Fakultet elektrotehnike, računarstva i informacijskih tehnologija, Osijek

Prepoznavanje maske

Seminar za kolegij Primjenjeno strojno učenje

Mentor: Ratko Grbić

Benjamin Montibeler, Lora Duvnjak, Heda Nikolić

Stručni studij računarstva

Osijek 20.06.2022.

Sadržaj

[**1.** **UVOD** 3](#_Toc106831011)

[**2.** **PODACI I OBRADA PODATAKA** 4](#_Toc106831012)

[**3.** **TRENIRANJE MODELA** 7](#_Toc106831013)

[**3.1.** **KONVOLUCIJSKA MREŽA** 7](#_Toc106831014)

[**3.2.** **TRANSFER LEARNING MREŽA** 9](#_Toc106831015)

[**4.** **TESTIRANJE RJEŠENJA NA VLASTITOM SKUPU PODATAKA** 13](#_Toc106831016)

[**5.** **KORIŠTENJE MODELA NA SNIMCI S KAMERE** 14](#_Toc106831017)

[**6.** **TESTIRANJE POMOĆU KAMERE** 18](#_Toc106831018)

[**7.** **PRIMJENA** 19](#_Toc106831019)

[**8.** **LITERATURA** 20](#_Toc106831020)

# **UVOD**

Strojno učenje je proučavanje računalnih algoritama koji se mogu automatski poboljšati korištenjem iskustva i podataka. Smatra se dijelom umjetne inteligencije. Algoritmi strojnog učenja grade modele na temelju podataka uzorka, koji se još nazivaju podaci za obuku (“training data”), kako bi mogli predvidjeti ili donijeti odluke bez eksplicitnog programiranja. Algoritmi strojnog učenja koriste se u nekoliko područja, kao što su medicina, filtriranje e-pošte i prepoznavanje govora, gdje je teško ili nepraktično razviti tradicionalne algoritme za obavljanje potrebnih zadataka.

Za temu našeg projektnog zadatka izabrali smo “Mask recognition” tj. Prepoznavanje maske. Pomoću raznih softvera za strojno učenje, poput Tenserflow, Sklearn i Numpy, razvili smo program koji u vremenu prepoznaje lice te nalazi li se na njemu medicinska maska ili ne.

# **PODACI I OBRADA PODATAKA**

Prije početka same izrade programa te pisanja koda, nužno je pronaći odgovarajuću bazu podataka. Baza koja je korištena pri izradi ovog projekta nalazi se na web stranici Kaggle (https://www.kaggle.com/datasets/vijaykumar1799/face-mask-detection?resource=download). Podaci unutar baze podataka raspoređeni su u dva direktorija:

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

U prvom direktoriju “with\_mask” nalaze se slike ljudi s medicinskim maskama na licu dok se u drugom direktoriju “without\_mask” nalaze ljudi bez maski.

U sljedećim linijama koda učitali smo bazu podataka u program:

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Zatim smo kreirali dvije prazne liste te u prvu pohranili slike iz dataset-a, a u drugu string koji nam govori nalazi li se na slici maska ili ne. To smo napravili pomoću for petlje koja prolazi kroz već kreiranu listu ‘CATEGORIES’. Deklarirali smo novu varijablu path te smo spojili listu ‘DIRECTORY’ i trenutnu kategoriju iz liste ‘CATEGORIES’ kao putanju koju smo pohranili u varijablu path. Pomoću druge for petlje prošli smo po svim slikama u pojedinom direktoriju. U varijablu ‘img\_path’ spremili smo putanju svake slike, zatim smo prilagodili dimenziju svake slike te pretvorili slike u array. Nakon svakog izvršavanja petlje, sliku spremamo u ‘data’ listu, a kategoriju u ‘labels’.

Slika na kojoj se prikazuje tekst, zaslon

Opis je automatski generiran

Iako su podaci unutar data liste numerički, unutar ‘labels’ nisu, stoga ih smo ih morali konvertirati.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Nakon konvertiranja ‘labels’, prebacili smo obje liste u numpy array tj. polje.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Koristeći se funkcijom ‘train\_test\_split’ odvojili smo 25% slika za testiranje, a preostalih 75% koristili smo kao validacijske podatke.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Zatim smo normalizirali i preoblikovali podatke.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Augmentacija podataka korisna je za poboljšanje performansi i ishoda modela strojnog učenja formiranjem novih i različitih primjera za obuku skupova podataka. Kako bi razvili bolje modele implementirali smo različite vrste augmentacije.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

# **TRENIRANJE MODELA**

Pojam ‘treniranje’ u strojnom učenju znači dati algoritmu dovoljno podataka iz kojih se on može obučiti tj. napraviti model pomoću kojeg će donositi predikcije. Za *face mask recognition* program izradili smo dva modela; konvolucijski model te model koristeći se transfer learning metodom.

