u prošlosti mogla raditi samo ono za što su bila programirana. Strojno učenje omogućava im da uče na sličan način kako to rade ljudi: stroj prikuplja znanje bazirano na prošlom iskustvu. Umjesto da mu se stalno mora ažurirati softverski kod, on je, kako vrijeme prolazi, samostalno sposoban poboljšavati svoj rad. Metoda strojnog učenja koja se pokazala najboljim kandidatom za učenje su umjetne neuronske mreže.

3.1. UMJETNE NEURONSKE MREŽE

Umjetne neuronske mreže (NM) su metoda strojnog učenja inspirirana načinom učenja u biološkom mozgu. One iterativnim postupkom iz prošlih podataka nastoje pronaći vezu između ulaznih i izlaznih varijabli modela, kako bi se za nove ulazne varijable dobila vrijednost izlaza, ili drugim riječima, uče na primjerima. NM su jedna od metoda strojnog učenja koja postaje sve popularnija u prediktivnoj analitici i u *BigData* platformi.

3.1.1. MOTIVACIJA

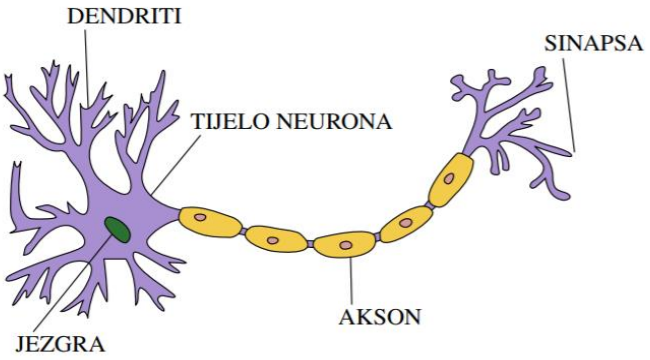
U svijetu u kojem živimo, susrećemo se na dnevnoj bazi s inteligentnim ponašanjem. Počevši od inteligentnog ponašanja kao rezultat suradnje mnoštva bioloških jedinki (mravi, pčele) do inteligencije u pojedinoj jedinki (čovjek). Ljude kao vrstu je oduvijek zanimao pojam inteligencije, možemo li ju razumjeti te možemo li izgraditi inteligentan umjetni sustav. Kroz brojna istraživanja spoznali smo da se mozak sastoji od izuzetno velikog broja neurona (živčanih stanica) koji obradu podataka rade paralelno. Ova spoznaja poslužila je kao motivacija za razvoj potpuno drugačijeg pristupa izgradnji tehničkih sustava i modeliranju znanja. Taj novi pristup gradi tehnički sustav oponašajući građu bioloških sustava. Definiraju se i grade jednostavne procesne jedinice (*umjetni neuroni*) koji se potom povezuju u paralelne strukture različitih arhitektura poznate kao *umjetne neuronske mreže* (engl. *Artificial Neural Networks*).

3.1.2. VRSTE NEURONSKIH MREŽA

Ono što pravi razliku između neuronskih mreža je arhitektura, način učenja i signali. Arhitektura se dijeli na: mreže u kojima veze idu isključivo prema sljedećem sloju, ali nisu zatvorene, pa čak nemaju i jednostruke veze (engl. *Feedforward ANN*), odnosno neuron može biti povezan s više neurona u sljedećem sloju, i na povratne mreže (engl. *Recurrent NN*), odnosno mreže gdje su veze zatvorene na način da se rezultat analize vraća na početak i uvježbavanjem ponavlja s ciljem minimiziranja pogreške.

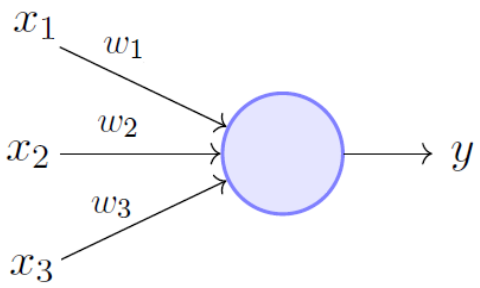
Po načinu učenja dijelimo mreže na: učenje s učiteljem (engl. *supervised*), odnosno učenje koje se provodi s primjerima u obliku para (ulaz, izlaz), te učenje bez učitelja (engl. *unsupervised*) u kojem mreža uči bez poznavanja izlaza.

3.1.3. NEURON



Slika 1. Izgled biološkog neurona

Da bi smo u potpunosti mogli razumjeti umjetne neurone, potrebno je znati osnove biološkog neurona. Biološki neuron (Slika 1.) je stanica koja prima informacije od drugih neurona putem dendrita u obliku električnog impulsa, zatim ih tijelo stanice obrađuje i donosi odluku, a zatim šalje impuls putem aksona i sinapsi drugim neuronima u mreži. Milijuni neurona u mreži mogu paralelno obrađivati informacije.



Slika 2. Izgled umjetnog neurona

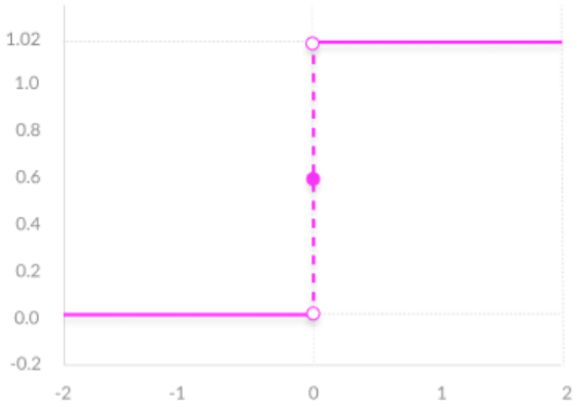
U prikazanom primjeru (Slika 2) možemo vidjeti primjer umjetnog neurona. Umjetni neuron na slici ima tri ulaza: x_1,x_2,x_3 (predstavljaju sinapse), u stvarnosti ih može imati x_n. Uvedene su težine w_1,w_2,w_3 – stvarni brojevi koji izražavaju snagu signala, odnosno važnost dotične ulazne vrijednosti. Umnožak težina i ulaza daje izlaznu vrijednost neurona y, koja se šalje dalje sljedećem neuronu ako je neuron aktiviran. Za aktivaciju neurona se koriste različite matematičke funkcije koje predstavljaju takozvanu prijenosnu funkciju, te nekad izlaz ovisi i o određenom pragu (engl. *threshold value*).

3.1.4. VRSTE UMJETNIH NEURONA

Općeniti model umjetnog neurona kakav je prikazan na slici (Slika 2) možemo dalje razmatrati prema ugrađenoj prijenosnoj funkciji. Prijenosna ili aktivacijska funkcija je matematička funkcija koja odlučuje hoće li se neuron aktivirati ili ne. Njihova uloga je također normalizirati izlaznu vrijednost u intervalima [0,1] ili [-1,1]. Osnovna podjela aktivacijskih funkcija:

1. Binarne „*step*“ funkcije
2. Linearne funkcije
3. Nelinearne funkcije

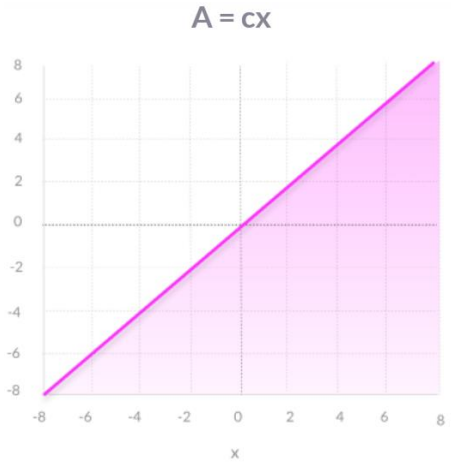
Binarna funkcija je zapravo jako jednostavna i ovisi o pragu koji je unaprijed određen. Ako je težinska suma ulaznih vrijednosti iznad određenog praga, neuron se aktivira i isti signal se šalje dalje idućem neuron, a ako nije, neuron se ne aktivira. (Slika 3.)



Slika 3. Binarna funkcija

Problem s binarnom funkcijom je što ne dopušta izlaze s višestrukim vrijednostima, nego samo 0 ili 1.

Sljedeće su nam linearne funkcije. Linearna uzima ulaze, pomnožene s težinama za svaki neuron i stvara izlazni signal proporcionalan ulazu. Linearna je funkcija bolja od binarne funkcije jer omogućuje višestruke izlaze, a ne samo „da“ i „ne“.

