Bihevioralna autentifikacija temeljem korištenja Android aplikacije

Rašetina, Matia

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Diplomski studij računarstva

Diplomski rad

Bihevioralna autentifikacija temeljem korištenja Android aplikacije

Rijeka, rujan 2024.

Matia Rašetina
0069088300

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
Diplomski studij računarstva

Diplomski rad

Bihevioralna autentifikacija temeljem korištenja Android aplikacije

Mentor: izv. prof. dr. sc. Jonatan Lerga
Komentor: dr. sc. Franko Hržić

Rijeka, rujan 2024.

Matia Rašetina
0069088300

Umjesto ove stranice umetnuti zadatak za završni ili diplomski rad

Izjava o samostalnoj izradi rada

Izjavljujem da sam samostalno izradio ovaj rad.

Rijeka, rujan 2024.

_____

Ime Prezime

Sadržaj

Poglavlje 1

Uvod

Porast digitalizacije u bankarstvu i smanjenje popularnosti gotovinskog plaćanja potaknuli su razvoj mnogobrojnih mobilnih aplikacija. Na današnjem tržištu, može se pronaći mnogo aplikacija koje omogućuju mobilno bankarstvo.[1] Tradicionalne metode autentifikacije, poput lozinki i osobnih identifikacijskih brojeva (eng. Personal Identification Number, PIN), sve su ranjivije na različite oblike kibernetičkih napada. Kao odgovor na ove izazove, bihevioralna autentifikacija na temelju korištenja mobilnih aplikacija postaje sve popularnija metoda za povećanje sigurnosti i poboljšanje korisničkog iskustva.

Bihevioralna autentifikacija je metoda sigurnosne provjere identiteta koja koristi jedinstvene obrasce ponašanja korisnika za autentifikaciju. Za razliku od tradicionalnih metoda koje se oslanjaju na znanje korisnika (kao što su lozinke), posjedovanje (poput pametnih kartica) ili osobne karakteristike korisnika (kao što su otisak prsta ili rožnica), bihevioralna autentifikacija prati kako korisnik koristi određenu aplikaciju i/ili uređaj. Ova tehnologija analizira različite aspekte korisničkog ponašanja, kao što su način tipkanja, brzina unosa teksta, ritam i pritisak prstiju na zaslonu, obrasci pomicanja, način na koji korisnik drži uređaj te kretanje i korištenje senzora poput akcelerometra i žiroskopa. Kroz strojno učenje i napredne algortime temeljem prikupljenih podataka od strane sustava i/ili aplikacija bihevioralne autentifikacije stvaraju se jedinstveni profili korisnika. Kada korisnik pristupa aplikaciji ili sustavu, njegovo ponašanje se uspoređuje s pohranjenim profilom, te se na temelju podudarnosti donosi odluka o autentifikaciji. Ovaj pristup omogućava kontinuiranu

autentifikaciju, što znači da se identitet korisnika može provjeravati neprekidno tijekom korištenja aplikacije, čime se značajno smanjuje rizik od neovlaštenog pristupa i poboljšava sigurnost. [2]

Prednosti bihevioralne autentifikacije su višestruke. Povećana sigurnost je jedna od glavnih prednosti, jer bihevioralna autentifikacija pruža dodatni sloj sigurnosti koji je teško prevariti, za razliku od tradicionalnih metoda koje se oslanjaju na statičke podatke. Osim što se ovom metodom daje dodatni sloj sigurnosti, prednost ove metode jest ta što je neprimjetna. Drugim riječima, provjera identiteta korisnika odrađuje se u pozadini, kroz uobičajenu korisničku interakciju te se na taj način korisnika ne prekida u zadacima koje odrađuje, što ubrzava vrijeme obavljanja zadatka i poboljšava korisničko iskustvo. Kontinuirana autentifikacija omogućuje stalno praćenje i potvrdu identiteta korisnika tijekom korištenja aplikacije, što dodatno povećava sigurnost.

Prema istraživanjima [3, 4], bihevioralna autentifikacija može značajno smanjiti rizik od neovlaštenog pristupa. Studije pokazuju da su bihevioralni podaci vrlo učinkoviti u prepoznavanju korisnika, s točnošću većom od 95% u mnogim slučajevima [3]. Također, prema istraživanju tvrtke MarketsandMarkets, tržište bihevioralne biometrike očekuje se da će rasti s godišnjom stopom rasta (CAGR) od 23,9% od 2020. do 2025. godine, dosežući tržišnu vrijednost od preko 2 milijarde dolara.[4]

Projekt izrade modela umjetne inteligencije za bihevioralnu autentifikaciju temelji se na prikupljanju i analizi podataka o korištenju aplikacije i uređaja. Odabir korištenih podataka zasnivati će se na kvaliteti i količini sadržane informacije svakog podatka. Prikupljeni podaci mogu uključivati:

- Način unosa lozinke ili PIN-a
- Korištenje senzora (akcelerometar, žiroskop)
- Obrasce dodirivanja zaslona
- Lokacijske podatke

Nakon prikupljanja podataka, razvija se model strojnoga učenja koji kao cilj ima prepoznati korisnika temeljem prikupljenih podataka. Algoritmi poput neuronskih mreža, stroja s potpornim vektorima (eng. Support Vector machine) i drugih tehnika

strojnog učenja koriste se za analizu podataka i prepoznavanje jedinstvenih obrazaca korisnika.

U ovom radu istražuje se hipoteza da je moguće autenticirati korisnika mobilnog uređaja tijekom korištenja specifične mobilne aplikacije namijenjene bankarskim aktivnostima. Kako bi se testirala navedena hipoteza, postavljeni su sljedeći koraci:

1. Prikupljanje podataka uz pomoć Android mobilne aplikacije
2. Procesiranje i vizualizacija podataka
3. Korištenje ANOVA Repeated Measurement statističkog testa, kako bi se doznalo ako postoji signifikantna statistička razlika između podataka korisnika
4. Priprema podataka za izradu modela umjetne inteligencije
5. Evaluacija rezultata korištenih modela umjetne inteligencije

Nastavak rada je organiziran na sljedeći način. U drugom poglavlju, opisuje se eksperiment te korištene tehnologije i metode, zatim u trećem poglavlju, pobliže će biti opisani modaliteti mobilne aplikacije i prikupljeni podaci. Nadalje, u četvrtom poglavlju, izvedba ANOVA Repeated Measurement statističkog testa će biti opisana, zajedno sa svim procesima pripreme podataka za treniranje modela umjetne inteligencije. Peto poglavlje sadrži opis treniranja modela umjetne inteligencije, šesto poglavlje sadrži rezultate istraživanja te naposljetku će se interpretirati rezultati zajedno s zaključkom ovog rada i opisom budućeg rada.

Python kod korišten u ovom radu je javno dostupan na `https://github.com/mate329/master-thesis`.

Poglavlje 2

Opis eksperimenta, korištene tehnologije i metode

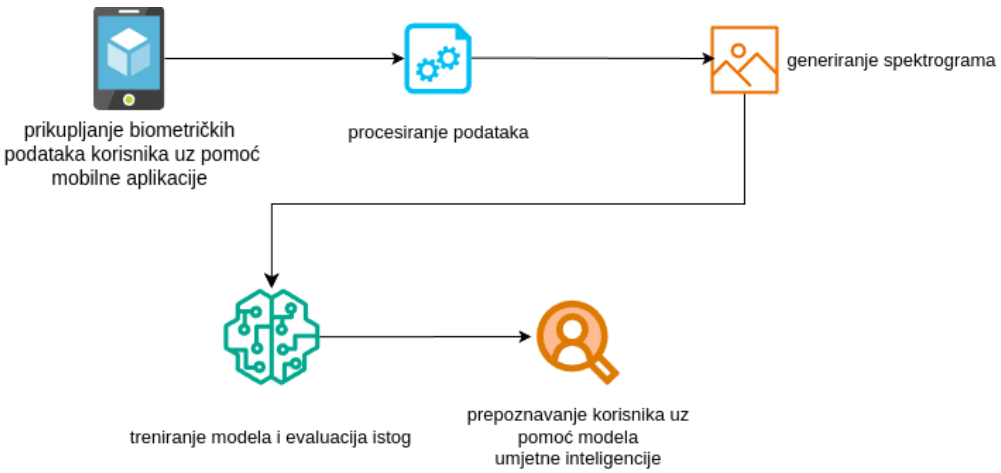
U nastavku ovog poglavlja, opisan je eksperiment za dobivanje biometričkih podataka korisnika, zatim korištene tehnologije i metode.

2.1 Opis eksperimenta

Mobilne aplikacije postale su sastavni dio svakodnevnog života, a mobilno bankarstvo je jedna od najčešće korištenih usluga.[5] [6] U svrhu znanstvenog istraživanja, razvijena je aplikacija koja simulira mobilno bankarstvo te prikuplja podatke iz senzora mobilnog uređaja. Cilj eksperimenta, na kojem je ovaj projekt baziran, je identificirati podatke i senzore koji se mogu koristiti za prepoznavanje korisnika, poboljšavajući tako sigurnost i korisničko iskustvo.

U ovom poglavlju, predstavit će se izrađena mobilna aplikacija dizajnirana za prikupljanje podataka sa različitih senzora mobilnog uređaja s ciljem prepoznavanja korisnika. Aplikacija simulira funkcionalnosti mobilnog bankarstva te prikuplja podatke o korištenju ekrana osjetljivog na dodir, akcelerometra i žiroskopa. Različiti moduli unutar aplikacije omogućuju detaljnu analizu načina na koji korisnici tipkaju, pišu PIN, rade akcije premotavanja (eng. *scroll*) i potezanja (eng. *swipe*) kroz tekstove.

Na sljedećoj slici je opisan tok istraživanja ovog rada. Svaki korak će u nastavku rada biti detaljno opisan.



Slika 2.1 Dijagram toka istraživanja

2.2 Korišteni programski alati: Python

Python je programski jezik koji je Guido van Rossum razvio 1990. godine. Za razliku od jezika poput C, C++ i Java, koji se prevode u binarni kod uz pomoć prevodilaca, Python je interpretirani jezik. Također, Python spada među objektno orijentirane jezike kao što su C++, Java, Javascript i C# [7]. Python se ističe kao jedan od najraširenijih programskih jezika za razvoj različitih aplikacija zbog svoje jednostavnosti, fleksibilnosti i širokog spektra knjižnica koje pokrivaju gotovo sve aspekte programiranja. Iz ovih razloga, Python je korišten u ovome projektu.

2.2.1 Korištene programske knjižnice

Tablica 2.1 opisuje knjižnice korištene za manipulaciju i analizu podatka, zatim tablica 2.2 opisuje knjižnice korištene za strojno i duboko učenje, tablica 2.3 opisuje knjižnice korištene za računalni vid i obradu slika, tablica 2.4 opisuje knjižnice korištene za vizualizaciju dobivenih podataka te posljednja tablica 2.5 opisuje knjižnice koje su ugrađene u Python programski jezik.

Tablica 2.1 Knjižnice za manipulaciju i analizu podataka

Knjižnica	**Verzija**	**Opis**
Pandas	2.0.3	Knjižnica za manipulaciju i analizu tabličnih podataka, omogućava filtriranje, agregaciju i transformaciju podataka. Pandas je neophodan alat za bilo kakvu analizu podataka u Pythonu zbog svoje fleksibilnosti i efikasnosti. [8]
Numpy	1.24.4	Knjižnica za rad s velikim, višedimenzionalnim nizovima i matematičkim operacijama na njima, temelj za znanstvene proračune. [9]

Tablica 2.2 Knjižnice za strojno učenje i duboko učenje

Knjižnica	**Verzija**	**Opis**
Torch	2.3.1	Framework za duboko učenje, omogućava definiranje i treniranje neuralnih mreža, pruža podršku za rad s GPU-om. Torch je popularan zbog svoje fleksibilnosti i performansi. [10]
Scikit-learn	1.3.2	Knjižnica za strojno učenje s alatima za analizu podataka i modeliranje, izgrađena na NumPy, SciPy i Matplotlib. [11]

Tablica 2.3 Knjižnice za računalni vid i obradu slika

Knjižnica	**Verzija**	**Opis**
Torchvision	0.18.1	Paket za rad s vizualnim podacima, pristup datasetovima i modelima za računalni vid, omogućava implementaciju naprednih modela dubokog učenja. [12]
Pillow	10.3.0	Knjižnica za otvaranje, manipulaciju i spremanje slikovnih datoteka, jednostavna za korištenje i fleksibilna. [13]

Tablica 2.4 Knjižnice za grafičko prikazivanje podataka

Knjižnica	**Verzija**	**Opis**
Plotly	5.22.0	Knjižnica za kreiranje interaktivnih grafova i vizualizacija podataka, omogućava integraciju u web aplikacije. [14]
Matplotlib	3.7.5	Knjižnica za crtanje statičkih, animiranih i interaktivnih grafova, pruža izuzetno fleksibilne i moćne alate za vizualizaciju podataka. [15]

Tablica 2.5 Standardne Python knjižnice za osnovne funkcionalnosti

Knjižnica	**Opis**
Logging	Knjižnica za evidentiranje događaja, omogućava praćenje i bilježenje različitih događaja i poruka unutar aplikacije. [16]
Os	Knjižnica za interakciju s operativnim sustavom, omogućava rad s datotekama i direktorijima, te izvršavanje sistemskih naredbi.
Collections	Modul koji pruža specijalizirane tipove kontejnera za napredne strukture podataka kao što su namedtuple, deque, Counter i OrderedDict.
Glob	Modul za pretraživanje datotečnog sustava prema uzorcima, pronalazi putanje koje odgovaraju specificiranom uzorku.
Time	Knjižnica za rad s vremenskim podacima, mjerenje vremena izvršavanja koda i rad s vremenskim intervalima.