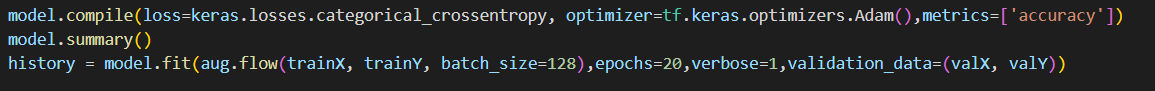
## **KONVOLUCIJSKA MREŽA**

U dubokom učenju, konvolucijska neuronska mreža (CNN ili ConvNet) je klasa umjetne neuronske mreže (ANN), koja se najčešće primjenjuje za analizu vizualnih slika. U Kerasu možemo jednostavno slagati slojeve dodavanjem željenog sloja jedan po jedan.Upravo to ćemo učiniti ovdje: prvo ćemo dodati prvi konvolucijski sloj pomoću Conv2D(). Ovu funkciju koristimo jer radimo sa slikama. Zatim dodajemo funkciju aktivacije Leaky ReLU koja pomaže mreži da nauči nelinearne granice odlučivanja. Budući da imamo dvije različite klase (nosi masku ili ne nosi masku), trebat će nam nelinearna granica odluke koja bi mogla odvojiti ove dvije klasa koje nisu linearno odvojive. Zatim ćemo dodati sloj max-pooling s MaxPooling2D() i tako dalje. Posljednji sloj je gusti sloj koji ima softmax funkciju aktivacije s 2 jedinice, što je potrebno za ovaj problem klasifikacije 2 klase.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Nakon što je model stvoren, kompajlirat ćemo ga pomoću Adam optimizera, jednog od najpopularnijih algoritama optimizacije. Dodatno, specificirat ćemo vrstu gubitka koja je kategorička unakrsna entropija koja se koristi za višeklasnu klasifikaciju. Na kraju, specificirat ćemo metriku kao točnost koju želimo analizirati dok se model obučava.



Nakon što se izvršilo treniranje modela, prikazat ćemo gubitke i točnost modela kao i matricu zabune. Grafove smo dobili sljedećim kodom.

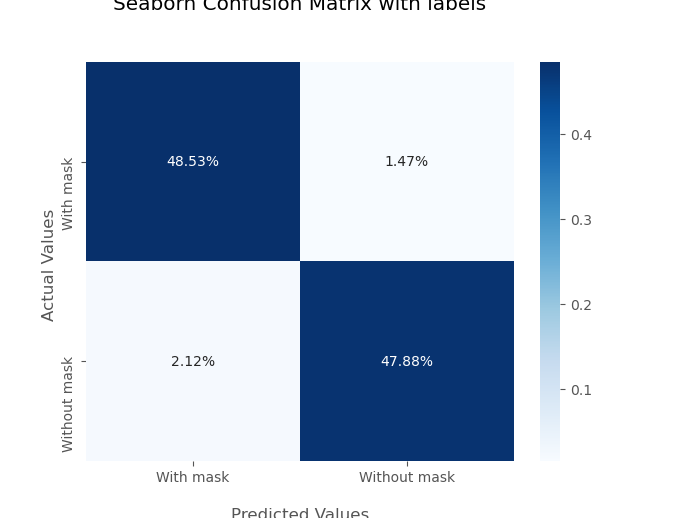
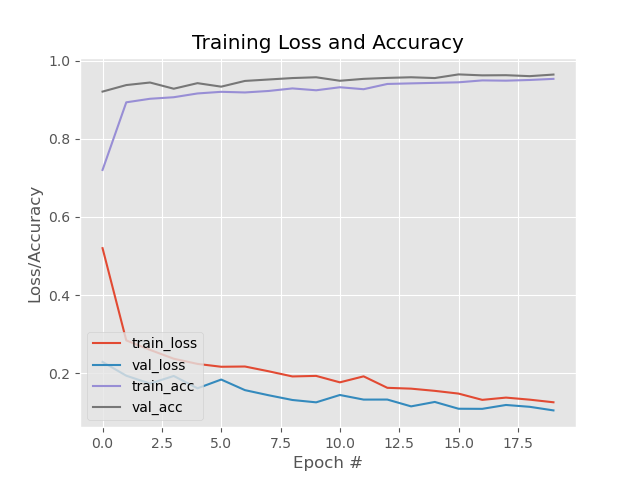
Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Graf gubitka i točnosti te matrica zabune:



## **TRANSFER LEARNING MREŽA**

Transfer leaerning (TL) je istraživački problem u strojnom učenju (ML) koji se usredotočuje na pohranjivanje znanja stečenog tijekom rješavanja jednog problema i njegovu primjenu na drugi sličan problem. Na primjer, znanje stečeno tijekom učenja prepoznavanja automobila moglo bi se primijeniti kada pokušavate prepoznati kamione. Postoji mnogo unaprijed obučenih modela, pa će pronalaženje pravog za rješavanje problema zahtijevati istraživanje. Keras nudi širok raspon unaprijed obučenih modela koji se mogu koristiti za prijenos učenja, izdvajanje značajki, fino podešavanje i predviđanje. Primjer Keras aplikacije je arhitektura Xception. Ovako smo inicijalizirali Xception na ImageNet-u.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Kao i kod razvijanja konvolucijske mreže, i kod transfer learning metode smo augmentirali podatke.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Sekvencijalni model je dobar za korištenje kao običan stog slojeva, što podrazumijeva kada svaki sloj ima jedan ulazni i jedan izlazni tenzor.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Sljedeći korak je izrada modela. Prvo ćemo započeti s instanciranjem osnovnog modela, Xceptiona, i učitavanjem unaprijed obučenih utega u njega. U ovom primjeru koristimo ImageNet. Zatim prelazimo na zamrzavanje svih slojeva u osnovnom modelu. Zamrzavanje pomaže u sprječavanju ažuriranja utega tijekom treninga.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Sljedeći korak je stvaranje novog sloja na vrhu zamrznutih slojeva, koji će naučiti svoje znanje o starim značajkama i koristiti to za određivanje predviđanja na novom skupu podataka. Velika je vjerojatnost da će trenutni izlaz na unaprijed obučenom modelu i izlaz koji želite od svog modela biti različiti, stoga će dodavanje novih slojeva poboljšati model općenito.

Slika na kojoj se prikazuje tekst

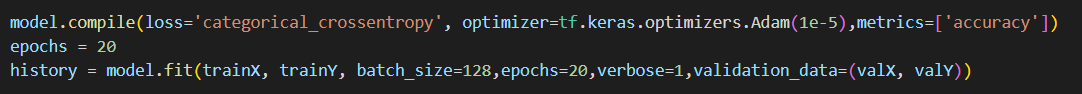
Opis je automatski generiran

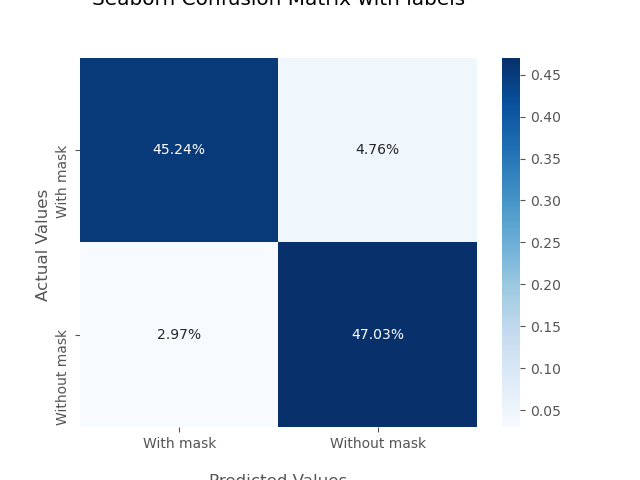
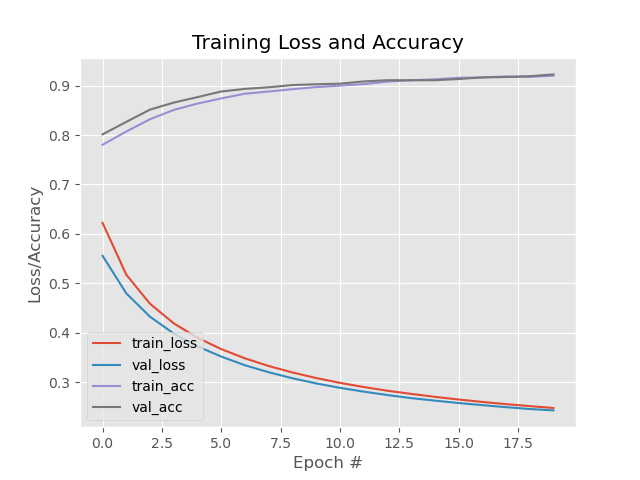
Dakle, nakon što smo pokrenuli model sa zamrznutim slojevima, moramo pokrenuti model s nezamrznutim osnovnim modelima, što bitno poboljšava model, uz nižu stopu učenja. Želimo smanjiti količinu prekomjernog opremanja, pa poduzmimo ovaj korak polako i odmrznimo osnovni model.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

Ponovno ćemo kompalirati i trenirati model te prikazati matricu zabune kao i točnost i pogreške.





Iz grafova možemo zaključiti da je bolje koristiti konvolucijsku mrežu koju smo kreairali nego mrežu koju smo izradili pomoću transfer learning metode.

# **TESTIRANJE RJEŠENJA NA VLASTITOM SKUPU PODATAKA**

Kao vlastiti skup podataka koristili smo slike prijatelja. Skup se sastoji od 5 različitih ljudi od kojih imamo po 5 slika bez maske i 5 s maskom u različitim osvjetljenjima, kutovima, frizurama itd. U nastavku je prikazano par primjera.