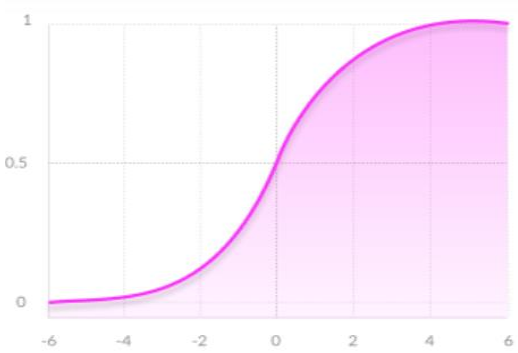


Slika 4. Linearna funkcija

Međutim, linearna također ima neke nedostatke. Prvi nedostatak je taj što nije moguće koristiti algoritam unazadne propagacije jer derivacija funkcije je konstanta te nema veze s našim pravim ulazom, tako da nije moguće vratiti se unatrag i vidjeti koje težine u ulaznim neuronima daju bolje predviđanje. Drugi nedostatak kod neuronskih mreža koji koriste ovu funkciju je taj što će posljednji sloj mreže biti linearna funkcija provg sloja zato što linerana kombinacija linearnih funkcija daje opet linearnu funkciju.

Sljedeće na redu su nelinerane funkcije koje se trenutno sve više upotrebljavaju u modernim neuronskim mrežama. One omogućuju modelu kompleksno mapiranje između ulaza i izlaza mreže, koje je neophodno za učenje i modeliranje složenih podataka, kao što su slike, video, audio i skupovi podataka koji su nelinearni ili imaju visoku dimenzionalnost. Nelinearne funkcije rješavaju probleme linearne funkcije: dopuštaju unazadnu propagaciju jer imaju izvedenu funkciju koja je povezana s ulazima. Također, omogućuju "slaganje" više slojeva neurona kako bi se stvorila duboka neuronska mreža. Nelinearne funkcije su: sigmoidalna, hiberbolički tangens, ReLU, Propustljiv ReLU, Parametarski Relu, Softmax i Swish. Objasniti ću ih u kratkim crtama.

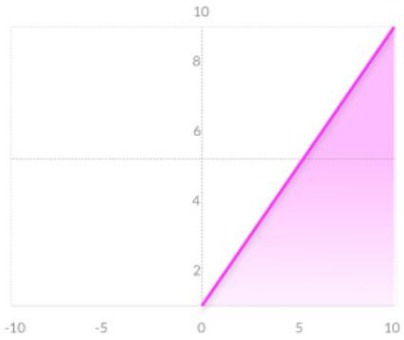
Prva koju ću spomenuti je sigmoidalna funkcija (Slika 5.). Kod ovakve funkcije je prednosti što je derivacija glatka te nema prevelikih skokova između izlaznih vrijednosti. Također izlazne vrijednosti su normalizirane između 0 i 1. Sljedeća prednost je ta što su predviđanja jasna, npr. na Slici 5. vidimo da za x vrijednosti manje od -2 izlaz će biti 0, a za vrijednosti veće od 2, izlaz je 1. Nedostatak je što za prevelike ili premale vrijednosti x-a nema razlike u izlazu te je mogućnost predviđanja slaba, te također nedostatak je što izlazi nisu centrirani na 0.



Slika 5. Sigmoidalna funkcija

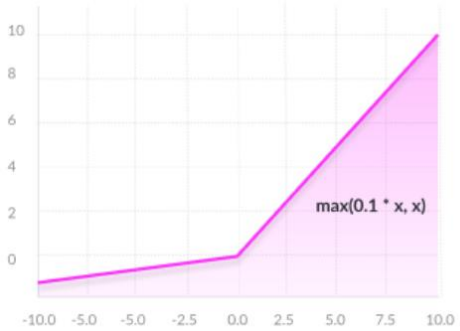
Slična sigmoidalnoj funkciji je tangens funkcija, jedina razlika je što je ona centrirana oko nule i daje izlaze u intervalu od -1 do 1.

Sljedeća funkcija je *ReLU*. Iako izgledom podsjeća na lineranu funkciju, zapravo je nelinerana i omogućuje unazadnu propagaciju. Prednost ove mreže što nije računalno zahtijevna. Problem nastaje kada se ulazi približe nuli ili su negativni, gradijent funkcije postaje nula pa mreža ne može izvoditi propagaciju unatrag i ne može učiti. Ovakav problem se naziva *The Dying ReLU*.



Slika 6. ReLU funkcija

The Dying ReLU problem se riješio koristeći Propustljiv ReLU. Ima iste karakteristike kao obični ReLU samo ova funkcija ima mali pozitivan nagib i za negativne vrijednosti ulaza, pa omogućuje propagaciju unatrag. Problem je što rezultati nisu konzistentni za negativne ulazne vrijednosti.



Slika 7. Propustljivi ReLU

Kao rješenje problema s negativnim dijelom predstavlja se funkcija ReLU sa parametrima, koja kao parametar funkcije uzima nagib negativnog dijela. Stoga je moguće izvesti unazadnu propagaciju i naučiti najprikladniju vrijednost alfa.

$$f(x) = \max(a\delta, \delta)$$

Zatim imamo Softmax funkciju. Softmax se često koristi kao izlazni neuron za klasifikatore, jer daje distribuciju za n različitih klasa. Ova funkcija normalizira izlazne vrijednosti y_i između 0 i 1, bitno je da suma svih vrijednosti y_i je jednaka 1.

$$softmax(z)_i = \frac{e^{z_i}}{\sum_j e^{z_j}}$$

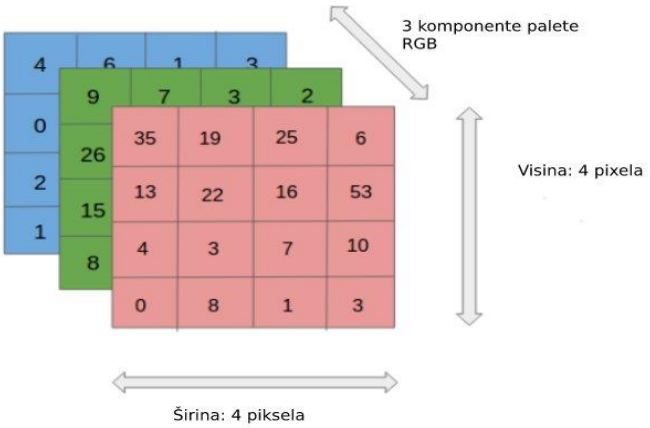
Zadnja funkcija je Swish funkcija. Swish je nova, samostalna aktivacijska funkcija koju su otkrili istraživači na Googleu. Prema njihovom radu, ona je bolja od ReLU-a iako imaju sličnu razinu računalne učinkovitosti. U pokusima na ImageNet-u s identičnim modelima koji su vodili ReLU i Swish, nova funkcija postigla je veću točnost za 0,6-0,9%.

3.2. KONVOLUCIJSKE NEURONSKE MREŽE

Konvolucijske neuronske mreže (eng. *Convolutional Neural Network, ConvNet/CNN*) su se pokazale kao najbolje rješenje kod obrade slika i prepoznavanju značajki s njih. Konvolucijska mreža se sastoji od jednog ulaznog, jednog izlaznog i jednog ili više skrivenih slojeva, baš kao i obična neuronska mreža. Kod konvolucijskih mreža specifični su konvolucijski slojevi i slojevi sažimanja. Najčešće započinju sa jednim ili više konvolucijskih slojeva, zatim slijedi sloj sažimanja, pa ponovo konvolucijskih sloj i tako par puta.