2.3 ANOVA statistički test

ANOVA, skraćeno od eng. *Analysis of Variance*, je statistička metoda koja se koristi za testiranje razlika između sredina više grupa. Osnovna ideja ANOVA-e je da podijeli ukupnu varijabilnost unutar skupa podataka na varijabilnost između grupa i varijabilnost unutar grupa. [17]

Postoji nekoliko vrsta ANOVA-e, uključujući [18]:

- **Jednosmjerna ANOVA (One-way ANOVA)**: Koristi se kada se ispituje utjecaj jednog faktora na varijablu ishoda.
- **Dvostruka ANOVA (Two-way ANOVA)**: Koristi se kada se ispituje utjecaj dva faktora i njihova interakcija na varijablu ishoda.
- **Ponovljena mjerenja ANOVA (Repeated Measures ANOVA)**: Koristi se kada se ista grupa subjekata mjeri više puta pod različitim uvjetima.

ANOVA se temelji na usporedbi srednje vrijednosti među grupama pomoću omjera između varijabilnosti između grupa i varijabilnosti unutar grupa. Taj omjer se naziva *F-statistika*, a izračunava se prema formuli [17]:

$$F = \frac{\mathrm{MS}_{\text{između}}}{\mathrm{MS}_{\text{unutar}}}$$

gdje je $\mathrm{MS}_{\text{između}}$ srednja kvadratna vrijednost između grupa, a $\mathrm{MS}_{\text{unutar}}$ srednja kvadratna vrijednost unutar grupa. Ako je F-statistika veća od kritične vrijednosti iz F-raspodjele, odbacujemo nultu hipotezu koja kaže da nema razlike između grupa. U ovom projektu, kritična vrijednost je 0.05.

U ovom radu, ANOVA Repeated Measurement je statistički test za koji je korišten za analizu podataka prikupljenih iz mobilne aplikacije. Cilj je bio utvrditi postoji li značajna razlika između podataka dobivenih od korisnika tijekom različitih dana.

2.4 Generiranje reprezentacija podataka pomoću spektrograma

Spektrogram je vizualni prikaz spektra frekvencija signala kako se one mijenjaju kroz vrijeme. Koristi se za analizu promjena u frekvencijskom sadržaju signala tijekom vremena.

Jedna od metoda izrade spektrograma je primjenom *Fourierove transformacije* na kratke, preklapajuće segmente signala. Ovaj postupak se naziva *kratkotrajna Fourierova transformacija* (eng. Short-time Fourier Transform, STFT). Rezultat STFT-a je kompleksna vrijednost koja predstavlja amplitudu i fazu frekvencije u određenom vremenskom intervalu [19].

Formula kratkotrajne Fourierove transformacije je:

$$X(m,n) = \sum_{k=-\infty}^{\infty} x(k)\, w(k-m)\, e^{-j\omega_n k}$$

[20]

Ovdje su varijable definirane na sljedeći način:

- $X(m,n)$: Kompleksni broj koji predstavlja STFT koeficijent u vremenskom indeksu m i frekvencijskom indeksu n.
- $x(k)$: Ulazni signal koji se analizira. Ovaj signal može sadržavati informacije u vremenskoj domeni.
- $w(k-m)$: Prozorska funkcija koja se koristi za ograničavanje analize na mali vremenski interval oko indeksa m. Ova funkcija pomaže lokalizirati analizu u vremenskom domenu.
- ω_n: Normalizirana kutna frekvencija koja odgovara frekvencijskom indeksu n. Koristi se za transformaciju signala iz vremenske u frekvencijsku domenu.
- $e^{-j\omega_n k}$: Kompleksni eksponencijalni član koji predstavlja fazu signala pri frekvencijskom indeksu n.

Proces izrade spektrograma uz pomoć kratkotrajne Fourierove transformacije uključuje sljedeće korake:

1. **Segmentacija**: Signal se dijeli na kratke segmente.
2. **Primjena prozorske funkcije**: Na svaki segment primjenjuje se prozorska funkcija kako bi se smanjili rubni efekti.
3. **Fourierova transformacija**: Na svaki segment se primjenjuje Fourierova transformacija.
4. **Kombiniranje rezultata**: Amplitude frekvencija iz svakog segmenta se kombiniraju i prikazuju u obliku slike.

Spektogrami su pronašli široku upotrebu u različitim domenama znanosti, počevši od obrade govora [21] i glazbe [22] pa sve do medicine i dijagnostike putem EEG signala [23].

2.5 Aplikacija za prikupljanje podataka

U svrhu prikupljanja podataka korištena je Android aplikacija koja je mjerila podatke poput dužine i brzine odvijanja korisnikove akcije premotavanja i potezanja, veličine prsta te prikupljala podatke sa senzora akcelerometra i žiroskopa. Navedena aplikacija nije bila razvijena tijekom ovog istraživanja, već je preuzeta postojeća implementacija. Ovu aplikaciju su izradili profesor Sandi Ljubić i asistent Alen Salkanović Tehničkog fakulteta u Rijeci. Više o samom eksperimentu i procesu prikupljanja podataka bit će opisano u poglavlju "Korišteni skup podataka" 3 *Korišteni skup podataka*. Ova aplikacija je u potpunosti korištena kao alat za prikupljanje podataka te njezino razvijanje nije dio niti jedne faze ovog diplomskog rada.

2.6 Umjetna inteligencija

Umjetna inteligencija je područje računalne znanosti usmjereno na stvaranje tehnologija koje mogu izvoditi kompleksne zadatke poput analize podataka, učenja i donošenja odluka. To uključuje zadatke poput prepoznavanja govora, razumijevanja prirodnog jezika, prepoznavanja objekata na slikama, donošenja odluka i igranja igara [24].

Umjetna inteligencija je ključna za mnoge industrije jer može analizirati velike količine podataka i pružiti vrijedne uvide i točna predviđanja koja pomažu u donošenju informiranih odluka. U zdravstvu, primjenjuje se za dijagnostiku bolesti, personaliziranu medicinu i robotsku kirurgiju [25]. U financijama, duboko učenje omogućuje algoritamsko trgovanje, procjenu kreditnog rizika i otkrivanje prijevara.[26] U transportu, koristi se za autonomna vozila, optimizaciju ruta i predikciju prometa.[27]. Sve ove primjene ilustriraju važnost dubokog učenja u modernom svijetu i njegovu sposobnost da poboljša različite aspekte poslovanja i svakodnevnog života. U ovome projektu, umjetna inteligencija se koristila kao alat za prepoznavanje biometrike korisnika. [28]

Neuronske mreže su sofisticirani matematički modeli inspirirani funkcioniranjem ljudskog mozga. Sastoje se od umjetnih neurona organiziranih u slojeve, koji zajedno obrađuju ulazne podatke. Glavna svojstva neuronskih mreža uključuju učenje iz podataka, sposobnost prepoznavanja uzoraka te efikasnu obradu kompleksnih informacija. One se široko koriste u područjima kao što su prepoznavanje slika, obrada prirodnog jezika i razumijevanje složenih uzoraka u podacima. [29]

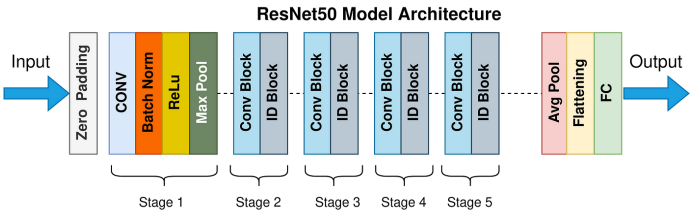
Duboko učenje je odabrano za ovaj projekt zbog njegove sposobnosti da samostalno nauči složene obrasce iz obimnih skupova podataka. To je ključno za analizu dubokih struktura podataka koje su bitne za ciljeve ovog projekta. Korištenjem dubokih neuronskih mreža postiže se optimizacija performansi kroz učenje, što omogućuje visoku točnost i skalabilnost u obradi podataka, ključne za uspješno rješavanje zadanih problema. [30]

2.6.1 ResNet

ResNet (Residual Network) je duboka neuronska mreža koju su razvili istraživači iz Microsoft Researcha. ResNet je poznat po uvođenju *residualnih blokova*, koji omogućuju izgradnju vrlo dubokih mreža (do nekoliko stotina slojeva) bez problema s nestajanjem gradijenata [31].

Arhitektura ResNet-a

ResNet koristi *residualne blokove* koji sadrže identitetske kratke spojeve (*skip connections*). Ovi blokovi omogućuju direktno prosljeđivanje ulaznih podataka slojeva dalje u mrežu, što olakšava treniranje vrlo dubokih modela. Na slici 2.2 se može vidjeti arhitektura ResNet modela.



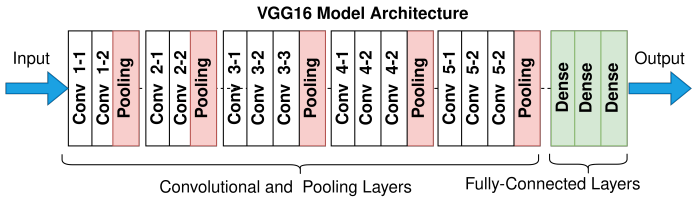
Slika 2.2 Arhitektura ResNet modela [32].

2.6.2 VGG16

VGG16 je duboka konvolucijska neuronska mreža razvijena na Sveučilištu Oxford. VGG16 je poznata po svojoj jednostavnoj i dosljednoj strukturi, koja se sastoji od konvolucijskih slojeva sa malim filtrom (3x3) i maksimalnih pooling slojeva [33].

Arhitektura VGG16

VGG16 ima 16 slojeva s težinama, uključujući 13 konvolucijskih slojeva i 3 potpuno povezana sloja. Struktura mreže je jednostavna, ali učinkovita, što omogućuje postizanje visokih performansi u zadacima računalnog vida [34]. Na slici 2.3 se može vidjeti arhitektura ResNet modela.



Slika 2.3 Arhitektura VGG16 mreže [33] [35]

Primjene VGG16

VGG16 se koristi u različitim zadacima računalnog vida, uključujući:

- **Klasifikacija slika**: Prepoznavanje i klasifikacija objekata unutar slika.
- **Ekstrakcija značajki**: Korištenje pretreniranih modela za ekstrakciju značajki iz slika koje se zatim koriste u drugim modelima.
- **Učenje s prijenosom znanja**: Prilagodba pretreniranih modela na nove zadatke s manjim datasetovima.

Poglavlje 3

Korišteni skup podataka

Prvo će se predstaviti modaliteti mobilne aplikacije koji su korišteni za prikupljanje podataka.

3.0.1 Opis modaliteta mobilne aplikacije

Kao što je već spomenuto, postoje četiri modaliteta koji se koriste u mobilnoj aplikaciji. U nastavku, slikama će se pokazati izgled mobilne aplikacije te definirati podatke koje je mobilna aplikacija u to vrijeme prikupljala i pokazati iste.

Upis teksta i PIN-a

Za upis teksta, korisnika se prvo pita spol te se otvore tri polja za upis teksta, koje korisnik mora ispuniti točno. Nadalje, u modalitetu za unos PIN-a, na ekranu mobilnog uređaja, stvara se tipkovnica s brojevima od 0 do 9 te na ekranu piše PIN koji korisnik treba unesti. Raspored brojeva na tipkovnici je nasumičan. U ovom modalitetu, mobilna aplikacija je prikupljala informacije sa žiroskopa i akcelerometra te prikupljala koordinate dodira na ekranu i mjerila vrijeme između korisnikovih dodira. Detaljniji opis informacije žiroskopa i akcelerometra će se opisati kasnije u radu. Oba modaliteta prikupljaju iste podatke.