Slike smo podijelili u dva direktorija; slike s maskom i slike bez maske. Koristili smo sljedeći kod kako bi testirali model.

Slika na kojoj se prikazuje tekst

Opis je automatski generiran

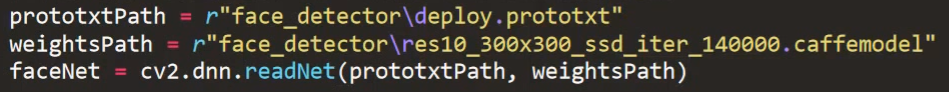
Testiranje modela dalo nam je dosta odstupanja. Oko 30% slika bez maske identificirano je kao slika s maskom. U nastavku su primjeri slika koje su krivo klasificirane.



Bolje rezultate dobili smo koristeći se integriranom web kamerom.

# **KORIŠTENJE MODELA NA SNIMCI S KAMERE**

S diska učitamo model detekcije lica i maske.



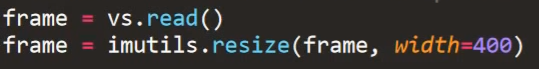


Pokrećemo kameru slijedećom naedbom:



Započinjemo loop petlju koja nas vodi kroz videoprijenos.

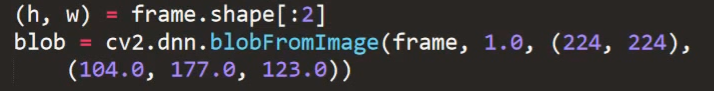
Video spremamo u niz kao slike.



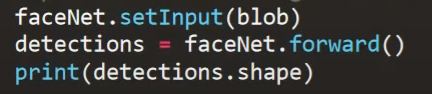
Sada imamo sve potrebne varijable za izradu nove funkcije koja nam vraća dvije informacije: koordinate kvadrata koji prati lice i sigurnost programa da osoba na videozapisu nosi ili ne nosi masku.

U funkciji radimo slijedeće:

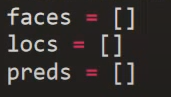
uzimamo dimenzije okvira i iz njega konstruiramo “blob”,



napravljeni “blob” pustimo kroz mrežu i dobijemo detekciju lica,



inicijaliziramo listu dobivenih lica, njihove lokacije i pretpostavku o nošenju maske,



napravimo loop za sve detekcije,



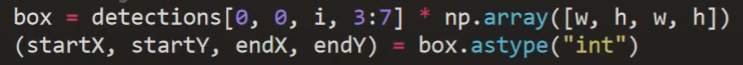
izvučemo sigurnost za svaku detekcijiu,



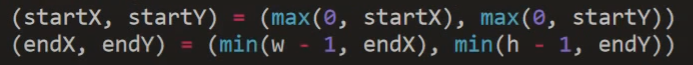
filtritamo slabe detekcije,



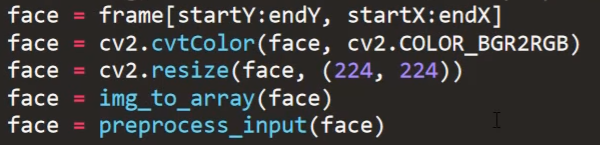
izračunamo koordinate kvadrata oko objekta,



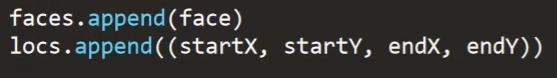
osiguravamo da kvadrati odgovaraju dimenzijama okvira,



izvučemo ROI lica, prebacimo iz BGR u RBG, promijenimo veličinu i preprocesiramo ,



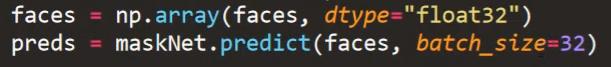
dodamo lica i kvadrate odgovarajućim listama,