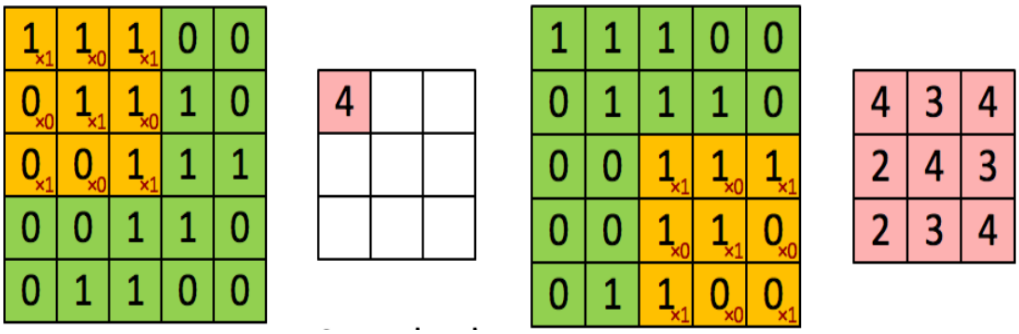
3.2.1. KONVOLUCIJSKI SLOJ

Računalo sliku vidi kao niz pixela ovisno o rezoluciji slike. Računalo sliku vidi kao visina x širina x dubina. Npr. ako imamo sliku u boji JPG formata 4x4, računalo to vidi kao niz pixela 4x4x3 (3 vrijedi jer je RGB paleta). Svaki od tih brojeva ima vrijednost od 0-255 ovisno o jačini piksela na tom području. Upravo to predstavlja ulaz u konvolucijski sloj.



Slika 8. Ulazna slika s 3 komponente

Recimo da nam je ulaz 5x5x1 niz piksela. Najbolji način da se shvati konvolucijski sloj je da zamislimo svjetiljku koja obasjava gornji lijevi dio slike i pretpostavimo da ona obasjava prostor od 3x3, te klizi po svim područjima ulazne slike. Stručno govoreći, ova „svjetiljka“ se naziva **filtrom**, a područje koje obasjava se naziva **receptivno polje**. Filter također prestavlja niz brojeva, najčešće je manjih prostornih dimenzija od ulaza, ali je bitno da je njegova dubina jednaka dubini ulaza. Kako filter „klizi“ po ulaznoj slici, tako se množe vrijednosti u filtru s izvornim vrijednostima slike (odnosno rezultat je skalarni produkt). Filter se pomiče po matrici ovisno o definiranom koraku, ako je korak 1, miče se piksel po piksel, ako je 2, onda svaka dva piksela itd. Ilustracijski prikaz možemo vidjeti na slici dolje (Slika 9).



Slika 9. Prikaz konvolucije 5x5x1 slike sa 3x3x1 filtrom; rezultat je 3x3x1 aktivacijska mapa

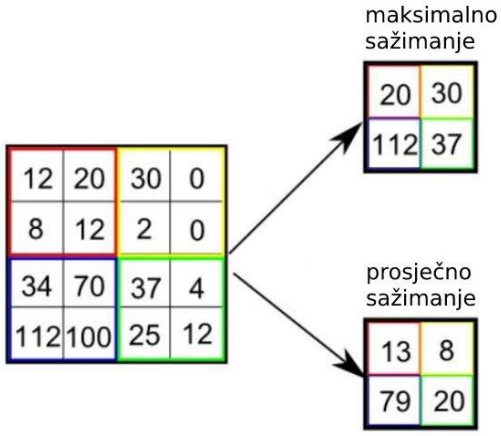
Ako se na primjer na ulazu konvolucijskog sloja nalazi slika s 3 komponente (RGB), filter može ostati isti kao na primjeru iznad, ali je potrebno imati 3 takve matrice, odnosno jednu za svaku komponentu boje. Tada pri izračunu vrijednosti za jedno polje potrebno je sumirati umnoške ulaza i težina svih dubina. Izlaz koji dobijemo se naziva **aktivacijska mapa (mapa značajki).**

Cilj konvolucije je ekstrahirati značajke visoke razine kao što su npr. rubovi. Konvolucijske mreže ne moraju biti ograničene samo na jedan kovolucijski sloj. Prvi sloj uglavnom je za detekciju rubova, boja itd. Uz dodatne slojeve, arhitektura mreže se prilagođava značajkama visoke razine te nam daje mrežu koja ima veliku moć detekcije slika u skupu podataka.

3.2.2. SLOJ SAŽIMANJA

Sloj sažimanja se često koristi s ciljem smanjivanja rezolucije aktivacijske mape, što uzrokuje manju računsku snagu za obradu podataka. Osim smanjivanja rezolucije, ovakvi slojevi povećavaju prostornu invarijantnost. Postoje dvije vrste sažimanja: maksimalno

sažimanje i prosječno sažimanje. Kao što ime samo kaže, maksimalno sažimanje vraća maksimalnu vrijednost iz dijela slike prekriveno filtrom, a prosječno sažimanje vraća prosjek vrijednost. Prednost maksimalnog sažimanja je što suzbija šum (Slika 10).

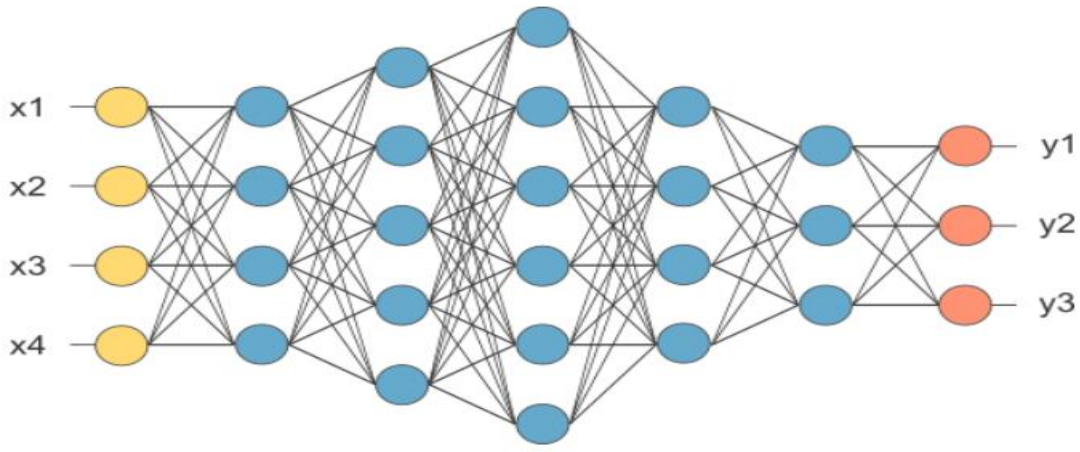


Slika 10.Prikazan način maksimalnog i prosječnog sažimanja

Nakon konvolucijskog sloja i sloja sažimanja, uspješno smo omogučili modelu da razumije značajke. Dalje, „izravnati“ ćemo naš izlaz u vektor i pohraniti ga u neuronsku mrežu radi klasifikacije.

3.2.3. POTPUNO POVEZANI SLOJ

U potpuno povezanom sloju (engl. *Fully Connected Layer, FC*), „izravnali“ smo matricu u vektor i ubacili je u potpuno povezani sloj (Slika 11).

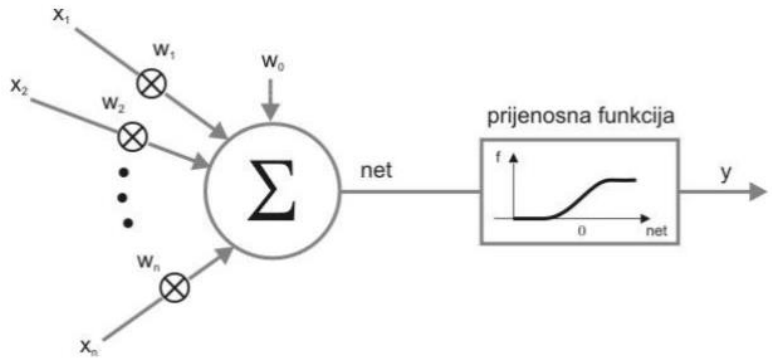


Slika 11. Izgled potpuno povezanog sloja

Nakon toga primjenjuje se algoritam za učenje neuronskih mreža, najčešće je to algoritam propagacije pogreške unatrag (engl. *error backpropagation*).

3.2.4. ALGORITAM PROPAGACIJE POGREŠKE UNATRAG

Kako bi neuronska mreža mogla predstaviti visoko nelinearne funkcije, potrebno je da prijenosna funkcija njezinih procesnih elemenata i sama bude nelinearna funkcija svojih ulaza. Nadalje, radi primjene gradijentne metode pri postupku učenja mreže, potrebno je da prijenosna funkcija bude derivabilna funkcija težinskih faktora. Funkcija koja ispunjava oba navedena uvjeta je ranije spomenuta sigmoidalna funkcija.