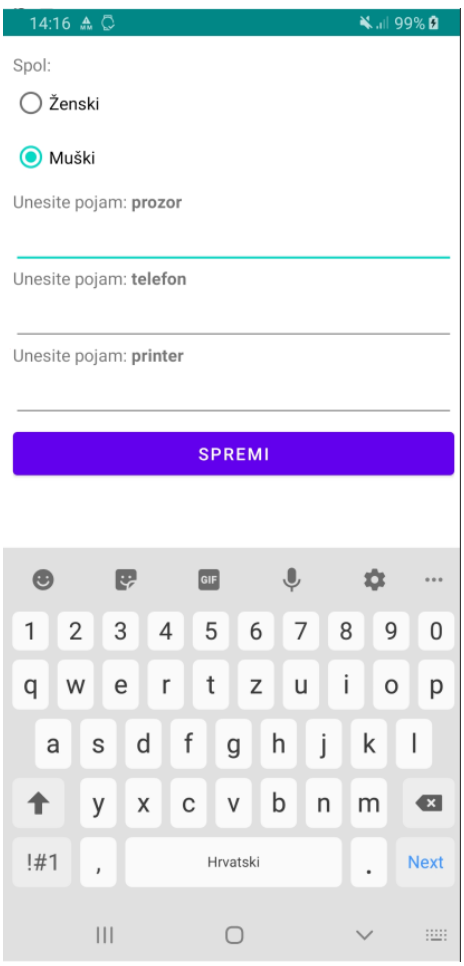
Polja koja se nalaze u Comma Seperated Values (nadalje CSV) datoteci koja sadrži vrijednosti senzora su:

- **Sensor Name**: Ime senzora
- **Sensor Type**: Vrsta senzora
- **Angular Speed X**: Kutna brzina po X osi
- **Angular Speed Y**: Kutna brzina po Y osi
- **Angular Speed Z**: Kutna brzina po Z osi
- **Timestamp**: Vremenska oznaka

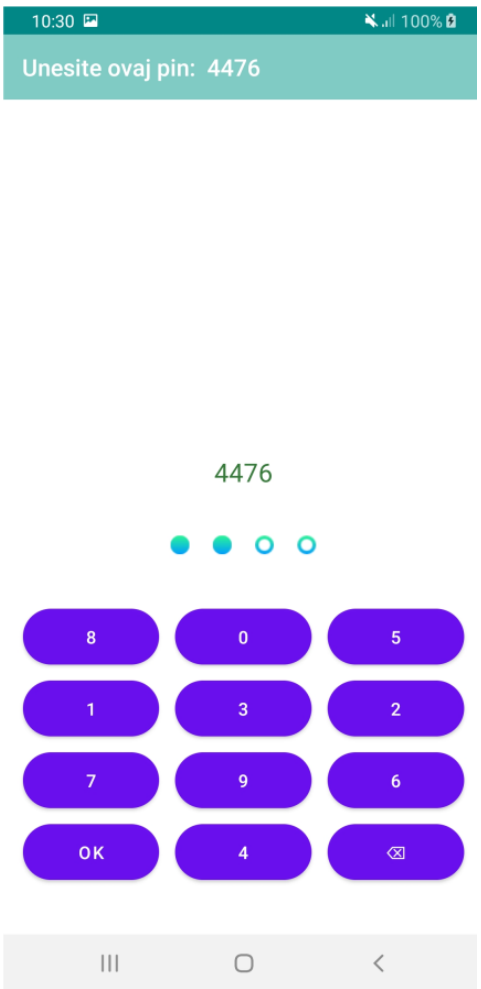
Polja koja se nalaze u CSV datoteci koja sadrži podatke o dodirima na ekranu su:

- **eventRawX**: Dodir na X koordinati ekrana
- **eventRawY**: Dodir na Y koordinati ekrana
- **eventTime**: Vremenska oznaka dodira u milisekundama
- **distance**: Udaljenost između prijašnjeg i trenutnog dodira ekrana
- **time_diff**: Vremenska razlika između prijašnjeg i trenutnog dodira ekrana
- **speed_pixels_per_ms**: Brzina prelaska u pikselima po milisekundi

Vrijedi spomenuti kako mjerenje koordinata kreće iz **gornjeg lijevog** kuta ekrana uređaja. Na slikama 3.1(a) i 3.1(b) se može vidjeti izgled ekrana tijekom modaliteta upisa teksta i upisa PIN-a, zajedno sa primjerima podataka koje mobilna aplikacija prikuplja tijekom spomenutih modaliteta - slike 3.2 i 3.3.



(3.1(a)) Izgled ekrana tijekom upisa teksta



(3.1(b)) Izgled ekrana tijekom upisa PIN-a

Slika 3.1 Prikaz različitih modaliteta: (a) upis teksta, (b) upis PIN-a

Sensor Name	Sensor Type	Angular Speed X	Angular Speed Y	Angular Speed Z	Timestamp
lsm6ds3c	Accelerometer	0.23	3.88	9.54	502668470321477
lsm6ds3c	Gyroscope	-0.06	-0.02	0.03	502668469120332
lsm6ds3c	Accelerometer	0.24	3.89	9.6	502668472724186
lsm6ds3c	Gyroscope	-0.05	-0.01	0.03	502668471522571
lsm6ds3c	Gyroscope	-0.04	0	0.03	502668473925748
lsm6ds3c	Accelerometer	0.24	3.89	9.66	502668475127675
lsm6ds3c	Gyroscope	-0.03	0.01	0.03	502668476329602
lsm6ds3c	Accelerometer	0.24	3.91	9.73	502668477531165
lsm6ds3c	Gyroscope	-0.02	0.02	0.02	502668478732727
lsm6ds3c	Accelerometer	0.25	3.9	9.78	502668479933873
lsm6ds3c	Accelerometer	0.24	3.9	9.79	502668482336530
lsm6ds3c	Gyroscope	-0.01	0.03	0.02	502668481134967
lsm6ds3c	Accelerometer	0.22	3.89	9.82	502668484740071
lsm6ds3c	Gyroscope	0.01	0.04	0.02	502668483538092
lsm6ds3c	Gyroscope	0.02	0.04	0.02	502668485941998
lsm6ds3c	Accelerometer	0.21	3.9	9.82	502668487143561
lsm6ds3c	Gyroscope	0.04	0.05	0.02	502668488345123

Slika 3.2 Primjer prikupljenih podataka sa senzora akcelerometra i žiroskopa

Event Raw X	Event Raw Y	Event Time	Distance	Time_diff	Speed_pixels_
404	806	11410587	227.71	254	0.9
386	1033	11410841	321.34	2728	0.12
603	796	11413569	211.68	349	0.61

Slika 3.3 Primjer prikupljenih podataka o koordinatama dodira ekrana tijekom upisa PIN-a

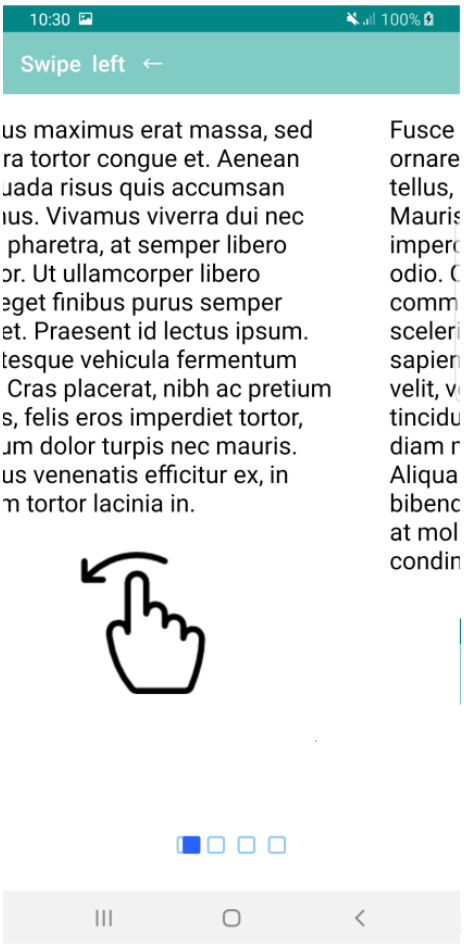
Modaliteti swipe i potezanja

U modalitetima premotavanja (eng. scroll) i potezanja (eng. swipe), korisnik može pregledavati ponuđeni tekst na dva načina: premotavanjem ili potezanjem. U modalitetu premotavanja, korisnik ima zadatak pronaći četiri kućice za potvrdu (eng. checkbox) koje su označene uz tekst "Slažem se s odredbama i uvjetima". Ova interakcija simulira korisnikovo čitanje i prihvaćanje uvjeta korištenja aplikacije. U modalitetu potezanja, korisnik treba što brže doći do zadnje stranice sučelja.

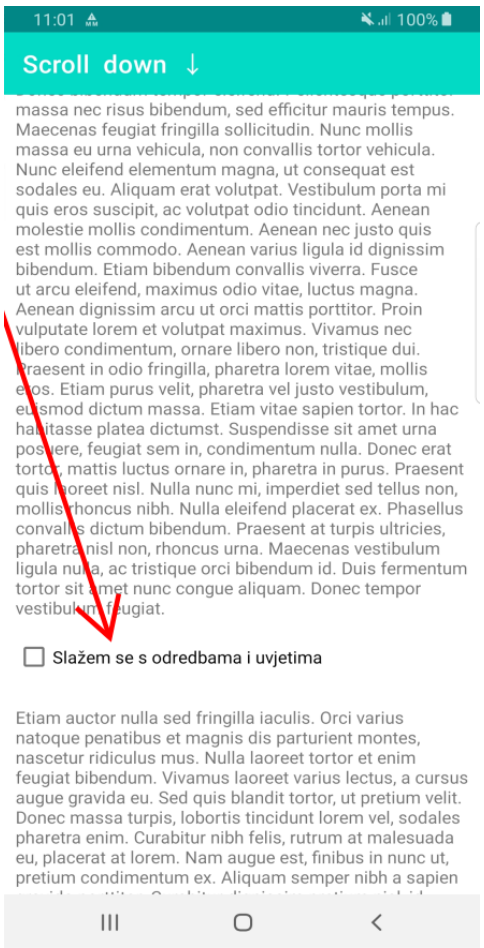
Za ova dva modaliteta, skupljaju se isti podaci:

- **e1RawX i e1RawY**: X i Y koordinate početka akcije
- **e2RawX i e2RawY**: X i Y koordinate kraja akcije
- **e1EventTime**: Vremenska oznaka početka akcije
- **e2EventTime**: Vremenska oznaka kraja akcije
- **distance**: Udaljenost između početka i kraja akcije
- **time_diff**: Vremenska razlika između dodira i otpuštanja dodira na ekranu
- **speed_pixels_per_ms**: Brzina prelaska u pikselima po milisekundi
- **swipe_angle**: Kut između koordinata početka i kraja akcije

Slike 3.4(a) i 3.4(b) predstavljaju izgled ekrana tijekom modaliteta premotavanja i potezanja, dok se primjer prikupljenih podataka može vidjeti na slici 3.5



(3.4(a)) Izgled ekrana tijekom modaliteta premotavanja



(3.4(b)) Izgled ekrana tijekom modaliteta potezanja

Slika 3.4 Prikaz različitih modaliteta: (a) premotavanje, (b) potezanje

E1Raw X	E1Raw Y	E2Raw X	E2Raw Y	E1Event Time	E2Event Time	Distance	Time_diff	Speed_pixels_	Swipe_angle
523	1039	492.49	1037	11424930	11424980	30.57	50	0.61	-176.25
599	885	559.59	908	11425182	11425230	45.63	48	0.95	149.73
599	885	513.78	934	11425182	11425247	98.31	65	1.51	150.1
599	885	481	954	11425182	11425259	136.69	77	1.78	149.68
603	847	570.88	860	11425418	11425463	34.65	45	0.77	157.97
603	847	509.68	887	11425418	11425480	101.53	62	1.64	156.8
603	847	450.13	920	11425418	11425497	169.4	79	2.14	154.47
603	847	425.5	936	11425418	11425505	198.56	87	2.28	153.37
601	833	572.73	845	11425664	11425696	30.71	32	0.96	157
601	833	519.88	872	11425664	11425713	90.01	49	1.84	154.32
601	833	461.69	907	11425664	11425730	157.74	66	2.39	152.02
579	828	540.38	850	11425900	11425930	44.44	30	1.48	150.33
579	828	481.19	886	11425900	11425946	113.72	46	2.47	149.33
579	828	444.13	915	11425900	11425963	160.49	63	2.55	147.18
579	828	441	918	11425900	11425965	164.75	65	2.53	146.89
526	851	503.15	867	11426112	11426146	27.9	34	0.82	145
526	851	474.84	890	11426112	11426163	64.33	51	1.26	142.68
526	851	438.27	925	11426112	11426179	114.78	67	1.71	139.85
526	851	390.22	978	11426112	11426196	185.91	84	2.21	136.91
526	851	351.35	1024	11426112	11426213	245.83	101	2.43	135.27

Slika 3.5 Primjer prikupljenih podataka tijekom modaliteta premotavanja i potezanja

3.1 Vizualizacija dobivenih podataka

3.1.1 Žiroskop i akcelerometar

Za vizualizaciju podataka senzora akcelerometra i žiroskopa, izrađena je funkcija `process_and_plot`, koja započinje pokušajem učitavanja CSV datoteke koristeći `pandas` biblioteku.

Sljedeći korak je provjera je li stupac `sensorType` tipa string. Ako nije, ispisuje se poruka i funkcija se zaustavlja. Ova provjera osigurava da se dalje obrađuju samo datoteke s ispravnim formatom podataka.

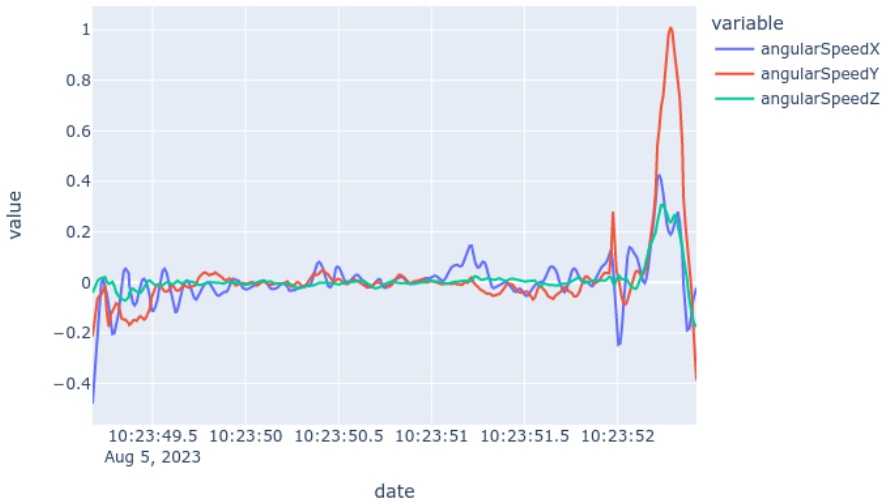
Datum se zatim ponovno formatira u specifičan oblik, osiguravajući dosljednost u interpretaciji vremenskih podataka unutar CSV datoteke.

Stupac `sensorType` se mapira u novi stupac `mappedSensorType` koristeći funkciju `map_sensor_type`. Ovo mapiranje omogućuje kategorizaciju podataka prema tipu senzora, što je ključno za daljnju analizu.