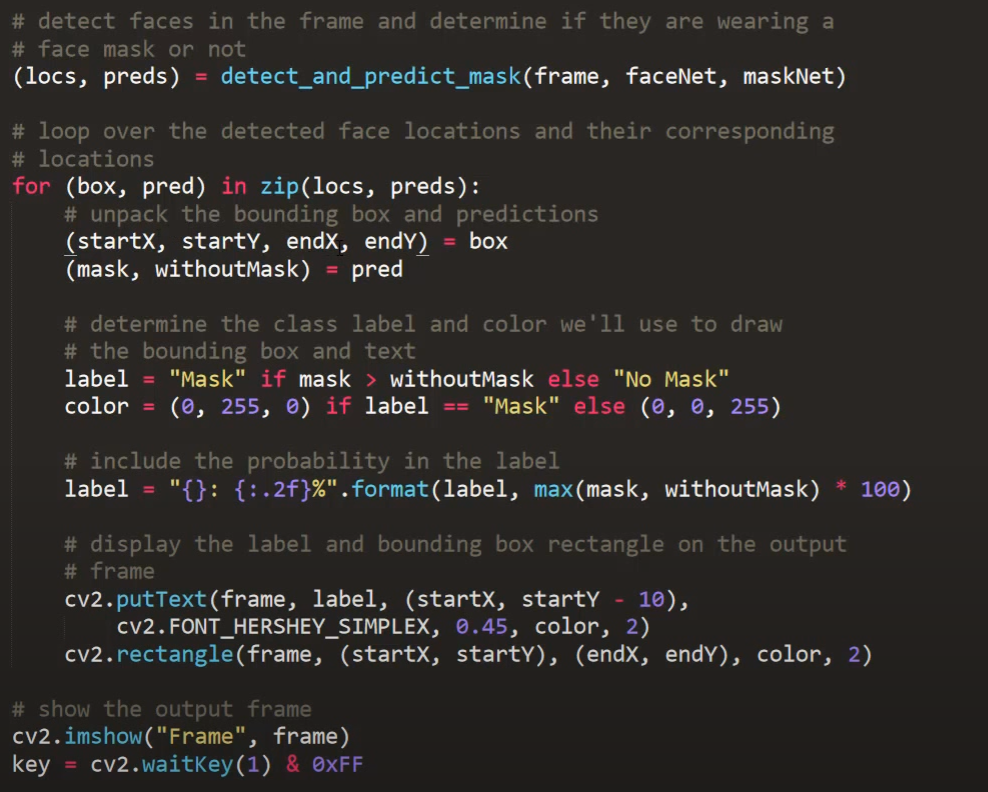
napravimo pretpostakve ako je otkriveno barem jedno lice,



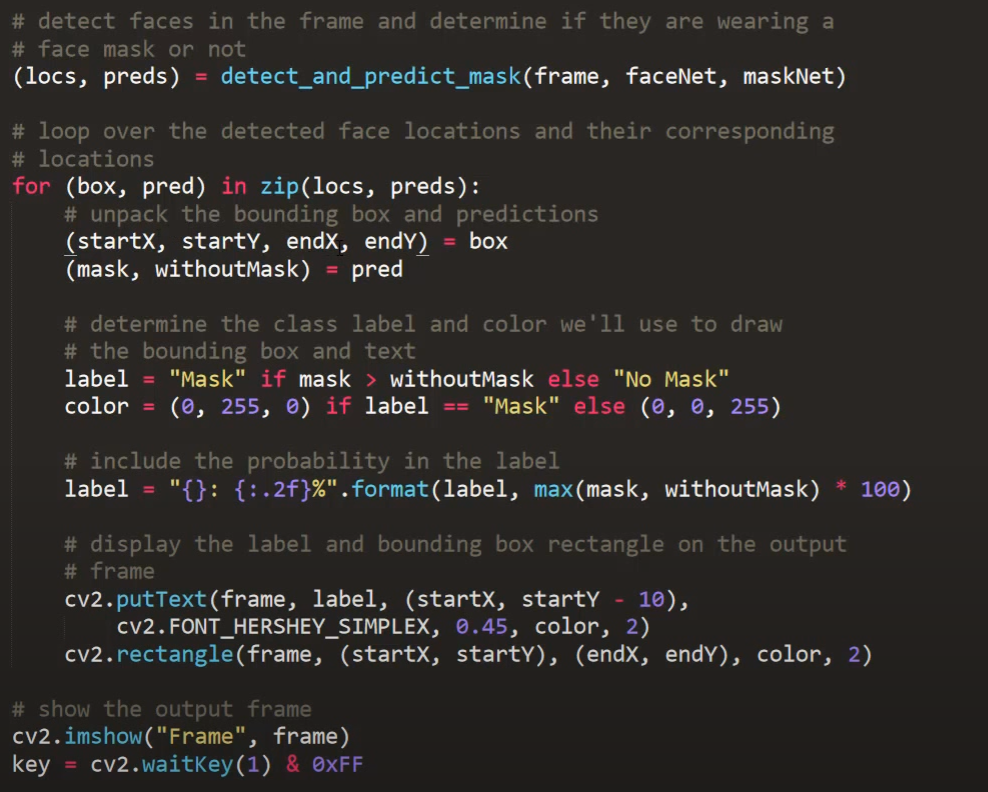
napravimo predikcije batch-eva na svim objektima odjednom,