Slika 12. Sigmoidalna jedinica

Algoritam učenja se sastoji od dvije faze: unaprijedne faze i unazadne faze. U prvoj fazi algoritma na ulaze neuronske mreže dovode se ulazi za koje je poznato rješenje, računaju se izlazi mreže za dane ulaze i određuje pogreška. U drugoj fazi izračunata greška propagira se unazad i računaju se gradijenti pogreške. Gradijenti se koriste kako bi znali kako treba promijeniti težine pojedinog neurona kako bi se greška smanjila. Pri računanju gradijenata najprije se određuju greške zadnjeg sloja. Zatim se ukupna greška zadnjeg sloja propagira na prezadnji sloj, te se za predzadnji sloj određuje kako je svaki neuron utjecao na grešku. Zatim se određuju komponente gradijenta i ažurira vrijednost težine u ovisnosti o izračunatom gradijentu. Učenje višeslojne mreže pomoću *backpropagation* algoritma svodi se na pretraživanje u n-dimenzionalnom prostoru hipoteza, gdje je n ukupan broj težinskih faktora u mreži. Pogrešku u takvom prostoru možemo vizualizirati kao hiper-površinu koja, za razliku od paraboličke površine jednog procesnog elementa, može sadržavati više lokalnih minimuma. Zbog toga postupak gradijentnog spusta lako može zaglaviti u nekom lokalnom minimumu. U praksi se ipak pokazalo da algoritam unatoč tome daje vrlo dobre rezultate.

Korištena je slijedeća notacija: x_{ij} je ulaz s jedinice *i* u jedinicu *j* (izlaz jedinice *i*), ω_{ij} je odgovarajuća težina, δ_n je pogreška izlaza jedinice *n*. Algoritam kao parametre uzima skup za učenje D, stopu učenja η, broj čvorova ulaznog sloja n_i, broj čvorova izlaznog sloja n_o i broj čvorova skrivenog sloja n_h. Mreži se predočavaju primjeri za učenje u obliku para (x, t) gdje je x vektor ulaznih vrijednosti, a t vektor ciljnih izlaznih vrijednosti.

```
Dok nije ispunjen uvjet zaključivanja čini:
    Za svaki (x, t) iz D:
        //Proslijedi ulazne vrijednosti unaprijed kroz mrežu
        Izračunaj izlaz o_u za svaku jedinicu u za ulaz x
        // Proslijedi grešku unazad kroz mrežu
        Za svaki izlazni čvor k izračunaj pogrešku δ_k
```

$$\delta_k \leftarrow o_k(1 - o_k)(t_k - o_k)$$

```
        Za svaku skrivenu jedinicu h izračunaj pogrešku δ_h
```

$$\delta_h \leftarrow o_h(1 - o_h) \quad \sum_{s \in Downstream(h)} w_{hs}\delta_s$$

```
        Ugodi svaki težinski faktor w_ij
```

$$w_{ij} \leftarrow w_{ij} + \Delta w_{ij}$$

```
    Gdje je
```

$$\Delta w_{ij} = \eta \delta_j x_{ij}$$

4. ODABIR TEHNOLOGIJA

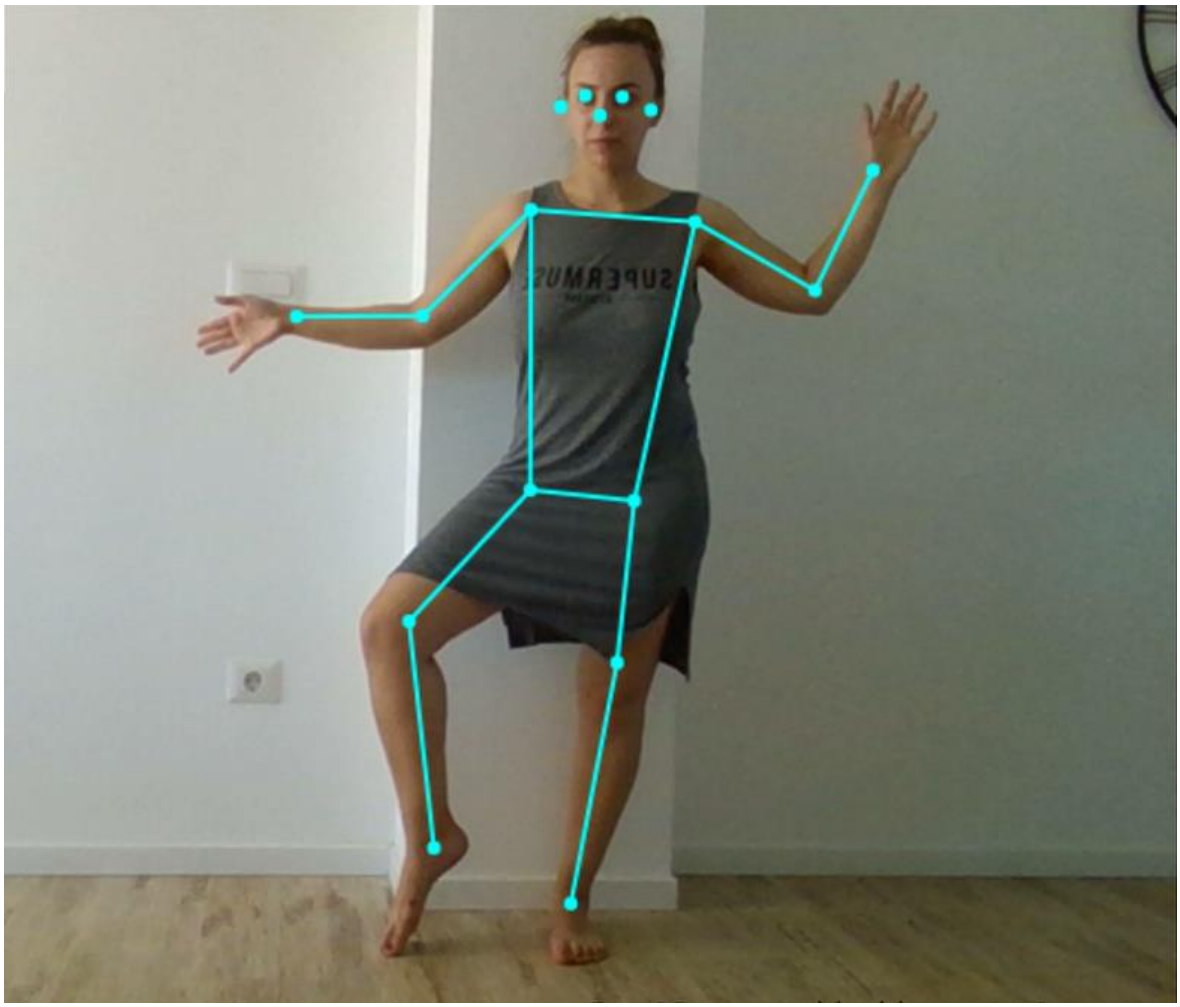
Nakon istraživanja i proučavanja, radi jednostavnijeg korištenja, jednostavnije implementacije te jednostavne dokumentacije, odlučila sam se za *Tensorflow* jer on podržava biblioteku *Tensorflow.js* koja omogućava strojno učenje u browseru, što je zapravo jako praktično i nikakve druge posebne komponente nisu potrebne. Također velika prednost strojnog učenja u browseru je privatnost, pogotovo ako se radi o velikoj količini podataka ili nekim podatcima koji moraju biti tajni. Sve se izvršava lokalno, na našim računalima, nema pohranjivanja na nečije „oblake“ i sl.

4.1. Kako *PoseNet* funkcionira?

PoseNet je algoritam koji prepoznaje samo položaje tijela, odnosno ključne točke na slici, ali ne prepoznaje tko je na slici.

Ako pričamo na visokoj razini, a ne toliko detaljno, možemo reći da se prepoznavanje pokreta pomoću *PoseNet-a* odvija u dvije faze:

1. RGB slika kao ulazni podatak se pohranjuje u konvolucijske neuronske mreže
2. Koristi se jedna od verzija algoritama (jedna-poza/više-poza) za dekodiranje izlaznih podataka modela kao što su poze, pouzdanost poze, pozicije ključnih točaka (koordinate) i pouzdanost ključnih točaka.