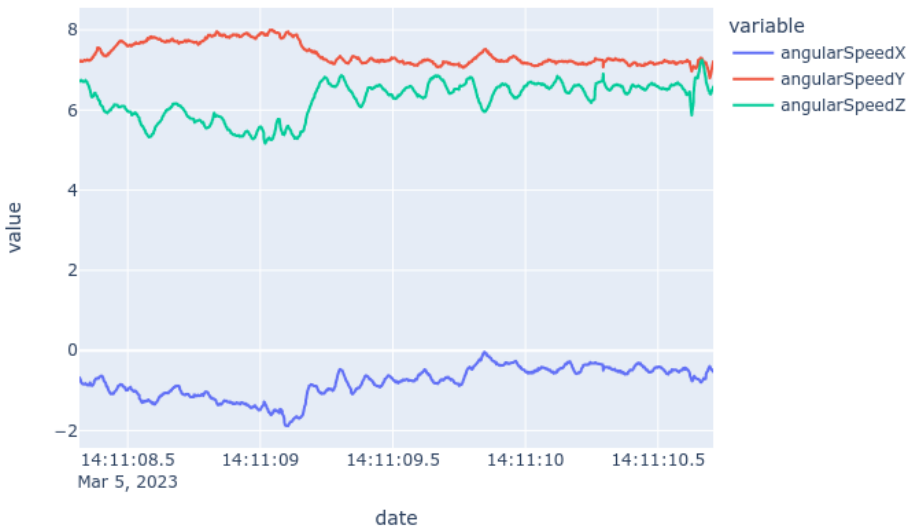
Funkcija zatim iterira kroz tipove senzora, konkretno `gyroscope` i `accelerometer`. Za svaki tip senzora, podaci se filtriraju i, ako postoje, kreira se linijski graf koristeći `plotly.express` biblioteku. Graf prikazuje promjene brzine (kutne brzine za žiroskop) kroz vrijeme. Ovi grafovi omogućuju vizualizaciju dinamike podataka prikupljenih od senzora, što može biti korisno za daljnju analizu i interpretaciju.

Ako dođe do bilo kakve pogreške tijekom izvršavanja, ispisuje se poruka o grešci. Ovo osigurava robusnost funkcije, omogućujući korisniku da identificira i dijagnosticira probleme s obradom podataka.

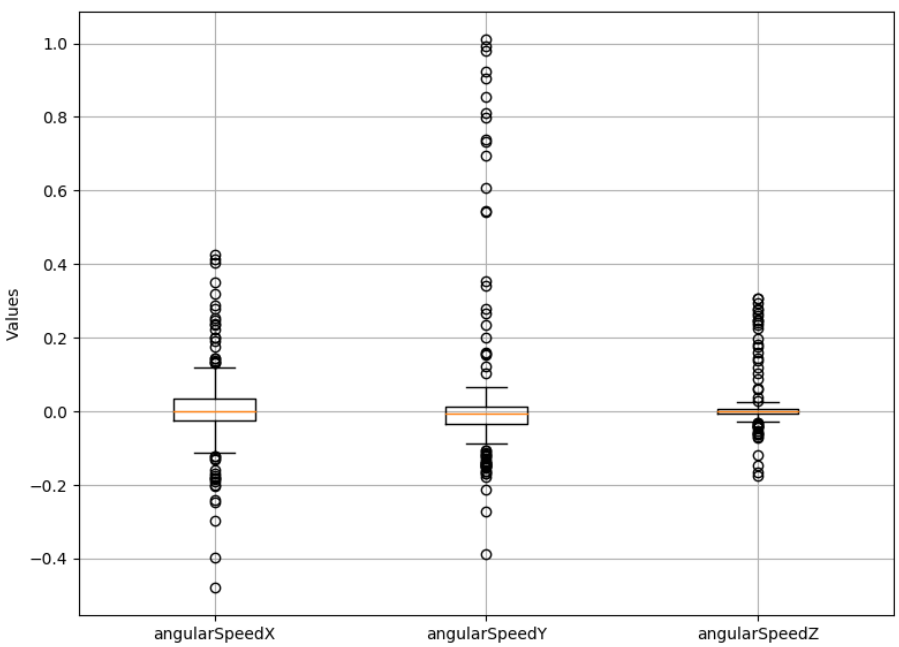
Funkcija `process_and_plot` je osmišljena za obradu i vizualizaciju podataka prikupljenih od senzora. Provjerava tip podataka, formatira datume, mapira tipove senzora i stvara odgovarajuće grafove koji se spremaju kao slike. U slučaju bilo kakvih pogrešaka tijekom obrade, funkcija osigurava ispisivanje korisnih poruka za dijagnostiku problema. Ovaj pristup omogućuje učinkovitu i pouzdanu analizu podataka prikupljenih od različitih tipova senzora.



Slika 3.6 Primjer grafa sa vrijednostima žiroskopa



Slika 3.7 Primjer grafa sa vrijednostima akcelerometra



Slika 3.8 Primjer boxplot grafa sa vrijednostima senzora

3.1.2 Unos PIN-a

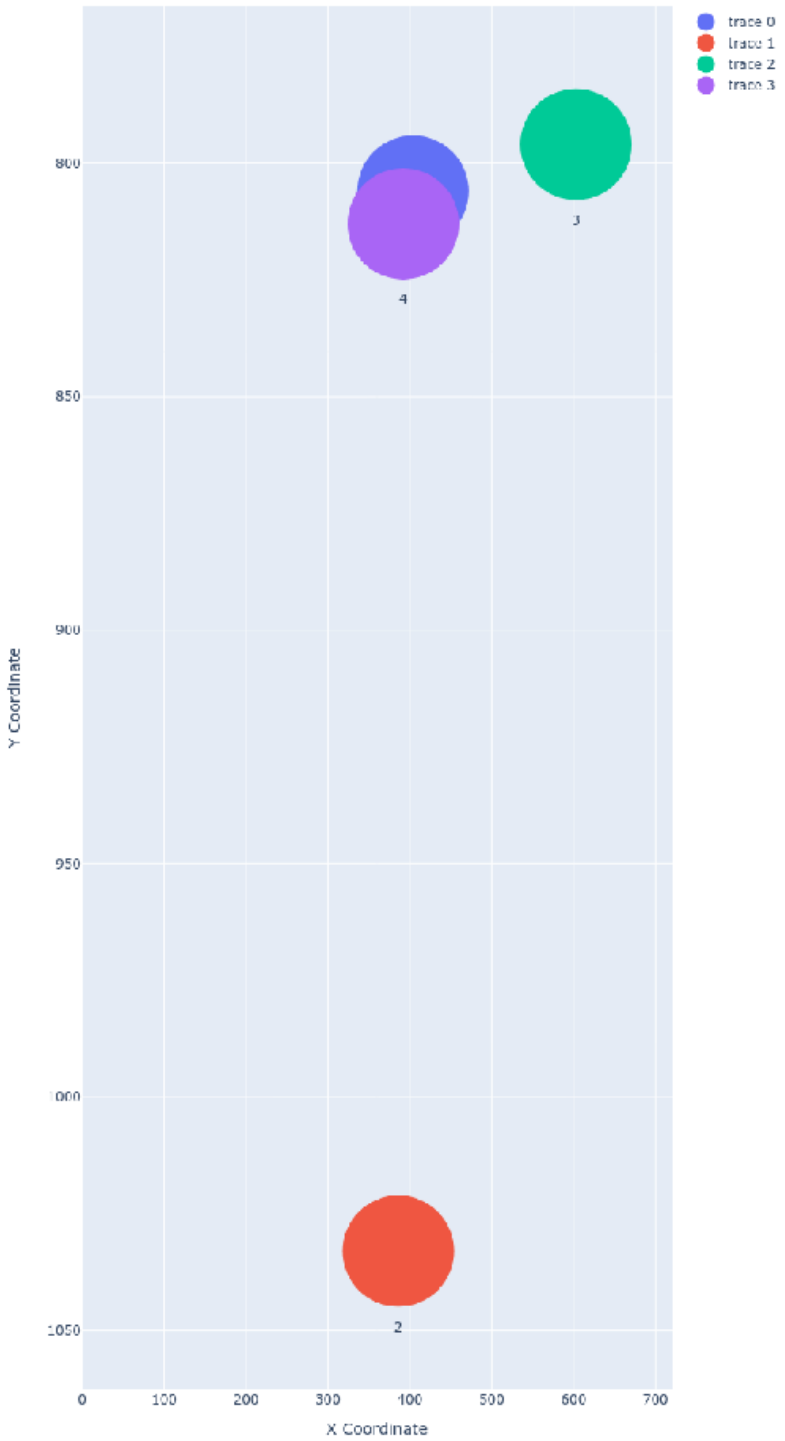
Funkcija `process_and_save_plot` služi za obradu podataka iz dviju CSV datoteka i stvaranje grafičkog prikaza interakcija s ekranom mobilnog uređaja. Prvo, funkcija pokušava učitati podatke iz datoteka koje su specificirane putanjama `pin_click_file` i `device_model_file` koristeći biblioteku `pandas` za manipulaciju podacima.

Nakon što su podaci uspješno učitani, funkcija ekstrahira informacije o rezoluciji zaslona mobilnog uređaja iz datoteke `device_model_file`. Ove informacije koristi se za postavljanje granica grafičkog prikaza.

Zatim, funkcija koristi biblioteku `plotly.graph_objects` za stvaranje scatter plot-a. Svaki redak iz datoteke `pin_click_file` predstavlja jedan klik na zaslonu, a svaka točka na grafu predstavlja jedan klik. Svaka točka na grafu je označena brojem koji odgovara redoslijedu klikova.

Grafički prikaz prilagođava se tako da naslov grafu sadrži naziv datoteke `pin_click_file`, dok su oznake na osima x i y postavljene prema širini i visini zaslona.

Konačno, graf se sprema kao PNG datoteka s istim imenom kao i ulazna datoteka `pin_click_file`, ali s ekstenzijom .png. Ovo omogućuje jednostavno povezivanje grafova s izvornim podacima.



Slika 3.9 Primjer grafa sa označenim korisnikovim dodirima

3.1.3 Modaliteti premotavanja i potezanja

Kako bismo mogli vizualizirati podatke prikupljene tijekom akcija premotavanja i potezanja, implementirana je funkcija `plot_swipes`, koja započinje pokušajem učitavanja CSV datoteke s putanjom specificiranom argumentom `file_path` koristeći `pandas` biblioteku. Nakon toga, podaci se pojednostavljuju korištenjem funkcije `simplify_swipes`, koja vjerojatno filtrira i strukturira podatke za daljnju analizu.

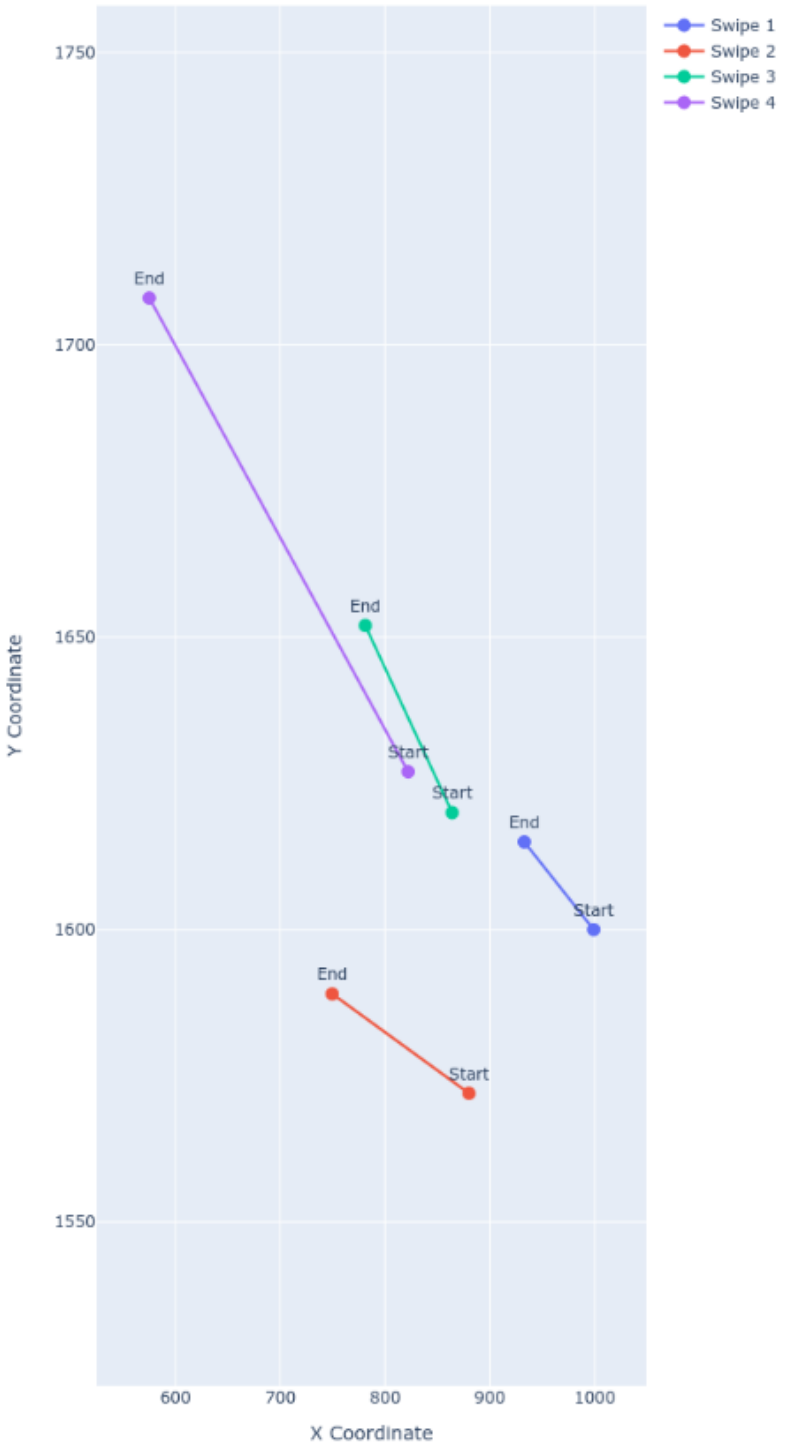
Provjerava se prisutnost ključnih stupaca (`e1RawX`, `e1RawY`, `e2RawX`, `e2RawY`) u pojednostavljenim podacima. Ako su svi potrebni stupci prisutni, kreira se novi graf koristeći `plotly.graph_objects` biblioteku.