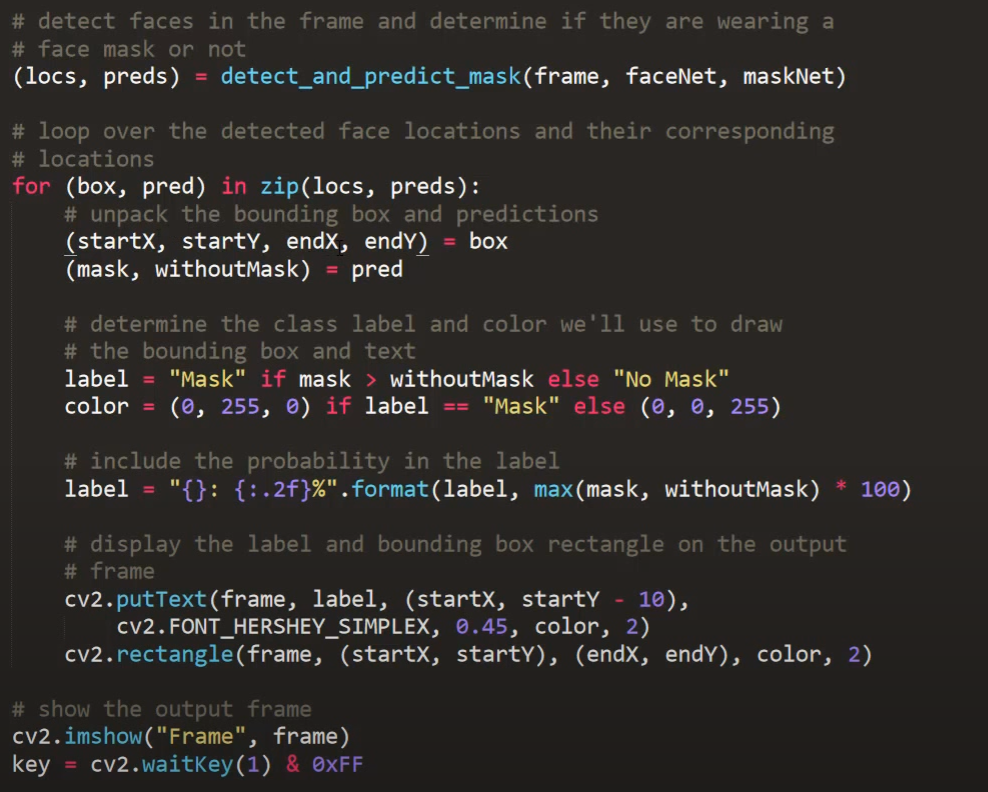
Tu funkciju sada možemo koristiti za detekciju lica u okviru i odrediti ako nose masku.



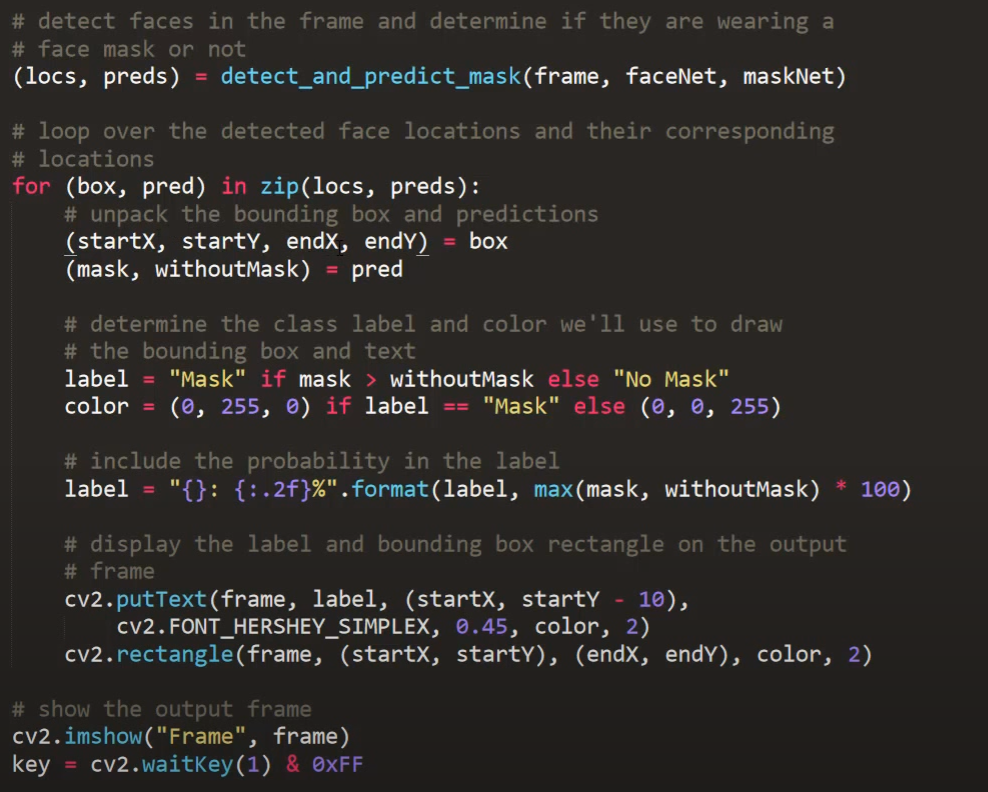
Prođemo kroz detektirana lica i njihove lokacije.