Slika 13. 17 ključnih točaka

Algoritam vraća *Promise* koji ima 2 atributa. Prvi atribut je pouzdanost poze, od 0 do 1, odnosno koliko je algoritam siguran da je prepoznao pozu. Drugi atribut je niz od 17 objekata, odnosno 17 ključnih točaka koje možemo vidjeti na slici gore (Slika 13.). Svaki objekt koji predstavlja ključnu točku ima atribute kao što su pouzdanost u vrijednosti od 0 do 1, odnosno da je algoritam prepoznao ključnu točku, te koordinate x i y koji predstavljaju gdje se ključna točka nalazi na slici.

Najlakši način za instalirati PoseNet je direktno putem paketa sa stranice:

```
<html>
  <body>
    <!-- Load TensorFlow.js -->
    <script
src="https://unpkg.com/@tensorflow/tfjs"></script>
    <!-- Load Posenet -->
    <script src="https://unpkg.com/@tensorflow-
models/posenet">
```

```
    </script>
    <script type="text/javascript">
      posenet.load().then(function(net) {
        // posenet model loaded
      });
    </script>
  </body>
</html>
```

Kao što je već navedeno postoje 2 verzije algoritma. Razlika je u brzini izvođenja i nekim ulaznim parametrima.

Bitan detalj za naglasiti je da su istraživači uvježbali *ResNet* i *MobileNet* model *PoseNet-a*. *ResNet* je model koji ima veću mrežu od *MobileNet-a* što rezultira njegovim sporim izvršavanjem. Također druga prednost *MobileNet-a* je ta što je dizajniran i za mobilne uređaje.

4.1.1. ALGORITAM S JEDNOM POZOM

Ulazni parametri za algoritam s jednom pozom:

1. Slika koja se obrađuje
2. Faktor skaliranja – broj između 0.2 i 1 (zadano je 0.5), odnosno za koliko skalirati sliku prije pohranjivanja u neuronsku mrežu; što je broj niži, veća je brzina izvođenja, ali može doći do nekih nepreciznosti
3. *Flip horizontal* – vrijednost koja može biti *true* ili *false* (zadano je f*alse*), odnosno da li da poze budu zrcaljene horizontalno; postavljamo vrijednost na *true* kad se radi o videima jer želimo da poze budu u pravilnoj orijentaciji u odnosu na nas
4. *Output stride* – vrijednost koja mora biti 32,16 ili 8 (zadano je 8). Ovaj parametar utječe na visinu i širinu slojeva u neuronskoj mreži; što je vrijednost veća - vrijeme izvođenja je duže ali preciznost veća, što je vrijednost manja – vrijeme izvođenja je kraće ali i preciznost manja.

4.1.2. ALGORITAM S VIŠE POZA

Ova vrsta algoritma ima iste navedene parametre kao navedeni algoritam iznad , te još par dodatnih kao što su:

1. Maksimalan broj poza za prepoznavanje – zadana vrijednost je 5
2. Prag pouzdanosti – broj u vrijednosti između 0.0 do 1.0 (zadano je 0.5); kontrolira najmanju vrijednost pouzdanosti koju algoritam vraća
3. Radijus bez maksimalnog potiskivanja (NMS) – broj piksela koji kontrolira minimalnu udaljenost između poza koje algoritam vraća

S obzirom na vrijeme i resurse, odlučila sam se za algoritam s jednom pozom iz razloga što je brži te takav sustav je lakše testirati, u protivnom bih trebala imati cijelu grupu ljudi kraj sebe.

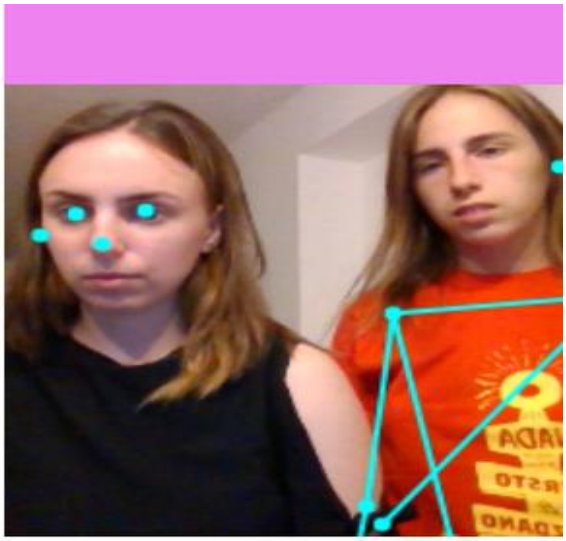
4.1.3. OGRANIČENJA

Problem s učitavanjem direktno paketa sa stranice je taj što paketi se često ažuriraju, izlaze nove verzije i link kojeg koristite učitava zadnju, najnoviju verziju. Učitavanjem nove verzije, aplikacija više ne izvršava ono što želimo jer paket više nema iste funkcionalnosti. Upravo to se dogodilo tijekom implementacije ovog projekta. Korištena verzija *PoseNet-a* je bila 1.0.3., te je stigla nova verzija 2.0.0. U novoj verziji neke funkcije iz prethodne verzije više ne postoje. Primjer: u verziji 1.0.3 funkcija kojom prepoznajemo jednu pozu je bila *EstimateSinglePoses()* te joj se šalju određeni parametri. U verziji 2.0.0 ta funkcija je *EstimatePoses()* koja prima iste parametre kao i metoda prije, ali joj se dodatno kao parametar šalje parametar *decodingMethod: „single-person“*. Kako bi riješili ovaj problem i postigli da aplikacija se izvršava kao prije, u *npm* linku kojim dohvaćamo pakete, trebamo referencirati verziju koju smo koristili. U ovom slučaju:

```
<script src="https://cdn.jsdelivr.net/npm/@tensorflow-models/posenet@1.0.3"></script>
```

Slika 14. Rješenje referenciranja određene verzije paketa

Također još jedno ograničenje je što ako na slici se nalaze 2 osobe, a pozvan je model za prepoznavanje jedne poze, poza će biti krivo procijenjena. Razlog tome je što algoritam će ključne točke obaju osoba gledati kao dio jedne poze (Slika 15).



Slika 15. Primjer krive procjene položaja tijela

4.2. WEB TEHNOLOGIJE

Razlog zbog koje su web tehnologije vrlo praktične i sve popularnije u zadnje vrijeme je taj što su dostupne u bilo koje vrijeme i s bilo kojeg mjesta. Web aplikacijom se smatra bilo koji računalni program koji koristi web preglednik kao klijenta.

Prednosti web aplikacije:

- Lako održavanje (promjene odmah vidljive korisnicima)
- Promjene se mogu raditi “on-line” dok je aplikacija aktivna i korisnici je koriste
- Dostupnost aplikaciji iz bilo kojeg dijela svijeta (potrebna samo internet veza)
- Rade bez obzira na operacijski sustav koji je instaliran na računalu
- Velika brzina i privatnost podataka

Nedostatci s kojima se susreću web aplikacije su nedovoljna usuglašenost web standarda, problemi sa nesigurnošću veze itd.

Dodatni razlog zašto sam se odlučila za web tehnologije je taj što, kao što je već navedeno, *Tensorflow* podržava biblioteku *Tensorflow.js* za pokretanje strojnog učenja u web pregledniku.

4.2.1. HTML

HTML je kratica za HyperText Markup Language, što znači prezentacijski jezik za izradu web stranica. On je najosnovnije gradivno sredstvo na Web-u. Koristi se za kreiranja i vizualno predstavljanje sadržaja na Web stranicama uključujući i osnovni raspored Web stranice. „HyperText“ u imenu se odnosi na link-ove koji povezuju Web stranice međusobno, bilo unutar jedne web-lokacije ili između web-lokacija.

HTML koristi „markup“ za označavanje teksta, slika i drugog sadržaja za prikaz u web pregledniku. HTML označavanje uključuje posebne „elemente“ kao što su <head>, <title>, <body>, <header>, <footer>, <article>, <section>, <p>, , , img>, <strana>, <audio>, <canvas>, <datalist>, <details>, <embed>, <nav>, <izlaz>, <progress>, <video> i mnoge druge.

4.2.2. CSS

CSS (Cascading Style Sheets) je jezik lista oblikovanja koji se koristi kako bi se opisala prezentacija dokumenta napisanog u HTML ili XML jeziku. CSS opisuje kako će se pojedini elementi prikazati na ekranu, papiru, u govoru ili na drugim medijima.

CSS je jedan od osnovnih jezika Web-a te je standardiziran kroz Web preglednike prema W3C specifikaciji. Razvijan je u razinama pa je tako CSS1 danas zastario, CSS 2.1 je preporuka za korištenje, dok CSS, trenutno podijeljen u manje module, napreduje u standardizaciji. Zajedno sa HTML-om, CSS čini statičku prezentaciju podataka klijentu.