Za svako prevlačenje u podacima, na graf se dodaje linija koja predstavlja putanju prevlačenja. Točke koje označavaju početak i kraj prevlačenja su označene kao "Start" i "End". Boje linija i točaka su odabrane iz skupa boja biblioteke `plotly` kako bi se razlikovale različite putanje.

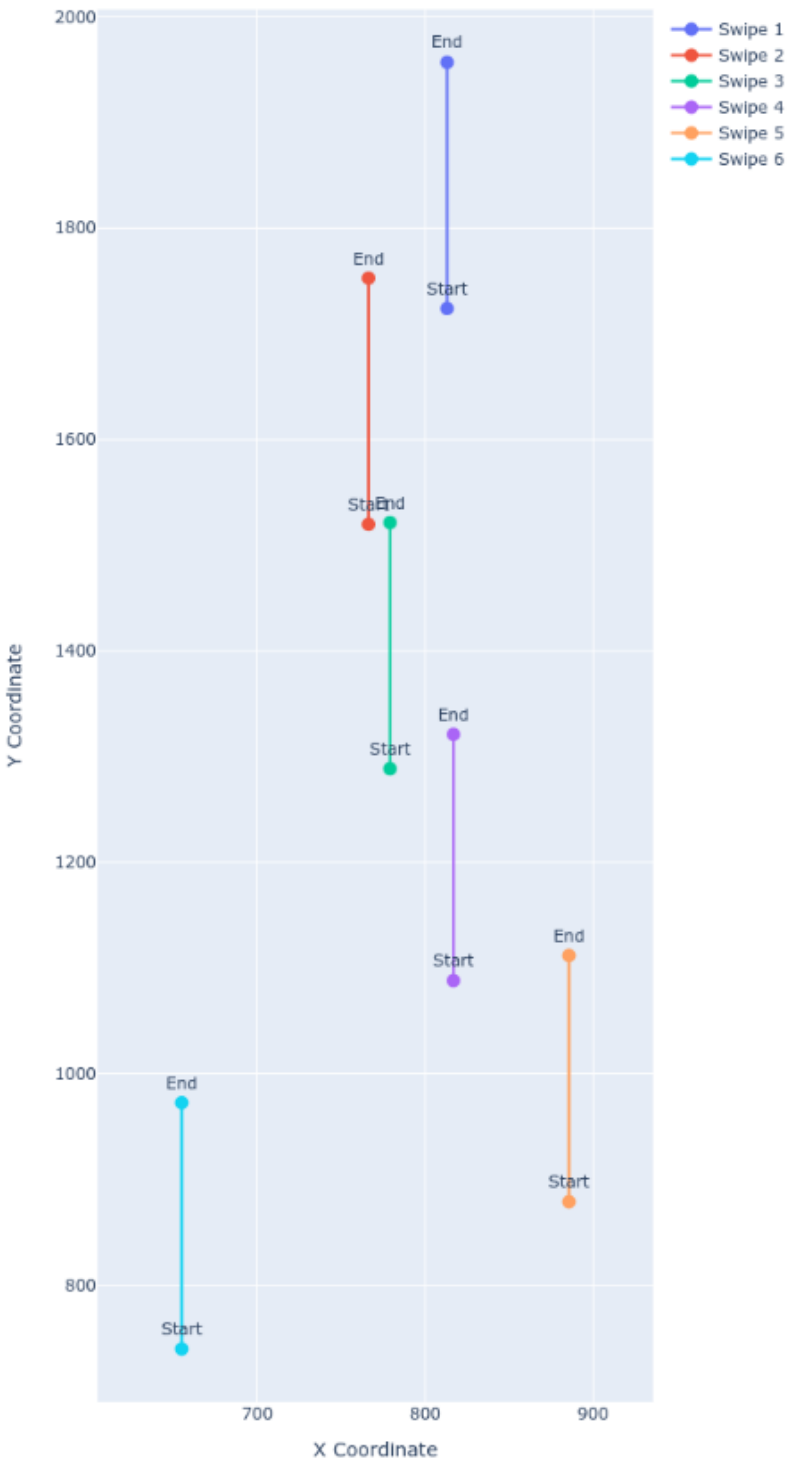
Granice osi x i y su postavljene na temelju minimalnih i maksimalnih vrijednosti koordinata u podacima, uz dodatni margina za bolji vizualni prikaz.

Izgled grafa se prilagođava tako da naslov uključuje naziv datoteke `file_path`, a osi x i y su označene kao "X Coordinate" i "Y Coordinate". Također, postavljaju se veličina grafa prema rezoluciji zaslona specificiranoj argumentom `screen_resolution`. Graf se sprema kao PNG datoteka s imenom izvedenim iz ulazne datoteke `file_path`, ali s dodatkom _scroll_plot.png. Ovo omogućuje jednostavno prepoznavanje i povezivanje grafova s izvornim podacima.

Ako dođe do bilo kakve pogreške tijekom izvršavanja, ispisuje se poruka o grešci. Ovo osigurava robusnost funkcije, omogućujući korisniku da identificira i dijagnosticira probleme s obradom podataka.



Slika 3.10 Primjer grafa sa označenim korisnikovim dodirima



Slika 3.11 Primjer grafa sa označenim korisnikovim dodirima tijekom akcije premotavanja

3.2 Korištenje podataka

Podaci koji će se koristiti za treniranje modela VGG16 i ResNet podijeljeni su omjerom 80:10:10, gdje 80% podataka će se koristiti za trening modela, 10% za validaciju treninga, dok će se ostalih 10% koristiti kao "test" skupina podataka. K-struka unakrsna validacija (eng. k-fold cross validation) nije korištena u ovome radu. Vrijedno je spomenuti kako su neki podaci eliminirani zbog loše kvalitete, na primjer zbog prekratkog mjerenja dodira ekrana mobilnog uređaja.

Poglavlje 4

Priprema podataka za treniranje modela dubokog učenja

Kao sljedeći korak ovog projekta, potrebno je pripremiti podatke koji će se koristiti za treniranje modela umjetne inteligencije. Uz to, koristit će se statistički test ANOVA, koji je opisan na početku metodologije ovog rada. Koristit će se ANOVA Repeated Measurement test, s obzirom da isti korisnik generira podatke kroz 3 dana korištenja mobilne aplikacije. [36]

Ovaj dio projekta, sastoji se od 6 koraka te ćemo sve korake pobliže opisati i objasniti:

1. Odvajanje podataka po danima
2. Sortiranje podataka unutar dana
3. Brisanje redaka koji ne sadrže podatke
4. Stvaranje CSV datoteke koja će se koristiti za ANOVA Repeated Measurement statistički test
5. Izvedba ANOVA Repeated Measurement statisičkog testa te analiza dobivenih podataka
6. Izrada spektrogram slika

4.1 Postupak procesiranja podataka

U sljedećoj tablici je prezentirana statistika o broju unosa te prosjeku i standardnoj devijaciji varijabli koje se koriste u ovom radu.

Tablica 4.1 Statistika dobivenih podataka

	Broj unosa	Prosjek	Standardna devijacija
angularSpeedX	9577359	1.395397e+00	3.247621e+00
angularSpeedY	9577359	2.700615e+13	1.650090e+14
angularSpeedZ	9577359	1.818917e+00	4.964015e+00

4.1.1 Učitavanje CSV datoteke

Na početku se koristi `pandas` knjižnica za učitavanje CSV datoteke. CSV datoteka se učitava u `DataFrame` objekt. U slučaju da datoteka nije pronađena ili da se ne može učitati, generira se odgovarajuća greška. Kôd koristi `pd.read_csv(file_path)` funkciju za učitavanje datoteke te ako se pojavi greška, skripta odmah prestaje sa radom.

Ako neki od tih stupaca nedostaje, kôd generira iznimku i prijavljuje koji stupci nedostaju.

4.1.2 Identifikacija jedinstvenih korisnika

Nakon provjere prisutnosti potrebnih stupaca, kôd identificira sve jedinstvene korisnike u stupcu `username` koristeći funkciju `unique()`.

4.1.3 Prerađivanje podataka

Za svakog korisnika, kôd kopira redove koji sadrže podatke samo za tog korisnika. Kôd dodaje nove stupce `angularSpeedX_dayX`, `angularSpeedY_dayX`, i `angularSpeedZ_dayX`, gdje `X` predstavlja dan. Novi stupci sadrže iste vrijednosti kao originalni stupci `angularSpeedX`, `angularSpeedY` i `angularSpeedZ`, ali su nazvani prema danu.

Nakon pripreme novih podataka za svakog korisnika, sve tablice se kombiniraju u CSV datoteku. U konačnoj tablici zadržavaju se samo stupci s podacima o kutnim brzinama, dok se svi drugi stupci uklanjaju.

Nakon filtriranja stupaca, rezultati se spremaju u novu CSV datoteku, koristeći `to_csv()` funkciju za spremanje rezultata

4.1.4 Rukovanje iznimkama

Tijekom cijelog procesa obrade podataka, kôd je zaštićen blokom `try-except`, koji omogućava hvatanje i prijavljivanje bilo kakvih grešaka koje se mogu pojaviti.

- Ako se pojavi greška tijekom učitavanja datoteke, provjere stupaca, obrade podataka ili spremanja rezultata, kôd prikazuje poruku o grešci i prekida izvršavanje.

```python
def process_angular_speed_data(file_path, output_file_path):
    try:
        # Load the CSV file
        df = pd.read_csv(file_path)

        # Check for required columns
        required_columns = ['username', 'angularSpeedX', 'angularSpeedY', 'angularSpeedZ']
        missing_columns = [col for col in required_columns if col not in df.columns]
        if missing_columns:
            raise ValueError(f"Missing columns in the file {file_path}: {', '.join(missing_columns)}")

        # Identify unique users and create a mapping to user numbers
        unique_users = df['username'].unique()
        user_mapping = {user: idx + 1 for idx, user in enumerate(unique_users)}

        # Prepare a list of new dataframes to be concatenated
        new_dfs = []

        for user in unique_users:
            user_df = df[df['username'] == user].copy()
            user_day = user_mapping[user]
            for axis in ['X', 'Y', 'Z']:
                user_df[f'angularSpeed{axis}_day{user_day}'] = user_df[f'angularSpeed{axis}']
            new_dfs.append(user_df)

        # Concatenate all new dataframes
        final_df = pd.concat(new_dfs, axis=0)

        # Drop all columns except the new angular speed columns and then the first three columns
        cols_to_keep = [col for col in final_df.columns if col.startswith('angularSpeed')]
        final_df = final_df[cols_to_keep]
        final_df = final_df.iloc[:, 3:]

        # Save the modified data to a new CSV file
        final_df.to_csv(output_file_path, index=False)
    except Exception as e:
        print(f"Error processing file {file_path}: {e}")
```

Slika 4.1 Korišteni kod za prvi korak procesiranja podataka

4.2 Brisanje redaka koji ne sadrže podatke

Kôd se sastoji od nekoliko ključnih koraka, od učitavanja podataka do spremanja očišćenih podataka. Svaki korak je detaljno opisan u nastavku.

Prvi korak u procesu čišćenja podataka je učitavanje skupa podataka iz CSV datoteke. Kôd koristi `pandas` knjižnicu za ovu operaciju. Datoteka se učitava pomoću funkcije `pd.read_csv(file_path)`.

Nakon učitavanja podataka, uklanjaju se redovi koji imaju prazne vrijednosti u stupcu `Measurement` koristeći `dropna(subset=['Measurement'])` za uklanjanje redova s praznim vrijednostima u specifičnom stupcu. Nakon uklanjanja, dobiva se nova tablica `data_cleaned` koja sadrži samo redove s ispunjenim vrijednostima u stupcu `Measurement`.

Kada su redovi s praznim vrijednostima uklonjeni, očišćeni podaci se spremaju natrag u istu CSV datoteku. Kod koji se koristi za spremanje je isti kao i u prethodnim koracima.

4.3 Stvaranje CSV datoteka za ANOVA statistički test

Kôd se sastoji od nekoliko ključnih koraka, od učitavanja podataka do spremanja pripremljenih podataka. Svaki korak je detaljno opisan u nastavku.

4.3.1 Učitavanje skupa podataka

Prvi korak u pripremi podataka je učitavanje skupa podataka iz CSV datoteke koristeći `pandas` knjižnicu. Datoteka se učitava pomoću funkcije `pd.read_csv(file_path)`, koja uzima putanju do datoteke kao argument. U slučaju da dođe do greške prilikom učitavanja datoteke, kôd generira iznimku koja je uhvaćena i ispisuje se poruka o grešci. Ova poruka služi za informiranje korisnika o problemima s datotekom ili njenom formatom, omogućujući brzo prepoznavanje i ispravljanje problema.