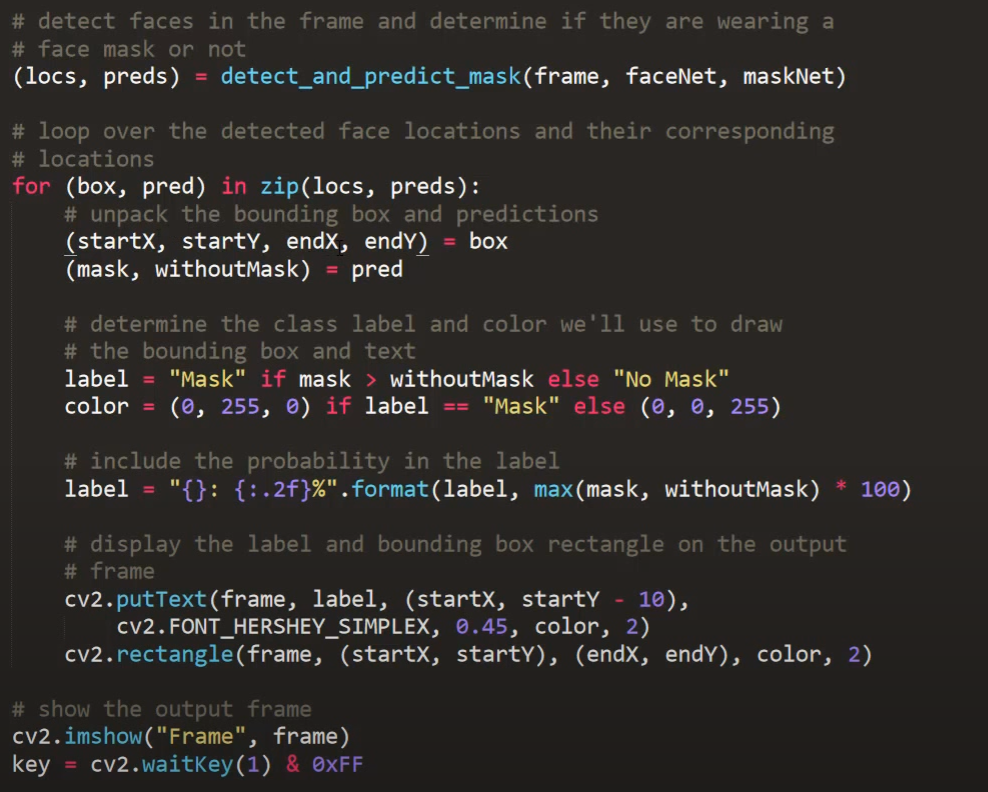
Otpakiramo kvadrat i pretpostavke.



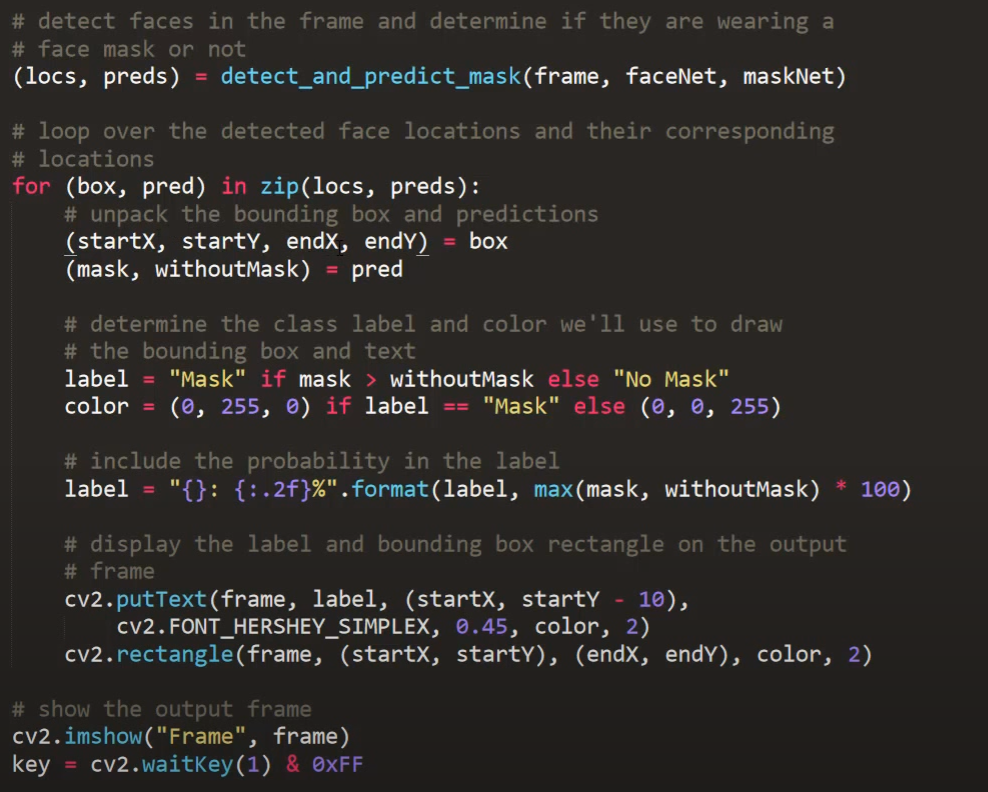
Odredimo oznaku i boju kojom ćemo označiti kvadrat.