4.2.3. JavaScript

JavaScript (JS) je programski jezik koji se uglavnom koristi za dinamičko skriptiranje web-stranica na strani klijenta, ali se često koristi i na strani poslužitelja, koristeći pakete kao što je Node.js. Važno je ne miješati JavaScript sa jezikom Java. „Java“ i „JavaScript“ su zaštitni

znakovi ili registrirani zaštitni znakovi tvrtke Oracle u Americi i drugim zemljama. Međutim, dva programska jezika se značajno razlikuju u svojoj sintaksi, semantici i upotrebi. JavaScript se uglavnom koristi u pregledniku, omogućujući programerima da manipuliranju sadržajem web-stranica putem DOM-a, manipuliraju podacima pomoću AJAX-a i IndexedDB-a, crtaju grafiku na platnu, komuniciraju s uređajem koji pokreće preglednik kroz razne API-je, itd. JavaScript je jedan od najčešće korištenih jezika na svijetu zahvaljujući nedavnom rastu i poboljšanju performansi API-ja dostupnih u preglednicima.

Standard za JavaScript je ECMAScript. Od 2012. godine svi Web preglednici podržavaju ECMAScript 5.1. Stariji Web preglednici podržavaju najmanje ECMAScript 3. 17. lipnja 2015. godine, ECMA International objavljuje šestu major verziju ECMAScript-a, ECMAScript 2015. Od tada, ECMAScript standardi se objavljuju na godišnjoj razini.

Prednosti JavaScripta:

- Manja potreba za komunikaciju sa serverom; ispravnost podataka se može provjeriti prije slanja na server
- Trenutni odziv korisniku; korisnik ne treba čekati da se stranica ponovno učita kako bi vidio je li zaboravio unijeti neki podatak
- Povećana interaktivnost; moguće je kreirati sučelje koje reagira na korisnikove akcije tipkovnice ili miša
- Bogatije sučelje; možete koristiti JavaScript za uhvati i pusti komponente ili animirati bogatije sučelje prema korisniku

Ograničenja JavaScripta:

- JavaScript nema mogućnost višenitnog ili višeprocesorskog izvođenja
- JavaScript ne može se koristiti za mrežne aplikacije jer za nju ne postoji podrška.
- JavaScript je klijentski programski jezik i nije mu dopušteno pisanje ili čitanje datoteka. Ova osobina je ugrađena radi sigurnosnih razloga.

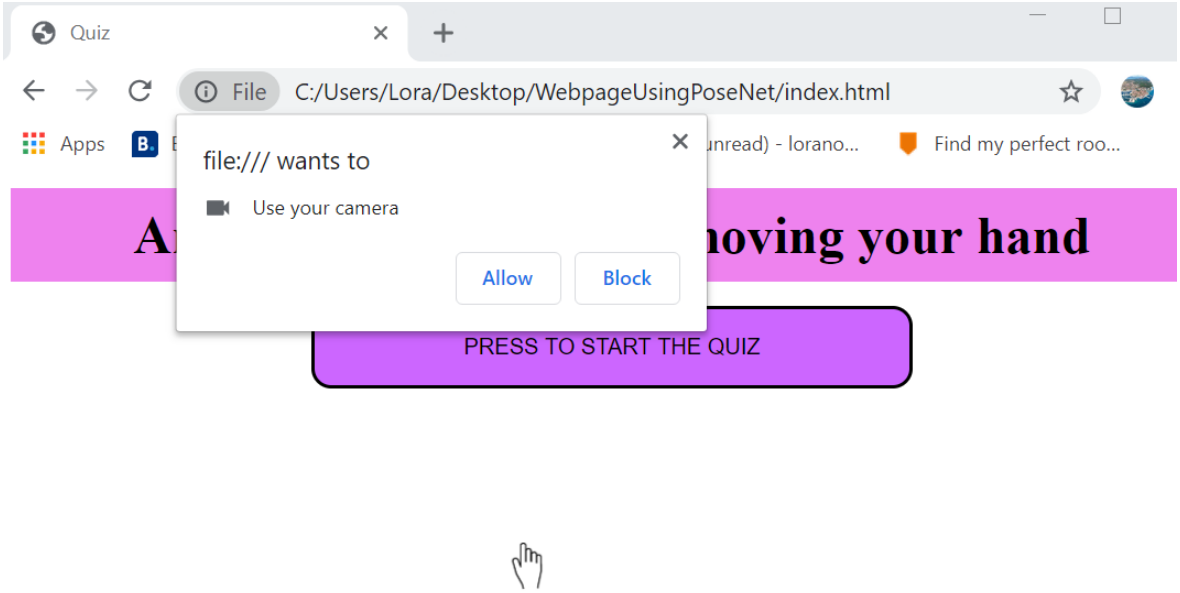
Zahtjevi prema aplikaciji:

1. Omogućiti prepoznavanje položaja ruke koristeći mobitel ili računalo
2. Izraditi prototip korisničkog sučelja osjetljivog na pomak
3. Izraditi okvir za definiranje interaktivnog kviza

5. IMPLEMENTACIJA APLIKACIJE

Početak procesa obrade podataka započinje prikupljanjem podataka. U našem slučaju radi se o slici osobe koja se nalazi ispred web kamere. Kako postoje različiti načini prikupljanja slike u nastavku ću opisati detaljno kako je realizacija napravljena u ovom prototipu.

Korisničko sučelje aplikacije realizirano je HTML-om i CSS-om, dok JavaScript služi za dinamiku stranice. U prvoj fazi aplikacije je potrebno učitati kameru, te dati dozvolu za korištenje kamere u svrhu prikupljanja podataka. (Slika 16).

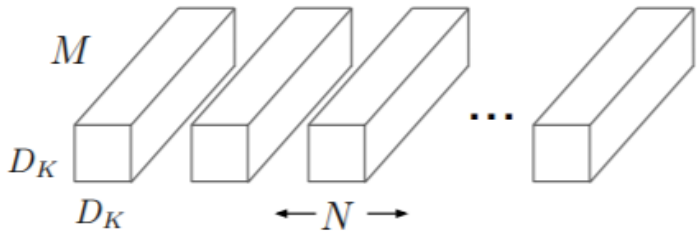


Slika 16. Pitanje korisnika dozvolu za korištenje kamere

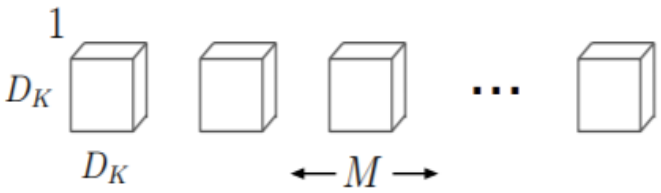
Nakon toga učitavamo *PoseNet* model, unutar asinkrone funkcije, jer se ona neprekidno izvodi u pozadini. **Jako važno** je prije svakog učitavanja modela koristiti *.dispose()* kako bi očistili memoriju grafičke kartice, inače se memorija jako brzo puni te se cijela aplikacija usporava i na kraju prestane funkcionirati jer ne može izdržati toliko opterećenje. U ovoj aplikaciji je korišten tzv. prijenos učenja (engl. *transfer learning*), odnosno koristimo uvježbani model. Testirajući aplikaciju nekoliko puta odlučila sam se za *MobileNet* model jer je dosta brži nego *ResNet*, a brzina je ključan pojam uzimajući u obzir da se radi o prepoznavanju položaja u stvarnom vremenu. *MobileNet* model je uvježban koristeći *ImageNet*, skup podataka koji sadrži preko 14 milijuna slika.

Već je navedeno i objašnjeno kako konvolucijske mreže obrađuju slike, a ono što *MobileNet* čini posebnim i pouzdanim je njegov način rada sa slikama u boji. Razlika u konvoluciji kod *MobileNet-a* je ta što ona primjenjuje filter za svaku komponentu boje, odnosno za sliku u boji - RGB , primjenjuje poseban filter za crvenu (R-*red*), zelenu (G-*green*) i plavu (B–*blue*) komponentu. Nakon toga se primjenjuje konvolucija 1x1 i kombinira izlaze prethodne konvolucije u izlaz. Standardna konvolucija filtrira sve skupa i kombinira ulaze u novi skup izlaza u jednom koraku. Kod *MobileNet-a* imamo dodatni sloj nazvan *depthwise separable convolutional layer*, koji sadrži poseban sloj za filtriranje po bojama i poseban sloj za kombiniranje izlaza. To uvelike smanjuje računanje i veličinu modela. Ukupan broj računanja je 8–9 puta manji od standardnih konvolucija i rezultira mogućnošću rada na mobilnim uređajima. Na slikama dolje vidimo kako izgleda standardna konvolucija (Slika 19.) i kako taj sloj izgleda razbijen u 2 dijela kod *MobileNet* modela (Slika 18. i Slika 19.).