4.3.2 Provjera broja dana i osi

Nakon uspješnog učitavanja podataka, kôd provjerava prisutnost minimalnog broja dana u skupu podataka kako bi se osigurala dovoljno velika baza za daljnju analizu. Kôd koristi funkciju `unique()` za dobivanje jedinstvenih vrijednosti iz stupaca `Day` i `Axis`. Ova provjera je ključna jer, ako postoji manje od dva različita dana u podacima, daljnja analiza postaje besmislena. Ako se utvrdi da nema dovoljno različitih dana, kôd ispisuje poruku o nedostatku varijabilnosti i preskače obradu te datoteke, čime se štedi vrijeme i resursi na nepotrebnu obradu nepotpunih podataka.

4.3.3 Inicijalizacija DataFrame-a za obrađene podatke

Kôd inicijalizira prazan `DataFrame` objekt koji će se koristiti za spremanje uzorkovanih podataka. Ovaj `DataFrame`, nazvan `parsed_df`, služi kao privremena pohrana za sve podatke koji će biti obrađeni i filtrirani tijekom daljnjih koraka. Inicijalizacija praznog `DataFrame`-a omogućava fleksibilnost u dodavanju i manipulaciji podacima

tijekom procesa, osiguravajući da se samo relevantni i kvalitetni podaci zadrže za konačnu analizu.

4.3.4 Uzorkovanje podataka za svaki dan i os

Za svaki dan i os u skupu podataka, kôd provodi proces uzorkovanja podataka kako bi se osigurala reprezentativnost za daljnju analizu. Kôd iterira kroz sve jedinstvene kombinacije dana i osi prisutne u skupu podataka. Za svaku kombinaciju, kôd stvara naziv stupca u formatu `Day_Axis`, koji olakšava identifikaciju i rukovanje podacima. Kôd zatim koristi funkciju `query()` za filtriranje podataka prema specifičnom danu i osi. Ako postoji dovoljan broj uzoraka (definiran varijablom `samples`), kôd uzima nasumične uzorke pomoću funkcije `sample()`, čime se osigurava reprezentativan izbor podataka za analizu. Uzorkovani podaci se potom dodaju u `parsed_df` s odgovarajućim nazivom stupca. U slučaju da nema dovoljno uzoraka za određenu kombinaciju dana i osi, kôd ispisuje poruku o nedostatku podataka i preskače tu kombinaciju, osiguravajući da se ne dodaju nepotpuni ili nereprezentativni podaci u konačnu analizu.

4.3.5 Spremanje obrađenih podataka

Nakon što su svi podaci uzorkovani, provjerava se postoji li dovoljno podataka za spremanje. Ako `parsed_df` nije prazan, kôd koristi funkciju `to_csv(output_file_path, index=False)` za spremanje podataka u novu CSV datoteku. Ovaj korak osigurava da svi obrađeni i uzorkovani podaci budu spremni za daljnju analizu ili arhiviranje. U slučaju manjka podataka, ispisuje se poruka koja informira korisnika da nije pronašao odgovarajuće podatke za ANOVA analizu, čime se izbjegava stvaranje nepotrebnih datoteka s neadekvatnim podacima.

Day1_angular SpeedX	Day1_angular SpeedY	Day1_angular SpeedZ	Day2_angular SpeedX	Day2_angular SpeedY	Day2_angular SpeedZ	Day3_angular SpeedX	Day3_angular SpeedY	Day3_angular SpeedZ
-0.28	-1.45	-0.8	0.6	0.67	0.59	0.34	3.77	9.91
-0.43	-1.37	-0.86	0.4	0.39	0.5	-2.1	4.01	5.62
-0.38	-1.1	-0.44	-0.62	2.73	9.37	-0.32	-0.12	-0.2
-0.09	-1.1	-0.54	-0.67	-1.27	-0.87	-0.19	0.03	-0.09
-0.38	-1.09	-0.45	-0.58	2.71	9.33	-0.34	-0.13	-0.21
0.05	-1.09	-0.69	-0.29	-0.11	-0.29	0.2	-0.23	-0.11
-0.04	-1.07	-0.64	0.03	-0.02	-0.03	-0.01	3.28	9.41
-0.33	-1.01	-0.04	-0.07	0.14	-0.08	-0.1	-0.58	-0.44
-0.34	-0.98	-0.39	0.03	-0.15	-0.07	-0.26	0.01	-0.33
-0.3	-0.96	-0.65	-1.93	3.68	7.83	-0.77	3.14	9.57
-0.38	-0.86	-0.74	-0.29	2.9	8.82	-0.01	0.3	0.2
-0.33	-0.85	-0.67	0.05	0.1	-0.07	-1.42	4.11	8.8
0.31	-0.74	-0.05	0.87	1.79	9.48	0.13	2.95	8.71
-0.15	-0.7	-0.68	0.53	2.08	9.52	0.15	0.12	0.08
0.44	-0.69	-0.4	-0.19	-0.34	-0.04	0.16	-0.17	0.12
-0.4	-0.69	0.46	0.14	2.35	10.1	0.09	0.29	-0.2
0.42	-0.67	-0.4	-0.18	-0.35	-0.04	0.15	-0.14	0.12

Slika 4.2 Primjer generirane CSV datoteke koji će se koristiti za ANOVA Repeated Measurement statistički test

4.4 Izvedba ANOVA Repeated Measurement statističkog testa

Priprema podataka započinje identifikacijom stupaca koji sadrže relevantne podatke za specifičnu os. Nakon identifikacije, provjerava se dostupnost dovoljnog broja dana za analizu. Ako broj stupaca nije zadovoljavajući (manje od dva), analiza se ne provodi dalje. U suprotnom, otvara se tekstualna datoteka za spremanje rezultata analize. Datoteka se inicijalizira s naslovom koji jasno identificira os koja se analizira, osiguravajući preglednost i organizaciju rezultata.

4.4.1 Izvođenje ANOVA analize s uzorkovanjem

Za svaku iteraciju uzorkovanja podataka provodi se analiza varijance s ponovljenim mjerenjima (ANOVA RM). Test ANOVA se iterira 10 puta kako bi se provjerila hipoteza da ne postoji statistička razlika između korisnikovih podataka. Svaka iteracija uključuje nasumično uzimanje uzorka od 100 podataka iz velikog broja podataka jednog korisnika.

Proces počinje iteriranjem kroz definirani broj puta, označen s `num_iterations`. U svakoj iteraciji, sjeme slučajnog broja resetira se pomoću funkcije `np.random.seed(None)` kako bi se osiguralo generiranje novog, nasumičnog stanja za uzorkovanje.

Nakon generiranja novog stanja, uzorkuju se podaci iz stupaca navedenih u `axis_columns`. Uzorkovanje se provodi pomoću funkcije `sample()`, uzimajući nasumične uzorke temeljem trenutnog stanja generiranja slučajnih brojeva. Uzorci se preimenuju u format `Day1, Day2, ...` kako bi se osigurala konzistentnost i preglednost podataka za daljnju analizu.

Podaci se dalje transformiraju u dugi format korištenjem funkcije `pd.melt()`, što organizira podatke tako da svaki redak predstavlja jedno mjerenje povezano s odgovarajućim danom. Ovaj format je prikladan za ANOVA RM analizu, koja procjenjuje varijaciju unutar i između skupina podataka prema različitim danima.

Na kraju, za provođenje ANOVA RM analize koristi se funkcija `AnovaRM()`. Ova funkcija postavlja `Measurement` kao zavisnu varijablu, označava subjekte pomoću

`index`, a unutargrupnu varijablu kao `Day`. Analiza omogućava detaljnu procjenu varijabilnosti i statističkih razlika među grupama podataka.

Rezultati ANOVA statističkog testiranja uključuju nekoliko ključnih varijabli koje pružaju uvid u statističku značajnost razlika između grupa. F-vrijednost (F) je omjer varijabilnosti između grupa i varijabilnosti unutar grupa, te se koristi za procjenu postojanja statistički značajnih razlika među srednjim vrijednostima grupa. Stupnjevi slobode za numeratore (Num DF) i denumeratore (Den DF) definiraju broj neovisnih jedinica podataka koje se koriste u testiranju. Numerator DF predstavlja broj grupa minus jedan, dok denumerator DF predstavlja ukupan broj mjerenja minus broj grupa. P-vrijednost ($Pr > F$) je vjerojatnost da se dobije rezultat testa koji je ekstremniji od onog koji je stvarno promatran, a koristi se za donošenje zaključaka o statističkoj značajnosti razlika između grupa. Niska p-vrijednost ispod prihvaćenog praga (obično 0.05) ukazuje na statistički značajne razlike između grupa, dok visoka p-vrijednost implicira nedostatak statističke značajnosti.

Tablica 4.2 Rezultati ANOVA RM statističkog testa za podatke žiroskopa gdje rezultati nemaju statistički značajnu razliku

Iteracija	F Value	Num DF	Den DF	Pr > F
1	26298.5125	2.0000	198.0000	0.0000
2	36160.9916	2.0000	198.0000	0.0000
3	33113.3870	2.0000	198.0000	0.0000

Tablica 4.3 Rezultati ANOVA RM statističkog testa za podatke akcelerometra gdje rezultati imaju statistički značajnu razliku

Iteracija	F Value	Num DF	Den DF	Pr > F
1	0.3244	2.0000	198.0000	0.7233
2	0.1716	2.0000	198.0000	0.8425
3	0.1089	2.0000	198.0000	0.8969

Uvidom u ove rezultate, može se zaključiti kako dobiveni podaci sa žiroskopa nemaju statistički signifikantnu razliku, dok podaci s akcelerometra imaju statis-

tički signifikantnu razliku. Iz tog razloga, nadalje će se koristiti podaci žiroskopa za predviđanje treniranje modela umjetne inteligencije.

4.5 Izrada spektrogram slika

Prvi korak u izradi spektrograma uključuje izračun vremenskih razlika između uzastopnih mjerenja unutar vremenske serije. Za svaki zapis koji bilježi korisnikove pomake u vremenskom razmaku t = 1 računa se vremenska razlika u odnosu na prijašnji korisnikov zapis unutar iste datoteke. Nakon toga, svi `NaN` podaci, koji se mogu pojaviti zbog izračuna razlike prvog elementa, uklanjaju se kako bi se osigurala valjanost podataka i spriječile pogreške u daljnjoj analizi.

Nakon izračuna vremenskih razlika između uzastopnih mjerenja, potrebno je odrediti frekvenciju uzorkovanja, koja je ključna za pravilnu analizu vremenskih serija. Frekvencija uzorkovanja određuje koliko često se podaci uzorkuju unutar zadanog vremenskog intervala. Da bi se izračunala frekvencija uzorkovanja, koristi se medijan vremenskih razlika, jer medijan pruža stabilniju i reprezentativniju vrijednost u prisutnosti iznimno visokih ili niskih vrijednosti. Ovakav pristup omogućava preciznije određivanje frekvencije uzorkovanja, eliminirajući utjecaj mogućih odstupanja ili anomalija u podacima.

Frekvencija uzorkovanja izračunava se kao recipročna vrijednost medijana vremenskih razlika. Ovaj proračun daje broj uzoraka po sekundi, što je ključno za točnu analizu i interpretaciju vremenskih serija. Precizno određena frekvencija uzorkovanja osigurava da su podaci dovoljno detaljni za analizu, ali i da se izbjegne nepotrebno visoka frekvencija koja bi mogla rezultirati viškom podataka i nepotrebno povećati složenost analize.

Sljedeći korak u postupku izrade spektrograma je primjena Short-Time Fourier Transform (STFT), koja je ključna za analizu promjena frekvencijskog sadržaja signala kroz vrijeme. STFT se koristi za pretvorbu signala iz vremenske domene u frekvencijsku domenu u malim vremenskim intervalima, stvarajući niz vremenski preklopljenih Fourierovih transformacija. Ova metoda omogućava da se frekvencijski sadržaj signala prati tijekom vremena, što je bitno za analizu vremenskih promjena

u signalu. STFT generira kompleksne spektrogramske koeficijente koji se mogu koristiti za vizualizaciju promjena frekvencijskog sadržaja signala kroz vrijeme.

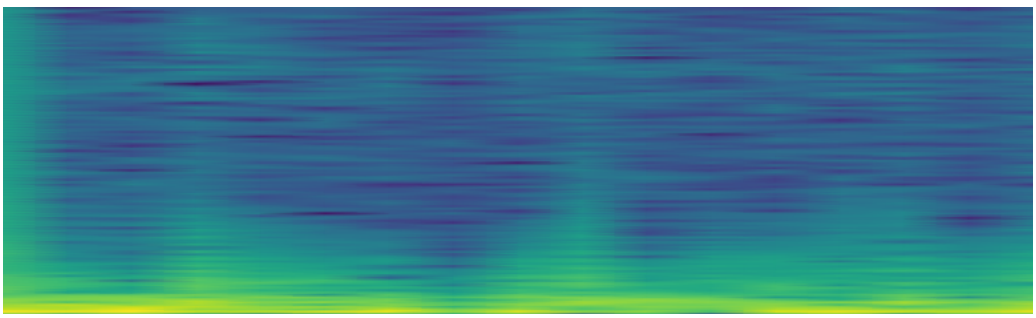
Na kraju, spektrogram, kao rezultat primjene STFT, pruža vizualni prikaz frekvencija u odnosu na vrijeme, omogućujući identifikaciju kako i kada se određeni frekvencijski sadržaji pojavljuju i nestaju unutar promatranog vremenskog intervala. Ovaj proces omogućava detaljnu analizu i interpretaciju signala, pružajući vrijedne informacije za daljnju analizu i interpretaciju podataka. Primjer spektrograma možemo vidjeti na slici 4.3.