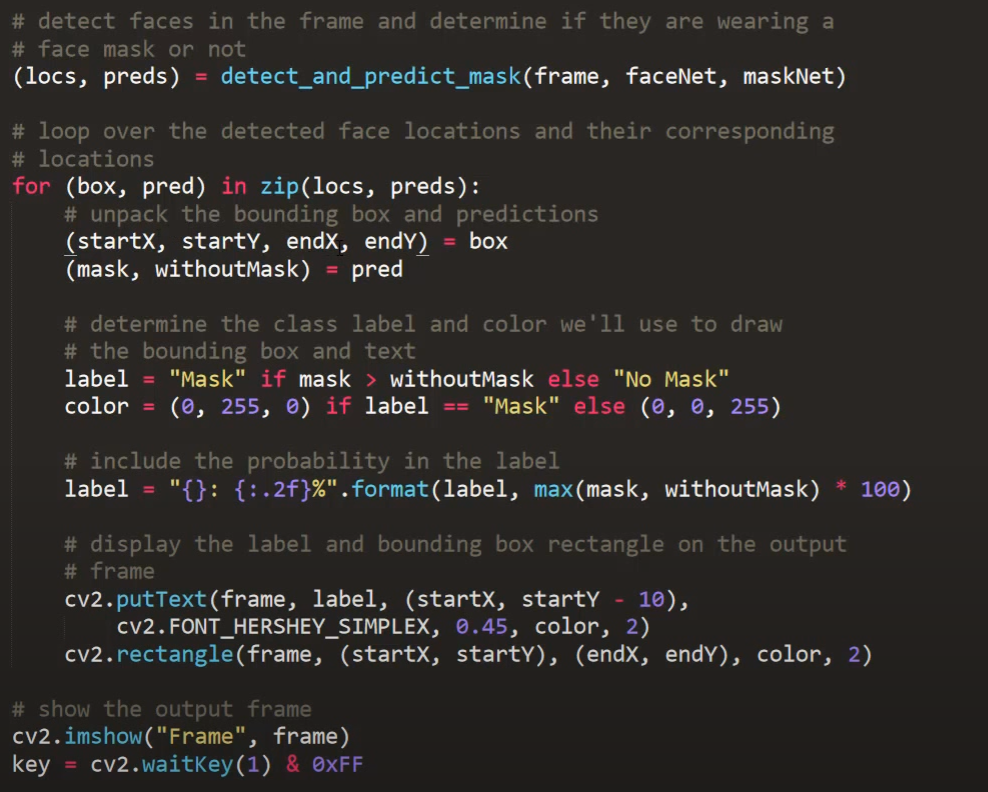
U oznaku postavimo i vjerojatnost.



Postavimo oznaku i kvadrat na output.



Pokažemo output.



Pritiskom na tipku izlazimo iz loop petlje i program završava.



# **TESTIRANJE POMOĆU KAMERE**

Naposljetku, trebalo je još samo testirati program.

1 osoba:

Slika na kojoj se prikazuje tekst, osoba, na zatvorenom

Opis je automatski generiranSlika na kojoj se prikazuje na zatvorenom, osoba, četkica za zube, zeleno

Opis je automatski generiran

Brzina kada program detektira jednu osobu: 0.0937495231628418 sekundi.

2 osobe:

Slika na kojoj se prikazuje tekst, osoba

Opis je automatski generiranSlika na kojoj se prikazuje tekst, na zatvorenom, osoba, zeleno

Opis je automatski generiran

Brzina kada program detektira dvije osobe: 0.11070394515991211sekundi.

Kao što vidimo, nema velike razlike u brzini prilikom detekcije više lica.

# **PRIMJENA**

Ovaj projekt može imati puno različitih primjena pogotovo u bolnicama, no možda najvažnija bi bila sad usred pandemije. Pomoću programa za prepoznavanje maski, puno lakše bi se moglo pratiti kršenje mjera te nepravilno korištenje maski unutar javnih objekata. To bi dovelo do više regulacija što bi rezultiralo manjim brojem zaraza unutar zatvorenih objekata. Također bi se punio državni proračun zbog više izdanih novčanih kazni.

# **LITERATURA**

<https://www.datacamp.com/tutorial/convolutional-neural-networks-python>

<https://www.kdnuggets.com/2022/01/transfer-learning-image-recognition-natural-language-processing.html>

<https://github.com/balajisrinivas/Face-Mask-Detection>

<https://medium.com/analytics-vidhya/image-augmentation-9b7be3972e27>

<https://www.kaggle.com/datasets/vijaykumar1799/face-mask-detection?resource=download>