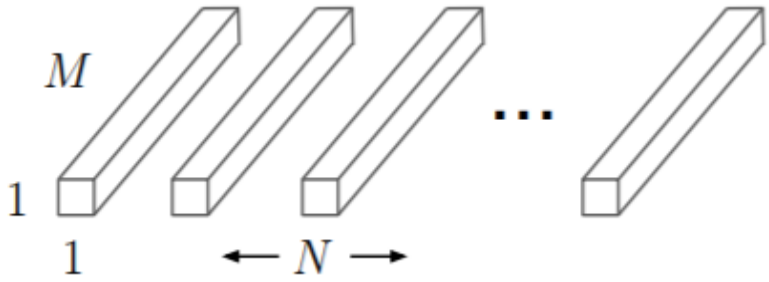
Korištena je notacija: D_K je prostorna dimenzija filtra, M je broj ulaza, a N broj izlaza (Slika 19.), oboje prethodno definirani.



Slika 17. Standardna konvolucija



Slika 18. Dubinska konvolucija



Slika 19. 1x1 konvolucijski filtri

MobileNet koristi *ReLu* nelinearnu funkciju za sve slojeve, osim zadnjeg sloja. Cijela mreža *MobileNet-a* je izgrađena od dubinskih konvolucijskih slojeva, osim prvog sloja koji je isti kao standardni, te zadnjeg, potpuno povezanog sloja. Zadnji, potpuno povezani sloj, koristi softmax funkciju za klasifikaciju slike. Ako brojimo sve ove duboke konvolucijske slojeve kao odvojene slojeve, *MobileNet* je mreža koja sadrži 28 slojeva.

Nakon što je slika prošla kroz *MobileNet* model , kao izlaz dobivamo *heatmap-u* sa *offset* vektorima koji prikazuju područja visoke pouzdanosti naših ključnih točaka. Svaka *heatmapa* je tenzor veličine rezolucija x rezolucija x 17, s obzirom da vraća 17 ključnih točaka. *Heatmapa* je samo aproksimacija naših ključnih točaka koja za svaki dio slike sadrži vjerojatnost da se naša ključna točka nalazi baš na tom području. *Offset* vektori su ti koje je lakše dekodirati i pomoću smjera vektora dolazimo do precizne lokacije naših ključnih točaka. Sad kad smo vidjeli kako model izgleda u teoriji, vrijeme je da pokažemo i vizualno.

Slika 20. Prepoznavanje ključnih točaka

Na slici iznad (Slika 20.) vidimo kako model prepoznaje naše ključne točke i povezuje ih. Rekli smo da *PoseNet* kao izlaz vraća koordinate naših ključnih točaka, na nama samo ostaje da ih vizualno prikažemo. Za povezivanje i crtanje ključnih točaka i skeleta se koriste metode

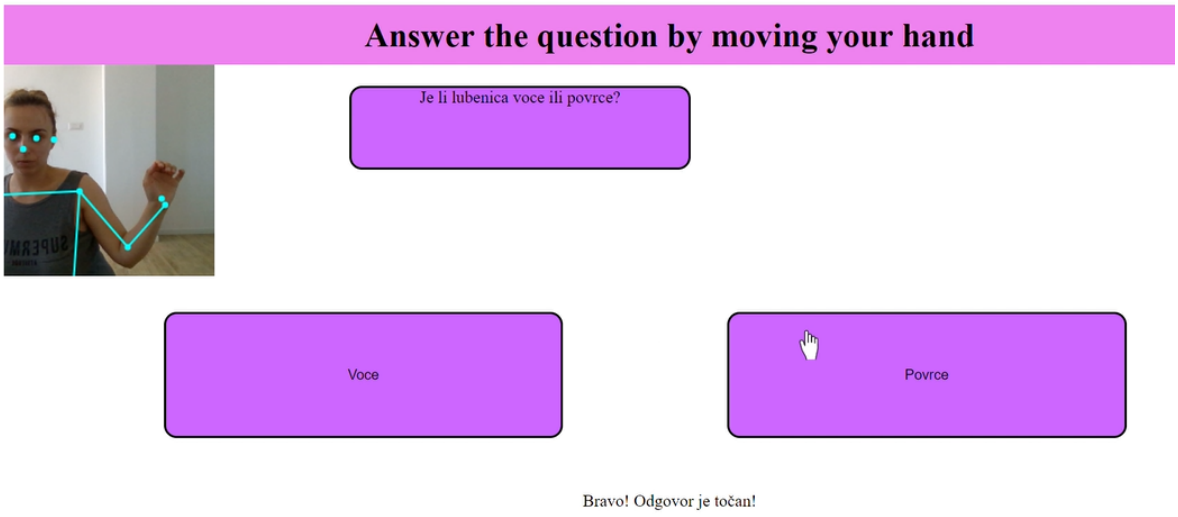
za crtanje po canvasu. Crtanje je jednostavan proces. Koristi se canvas, odnosno „platno“ za crtanje grafike. Na mjestima koordinata crtamo kružnice ispunjene bojom, a za povezivanje ključnih točaka se koriste obične ravne linije. PoseNet sam ima ugrađenu funkciju koja prepoznaje koje ključne točke treba povezati, a koje ne, kao što su npr. (oči, nos, itd.). Za potrebe naše aplikacije je dovoljan samo gornji dio tijela i desna ruka, te zato nije prikazano prepoznavanje ključnih točaka cijelog tijela.

Na slici dolje (Slika 21) je prikazan početak interaktivnog kviza.

Slika 21. Početak kviza

Potrebno je ruku dovući na botun za početak kviza. Za upravljanje aplikacijom i odgovaranje na pitanja ne pomičemo pravi miš, već sliku pokazivača. Razlog je taj što se koordinate pravog miša ne mogu dinamički mijenjati iz sigurnosnih razloga. Kako bi uskladili koordinate naše ruke i koordinate našeg improviziranog miša, potrebno je skalirati koordinate jer algoritam vraća na kojim koordinatama se nalazi naša ruka (odnosno zglob) unutar našeg videa na ekranu. Aplikacija prati ključnu točku „*rightWrist*“ i provjerava stalno njene koordinate. Ako se slučajno koordinate poklapaju sa koordinatama botuna, aplikacija ponovo provjera koordinate nakon 1000 ms, te ako je ruka još unutar zadanih koordinata, pokreće se aplikacija. Tako se koordinate provjeravaju i za ostale odgovore. Koristila sam takav princip rada zato što ne postoji nešto ugrađeno kako bi mogli detektirati „klik“ rukom, već samo možemo imitirati naš pravi miš i njegovu metodu „*hover*“.

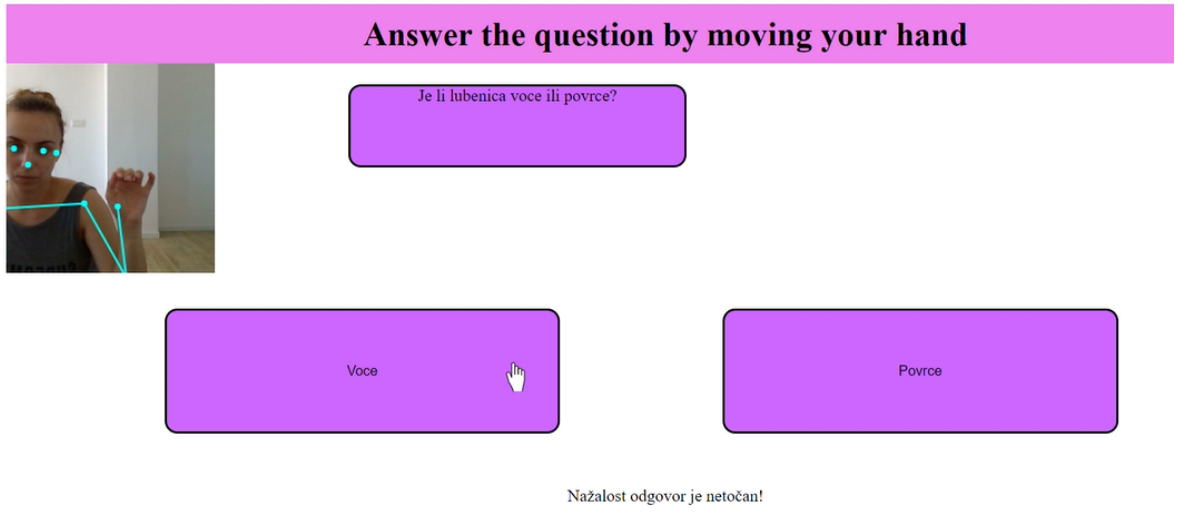
Sljedeće fotografije pokazuju reakcije aplikacije na pomak ruke.