4.5.1 Ekstrakcija podataka za specifičnu os

Kako bi se analizirali podaci za specifičnu os, prvo se iz skupa podataka izdvajaju informacije relevantne za tu os. Podaci se ekstrahiraju iz stupca koji je nazvan prema vrijednosti `axis_name`. Ovaj korak je važan jer osigurava da se analiziraju samo relevantni podaci za odabranu os. Nakon ekstrakcije, podaci za specifičnu os se spremaju u varijablu `axis_data`, koja se koristi kao ulaz za nadaljnje analize. Na ovaj način, fokus analize usmjeren je na specifičnu os, što omogućuje detaljnu analizu ponašanja podataka u odnosu na tu os.

4.5.2 Izračun spektrograma

Za izračun spektrograma podataka koristi se funkcija `spectrogram()` iz knjižnice `scipy.signal`. Spektrogram je vizualni prikaz spektralne gustoće snage signala u odnosu na frekvenciju i vrijeme. Kao ulazni podaci koristi se `axis_data`, a frekvencija uzorkovanja definirana je varijablom `fs`. Funkcija `spectrogram()` računa spektrogram tako da analizira signal prema njegovim frekvencijskim komponentama u određenim vremenskim intervalima. Parametar `scaling='spectrum'` koristi se za skaliranje spektrograma kako bi prikazao spektralnu gustoću snage, omogućujući vizualizaciju spektra snage u odnosu na frekvenciju. Funkcija vraća tri osnovne vrijednosti: frekvencije (`f`), vremena (`t`) i spektralnu gustoću snage (`Sxx`). Ove vrijednosti omogućavaju detaljnu analizu frekvencijskog sadržaja signala tijekom vremena.



Slika 4.3 Primjer generiranog spektrograma

Poglavlje 5

Treniranje modela umjetne inteligencije

U posljednjem djelu ovog rada, prezentirat će se kroz kod koji se koristi za stvaranje CSV datoteke gdje se nalaze informacije koje će se proslijediti modelu umjetne inteligencije te će biti objašnjeno objasniti postupak treniranja VGG16 i ResNet modela.

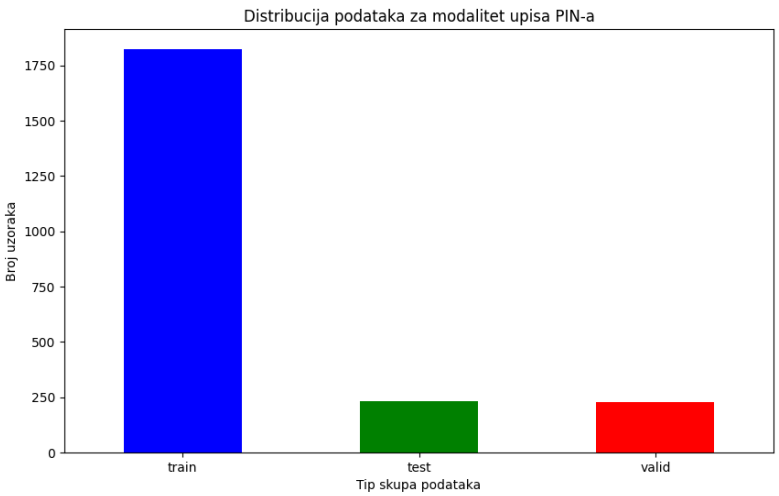
5.1 Generiranje CSV datoteke

Kao što je ranije spomenuto, podaci su podijeljeni u omjeru 80:10:10, gdje je 80% podataka korišteno za trening modela, 10% za validaciju treninga i 10% za testnu skupinu podataka te je ova distribucija spremljena u jednu zajedničku datoteku.

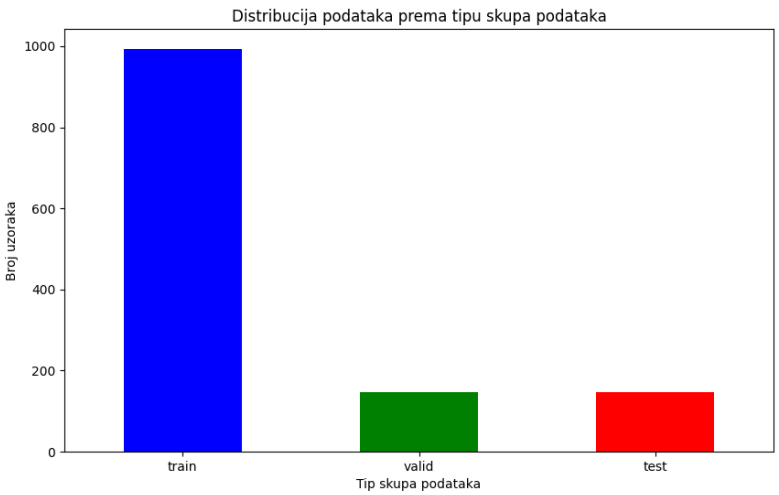
CSV datoteka ima unaprijed definirana zaglavlja koja opisuju informacije koje će biti pohranjene za svaku sliku. Zaglavlja su:

- `image_path`: Putanja do slike.
- `dataset_type`: Tip podatkovnog skupa (trening, validacija, test).
- `label_type`: Tip oznake (os X, Y, Z).
- `user_id`: ID korisnika, izvučen iz naziva datoteke.

Na fotografijama 5.1 i 5.2 se može vidjeti količina podataka zastupljena u svakoj skupini podataka.



Slika 5.1 Distribucija podataka za modalitet pisanja teksta



Slika 5.2 Distribucija podataka za modalitet upisa PIN-a

5.1.1 Priprema učitavača podataka (DataLoader)

Sljedeći korak je priprema učitavača podataka (*eng. DataLoader*). Učitavač podataka je klasa koja učitava podatke iz podatkovnog skupa u veličinu serija i priprema ih za model. Specifičnosti postavki su:

- `dataset` je prilagođeni dataset kreiran u prethodnom koraku.
- `batch_size` određuje veličinu batch-a, odnosno broj uzoraka koji će biti obrađeni odjednom. U ovom slučaju, batch veličina je postavljena na 16.
- `shuffle` osigurava da su podaci u training setu izmiješani pri svakom novom učenju, što poboljšava generalizaciju modela. Ova opcija je omogućena.

5.2 Definiranje i treniranje modela umjetne inteligencije

Koraci u procesu treniranja modela VGG16 i ResNet, koji su korišteni u ovom projektu, su isti. Postoje razlike u kodu, no iste će biti definirane u nastavku.

Koraci su sljedeći:

1. Inicijaliziranje modela i učitavanje podataka
2. Treniranje modela i spremanje modela u obliku .pth datoteke
3. Zapisivanje rezulata generiranih tijekom treniranja

5.2.1 Inicijalizacija modela i učitavanje podataka

Prije treniranja modela, definirani su optimizator, stopa učenja, veličina serije, broj epoha te funkcija gubitaka. Stopa učenja pronađena je iscrpnim pretraživanjem prostora, gdje je testirani raspon stope učenja bio [0.001, 1e-6]. Najbolji rezultati dobiveni su pri stopi učenja 10^{-5} te su u ovom radu prezentirati rezultati koji su dobiveni upravo iz tih najboljih modela Kao optimizator ovog modela, korišten je AdamW iz Python knjižnice Torch [37]. veličina serije je 16 i funkcija gubitaka je

"Cross Entropy Loss" [38]. Formula "Cross Entropy Loss" funkcije gubitaka je:

$$H(y, \hat{y}) = -\sum_{i} y_i \log(\hat{y}_i)$$

gdje y predstavlja stvarne oznake (obično u obliku vektora s jednim aktivnim bitom za svaku klasu), $\hat{y}$ predstavlja predviđene vjerojatnosti (izlaz funkcije softmax), a i indeksira kroz klase.

Vrijedi spomenuti kako je korišten i mehanizam ranijeg zaustavljanja (eng. Early Stop Mechanism) koje zaustavlja treniranje modela ranije ako se performanse modela nisu poboljšale tijekom 10 uzastopnih epoha.

5.2.2 Definicija dataset-a za trening modela VGG16

Prvi korak u funkciji je definicija dataset-a. Ovdje se koristi prilagođena klasa za dataset koja uzima CSV datoteku, specifičnu os i transformacije kao ulazne parametre. Parametri su sljedeći:

- `csv_file` postavlja putanju do CSV datoteke koja sadrži informacije o slikama.
- `axis` omogućuje datasetu da filtrira podatke prema zadanoj osi.
- `transform` predstavlja niz transformacija koje će biti primijenjene na slike prije nego što se proslijede modelu.

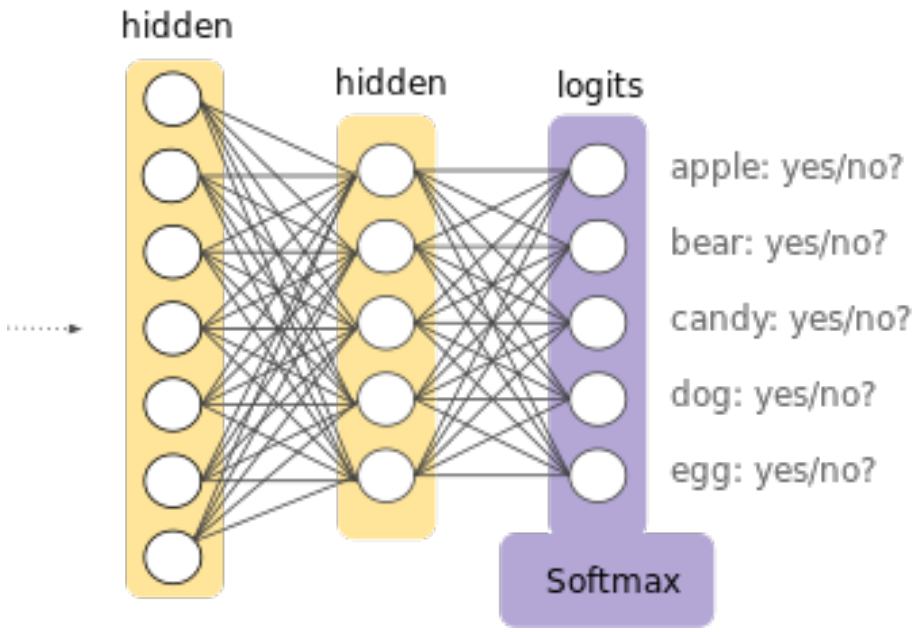
Kroz ove korake dataset se inicijalizira s podacima specifičnim za zadanu os.

5.2.3 Definiranje modela

Model koji se koristi je unaprijed definiran model *VGG16* i *ResNet* iz PyTorch biblioteke. U funkciji se koristi unaprijed predtrenirane težine za navedene modele, koje su trenirane na ImageNet-u, što omogućava bolje početne performanse jer koristi već naučene značajke s velikog skupa podataka [39]. Vrijedi spomenuti činjenicu kako je su u ovom radu rađene dvije metrike VGG16 i ResNet modela - jedna inačica spomenutih modela pokušava predvidjeti korisnika, dok druga pokušava predvidjeti ako se korisnik nalazi u skupini od 3 najvjerojatnije osobe. Na ovaj način, pokušat će se poboljšati performanse modela.

5.2.4 Prilagodba modela

Model se prilagođava tako da zadnji sloj klasifikatora odgovara broju klasa u dataset-u, koji je u ovom projektu broj korisnika, odnosno 39. Zadnji sloj modela koji se koriste u ovom radu sadrže SoftMax sloj, koji za svaku klasu, daje postotak vjerojatnosti [40].



Slika 5.3 SoftMax sloj

5.2.5 Treniranje modela dubokog učenja

Treniranje VGG16 i ResNet modela, koji su duboke konvolucijske neuronske mreže, započinje inicijalizacijom modela s prethodno treniranim težinama kako bi se iskoristilo postojeće znanje o prepoznavanju slikovnih značajki. U ovom postupku, optimizator AdamW se koristi za ažuriranje težina modela, jer kombinira prednosti Adam optimizatora, kao što su adaptivne brzine učenja za svaki parametar, s L2 regularizacijom, što pomaže u prevenciji pretreniranja modela.

Prvi korak algoritma uključuje definiranje modela i njegovih parametara. Oba već spomenuta modela se inicijaliziraju i modificiraju prema potrebama specifičnog

zadatka, na primjer klasifikacije slika. Zatim se postavlja optimizator AdamW s odgovarajućom brzinom učenja. AdamW, za razliku od klasičnog Adam optimizatora, uključuje težinski raspad (weight decay) kao dodatni faktor regularizacije koji sprječava model da se previše prilagodi na trening podatke. [41]

Proces treniranja modela započinje inicijalizacijom modela, optimizatora, učitavača podataka za treniranje i validaciju te gubitka i funkcije za rano zaustavljanje. Model se trenira kroz unaprijed definirani broj epoha (u ovom projektu, maksimalni broj epoha je 100), gdje se unutar svake epohe izmjenjuju faze treniranja i validacije.

U fazi treniranja, model se postavlja u način rada za treniranje i iterira kroz učitavač podataka za treniranje. Podaci se prebacuju na odgovarajući uređaj, optimizator resetira gradijente, a model prolazi kroz naprijed-nazad prolaz. U naprijed prolazu, model generira predikcije, a gubitak se izračunava na temelju funkcije gubitka. Gradijenti se izračunavaju u natrag prolazu, a optimizator ažurira težine modela. Svi gubici i ispravne predikcije se prikupljaju kako bi se izračunali ukupni gubitak i točnost po epohi. Također, bilježe se i vrijednosti za odziv i F1 vrijednost.