Slika 22. Primjer pitanja i reakcija na točan odgovor

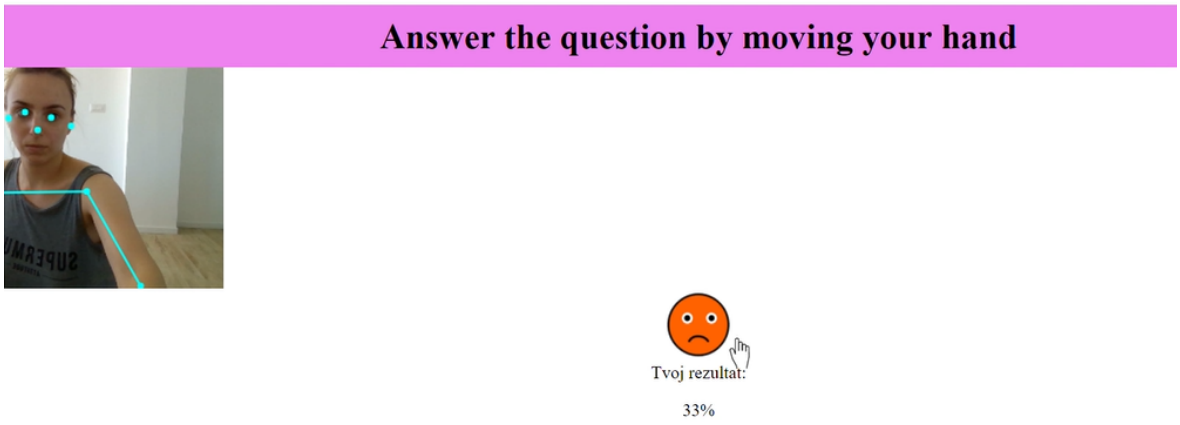
Na slici iznad (Slika 22) vidimo kako izgleda primjer pitanja kviza. Ruku pomičemo u smjeru točnog odgovora kako bi odgovorili na zadano pitanje. Pomicanjem naše ruke pomiče se i naš pokazivač. Na slici je također i prikazana reakcija na točan odgovor. U pozadini se računaju koordinate pozicije na kojoj se ruka nalazi, odnosno prepoznata ključna točka, te se računaju početne i krajnje točke naših odgovora. Nakon toga se provjerava da li se koordinate ruke nalaze u točnom intervalu koordinata, te se ispisuje prigodna poruka ovisno o našem odgovoru.

Na sljedećoj slici (Slika 23) se odvija isti proces, te se ispisuje odgovarajuća poruka za netočan odgovor.



Slika 23. Primjer pitanja i reakcija na netočan odgovor

Na kraju odigranog kviza , ispisuje se konačan rezultat kviza u postotcima (Slika 24). S obzirom da se kviz sastoji od samo 3 pitanja , mogući rezultati su 33%, 67% i 100%. Također je prikazan i odgovarajući smiješak uz ostvareni rezultat.



Slika 24. Krajnji rezultat

6. ZAKLJUČAK

Kroz ovaj seminar je obrađena tema prepoznavanja položaja tijela u stvarnom vremenu koristeći web kameru. Opisan je detaljno način djelovanja, učenja i uporabe neuronskih mreža, s naglaskom na konvolucijske neuronske mreže. Seminar je pisan s namjerom da se što jednostavnije pojasni način obrade slika putem konvolucijskih neuronskih mreža koje su se pokazale kao najbolji kandidat za obradu.

Predstavljene su odgovarajuće biblioteke za korištenje strojnog učenja koje pojednostavljuju i ubrzavaju proces implementacije željenih aplikacija. Implementirana je jednostavna web aplikacija koja može poslužiti kao primjer interaktivnog učenja za djecu.

Aplikacija koristi biblioteku *PoseNet*. Nakon testiranja aplikacije odlučila sam se za uvježbani *MobileNet* model jer je dosta brži nego *ResNet*, a brzina je ključna za ovakav tip aplikacija. Prikazano je kako je jako jednostavno implementirati aplikaciju koja koristi neuronske mreže, a može biti jako korisna i efektivna.

Zanimljivo je kako umjetne neuronske mreže mogu djelomično kopirati biološke neuronske mreže. Umjetne neuronske mreže imaju prednost što se tiče brzine obrade podataka, no ljudski mozak je ipak mnogo kompleksniji i njegov način zaključivanja nikad neće moći zamijeniti umjetna neuronska mreža.

Literatura

[1] Michael A. Nielsen, „*Neural Networks and Deep Learning“*, Determination Press, 2015

[2] Prof.dr.sc. Bojana Dalbelo Bašić, mr.sc. Marko Čupić, mr.sc. Jan Šnajder, *„Umjetne neuronske mreže“,* Zagreb, svibanj 2008.

[3] Carlos Gershenson, članak „*Artificial Neural Networks for Beginners*“

(https://www.researchgate.net/publication/1956697_Artificial_Neural_Networks_for_Beginners)

[4] https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148

[5]https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[6] https://developer.mozilla.org/en-US/docs/Web/HTML

[7] https://developer.mozilla.org/en-US/docs/Web/CSS

[9] https://developer.mozilla.org/hr/docs/Web/JavaScript

[10]https://medium.com/tensorflow/real-time-human-pose-estimation-in-the-browser-with-tensorflow-js-7dd0bc881cd5

[11]https://adeshpande3.github.io/adeshpande3.github.io/A-Beginner%27s-Guide-To-Understanding-Convolutional-Neural-Networks/

[12]https://missinglink.ai/guides/neural-network-concepts/7-types-neural-network-activation-functions-right/

[13] *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,* Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, Hartwig Adam

Sažetak

Umjetna inteligencija je danas sve popularnija i potreba za njenim mogućnostima je sve veća. Jedna od tih potreba je opisana u ovom radu, a to je prepoznavanje pokreta putem web-kamere. Kako bi računalo moglo samo prepoznati naše dijelove tijela potrebno ga je i naučiti kako to napraviti. Za to se koristi disciplina strojno učenje, kao što samo ime kaže „stroj sam uči“. Algoritam koji se pokazao najbolji za učenje su neuronske mreže, čije su osnove pojašnjenje u ovom radu. S obzirom da je zadatak prepoznati pokrete tijela, ulazni podatak nam je slika (obična slika, video), a najboljim i najpouzdanijim kandidatom za obradu slike su se pokazale konvolucijske neuronske mreže. Njihov princip i rad je također prikazan u ovom radu. S obzirom da naš zadatak nije bio istrenirati mrežu, već smo uzeli gotov istreniran model, u seminaru se ne objašnjava detaljno kako istrenirati konvolucijsku mrežu, već samo kako ona funkcionira. Također u radu su spomenute neke gotove biblioteke s istreniranim modelima pogodne za prepoznavanje položaja tijela koje ubrzavaju proces izgradnje prototipa.

Ključne riječi: prepoznavanje pokreta, strojno učenje, neuronske mreže, konvolucijske neuronske mreže, *PoseNet , MobileNet*

SKRAĆENICE

HTML - Hypertext Markup Language

CSS – Cascading Sheet Style

JS – JavaScript

BSD - Berkeley Source Distribution

RGB – Red, Green, Blue

NM – Neuronske mreže

FC – Fully Connected Layer

JPG – Joint Photographic Expert Group

CNN – Convolutional Neural Network

TLU – Threshold Logic Unit

Itd. – i tako dalje

sl. – slično

engl. - engleski

POPIS SLIKA