U fazi validacije, model prelazi u način rada za evaluaciju, a proces je sličan onom u fazi treniranja, s tom razlikom što se ne ažuriraju težine modela. Na temelju rezultata validacije, pohranjuju se metrika gubitka, točnosti, odziva i F1 vrijednosti. Ako točnost modela na validacijskom skupu nadmašuje prethodno najbolju zabilježenu točnost, ažuriraju se najbolje težine modela. Funkcija za rano zaustavljanje prati gubitak na validacijskom skupu i, ukoliko se gubitak ne smanjuje kroz određeni broj epoha, zaustavlja treniranje kako bi se spriječilo pretreniranje modela.

Nakon završetka treniranja kroz sve epohe ili ranog zaustavljanja, najbolji postignuti rezultat modela se ispisuje, a model se vraća na najbolje pohranjene težine. Na kraju se vraća trenirani model zajedno s metrikama treniranja i validacije, uključujući gubitke, točnosti, odziva i F1 vrijednosti za svaku epohu. Ova metodologija osigurava temeljitu evaluaciju modela i omogućava njegovu optimizaciju za što bolji učinak na danom skupu podataka.

Poglavlje 6

Rezultati

U ovom poglavlju usporedit će se podaci generirani tijekom eksperimenta, koji će prikazati performanse dubokog modela učenja. Evaluacija će biti provedena u dva scenarija: u prvom scenariju mjeri se sposobnost modela na zadatku klasifikacije korisnika, dok se u drugom scenariju kriterij klasifikacije olakšava te se mjeri točnost klasificiranja korisnika prema top-k metrici u kojoj je k postavljen na 3. Na kraju, bit će prezentirani rezulati dvaju modela, jedan iz modaliteta upisa teksta te drugi iz modaliteta upisa PIN-a, kako bismo mogli zaključiti koji modalitet bolje predviđa korisnika.

6.1 Usporedba rezultata modaliteta upisa teksta

Metrika	**Trening -> Top 1**	**Validacija -> Top 1**	**Trening -> Top 3**	**Validacija -> Top 3**
Najbolja vrijednost funkcije gubitka	0.05	0.74	2.7	2.93
Najbolja preciznost	99.61%	75.00%	99.94%	100.00%
Najbolji odziv	100.00%	75.44%	99.19%	77.84%
Najbolji F1 score	100.00%	75.74%	99.75%	77.01%

Tablica 6.1 Metrike ResNet modela tijekom treninga i validacije X osi tijekom modaliteta upisa teksta

Metrika	**Trening -> Top 1**	**Validacija -> Top 1**	**Trening -> Top 3**	**Validacija -> Top 3**
Najbolja vrijednost funkcije gubitka	0.07	0.65	2.69	2.90
Najbolja preciznost	99.39%	77.19%	100.00%	100.00%
Najbolji odziv	99.23%	77.75%	100.00%	79.65%
Najbolji F1 score	99.34%	77.26%	100.00%	79.16%

Tablica 6.2 Metrike ResNet modela tijekom treninga i validacije Y osi tijekom modaliteta upisa teksta

Metrika	**Trening -> Top 1**	**Validacija -> Top 1**	**Trening -> Top 3**	**Validacija -> Top 3**
Najbolja vrijednost funkcije gubitka	0.04	0.64	2.74	2.93
Najbolja preciznost	99.67%	76.31%	99.89%	100.00%
Najbolji odziv	100.00%	76.33%	96.71%	78.99%
Najbolji F1 score	100.00%	76.90%	96.15%	78.18%

Tablica 6.3 Metrike ResNet modela tijekom treninga i validacije Z osi tijekom modaliteta upisa teksta

Metrika	**Trening -> Top 1**	**Validacija -> Top 1**	**Trening -> Top 3**	**Validacija -> Top 3**
Najbolja vrijednost funkcije gubitka	0.02	0.75	2.71	2.83
Najbolja preciznost	99.67%	81.14%	99.94%	100.00%
Najbolji odziv	100.00%	81.75%	97.65%	86.50%
Najbolji F1 score	100.00%	81.26%	97.11%	86.61%

Tablica 6.4 Metrike VGG16 modela tijekom treninga i validacije X osi tijekom modaliteta upisa teksta

Metrika	**Trening -> Top 1**	**Validacija -> Top 1**	**Trening -> Top 3**	**Validacija -> Top 3**
Najbolja vrijednost funkcije gubitka	0.06	0.58	2.79	2.91
Najbolja preciznost	98.24%	82.01%	99.39%	99.56%
Najbolji odziv	98.18%	82.58%	89.14%	77.94%
Najbolji F1 score	98.34%	82.11%	89.23%	77.10%

Tablica 6.5 Metrike VGG16 modela tijekom treninga i validacije Y osi tijekom modaliteta upisa teksta

Metrika	**Trening –> Top 1**	**Validacija –> Top 1**	**Trening –> Top 3**	**Validacija –> Top 3**
Najbolja vrijednost funkcije gubitka	0.05	0.52	2.71	2.86
Najbolja preciznost	98.51%	83.33%	99.56%	99.56%
Najbolji odziv	99.45%	83.14%	95.29%	79.86%
Najbolji F1 score	98.69%	83.15%	95.66%	79.15%

Tablica 6.6 Metrike VGG16 modela tijekom treninga i validacije Z osi tijekom modaliteta upisa teksta

6.2 Usporedba rezultata modaliteta upisa PIN-a

Metrika	**Trening -> Top 1**	**Validacija -> Top 1**	**Trening -> Top 3**	**Validacija -> Top 3**
Najbolja vrijednost funkcije gubitka	0.12	0.72	2.93	3.02
Najbolja preciznost	99.38%	72.00%	99.73%	100.00%
Najbolji odziv	100.00%	71.33%	99.41%	76.00%
Najbolji F1 score	99.68%	71.24%	99.70%	75.91%

Tablica 6.7 Metrike ResNet modela tijekom treninga i validacije X osi tijekom modaliteta upisa PIN-a

Metrika	**Trening -> Top 1**	**Validacija -> Top 1**	**Trening -> Top 3**	**Validacija -> Top 3**
Najbolja vrijednost funkcije gubitka	0.15	0.75	2.92	3.01
Najbolja preciznost	99.25%	71.56%	99.68%	100.00%
Najbolji odziv	99.26%	71.58%	98.25%	78.93%
Najbolji F1 score	99.14%	71.60%	98.22%	78.89%

Tablica 6.8 Metrike ResNet modela tijekom treninga i validacije Y osi tijekom modaliteta upisa PIN-a

Metrika	**Trening -> Top 1**	**Validacija -> Top 1**	**Trening -> Top 3**	**Validacija -> Top 3**
Najbolja vrijednost funkcije gubitka	0.14	0.71	2.85	3.10
Najbolja preciznost	99.33%	72.93%	99.61%	100.00%
Najbolji odziv	99.57%	72.54%	97.61%	79.12%
Najbolji F1 score	99.11%	72.60%	97.48%	79.30%

Tablica 6.9 Metrike ResNet modela tijekom treninga i validacije Z osi tijekom modaliteta upisa PIN-a

Metrika	**Trening -> Top 1**	**Validacija -> Top 1**	**Trening -> Top 3**	**Validacija -> Top 3**
Najbolja vrijednost funkcije gubitka	0.05	0.64	2.70	2.91
Najbolja preciznost	97.34%	80.23%	99.45%	99.57%
Najbolji odziv	99.21%	80.69%	93.69%	81.54%
Najbolji F1 score	97.98%	80.28%	94.14%	81.13%

Tablica 6.10 Metrike VGG16 modela tijekom treninga i validacije X osi tijekom modaliteta upisa PIN-a

Metrika	**Trening -> Top 1**	**Validacija -> Top 1**	**Trening -> Top 3**	**Validacija -> Top 3**
Najbolja vrijednost funkcije gubitka	0.08	0.71	2.73	3.12
Najbolja preciznost	98.12%	78.10%	98.14%	100.00%
Najbolji odziv	98.13%	78.03%	92.68%	83.14%
Najbolji F1 score	98.09%	78.08%	92.85%	83.17%

Tablica 6.11 Metrike VGG16 modela tijekom treninga i validacije Y osi tijekom modaliteta upisa PIN-a

Metrika	**Trening -> Top 1**	**Validacija -> Top 1**	**Trening -> Top 3**	**Validacija -> Top 3**
Najbolja vrijednost funkcije gubitka	0.04	0.62	2.77	3.05
Najbolja preciznost	98.25%	81.14%	99.46%	99.58%
Najbolji odziv	98.22%	81.11%	91.78%	83.24%
Najbolji F1 score	98.30%	81.20%	92.01%	83.09%

Tablica 6.12 Metrike VGG16 modela tijekom treninga i validacije Z osi tijekom modaliteta upisa PIN-a

6.3 Rezultati modela s test skupinom podataka

Model	**Upis teksta**	**Upis PIN-a**
VGG16 Top 1 X os	82.07%	72.78%
VGG16 Top 1 Y os	82.54%	75.34%
VGG16 Top 1 Z os	81.97%	76.87%
VGG16 Top 3 X os	99.95%	98.11%
VGG16 Top 3 Y os	99.41%	100%
VGG16 Top 3 Z os	100%	98.96%
ResNet Top 1 X os	77.57%	76.12%
ResNet Top 1 Y os	74.11%	69.96%
ResNet Top 1 Z os	76.87%	72.80%
ResNet Top 3 X os	100%	94.78%
ResNet Top 3 Y os	100%	97.57%
ResNet Top 3 Z os	100%	96.12%

Tablica 6.13 Preciznost tijekom korištenja test dataseta

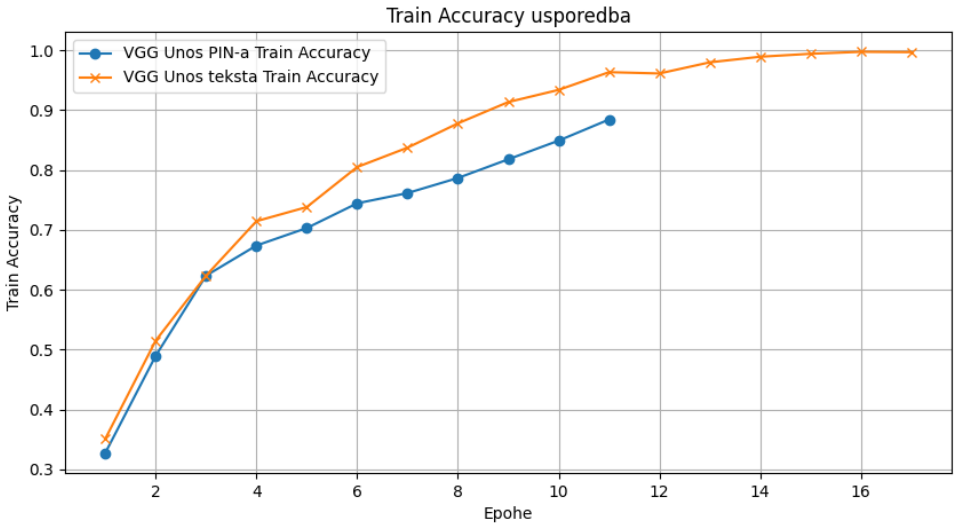
Poglavlje 7

Interpretacija rezultata

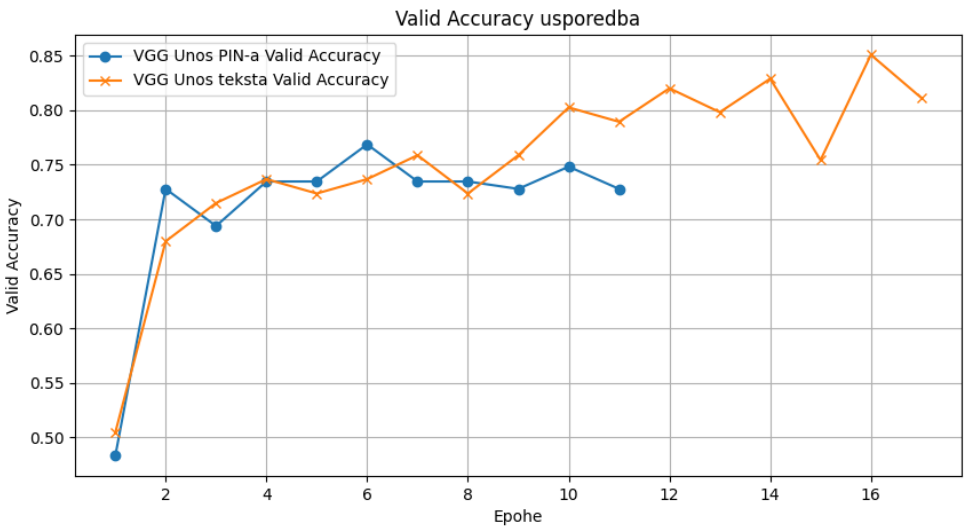
Na temelju ANOVA statističkog testa iz tablice 4.2, možemo zaključiti kako ne postoji statistički signifikantna razlika između podataka žiroskopa za korisnike, dok s druge strane, za podatke akcelerometra, postoji signifikantna razlika, koja se može vidjeti u tablici 4.3. Nadalje, uvidom u dobivene rezultate modela umjetne inteligencije, možemo zaključiti kako je unutar modaliteta upisa teksta model VGG16 bolji od modela ResNet, s obzirom da u svim validacijskim i testnim mjerenjima, VGG16 ima bolju preciznost te bolje vrijednosti za odziv i F1 rezultat.

S druge strane, u modalitetu upisa PIN-a, model ResNet ima bolje rezultate tijekom treniranja, no model VGG16 ima bolje rezultate kada se koristi s validacijskim i test skupovima podataka.

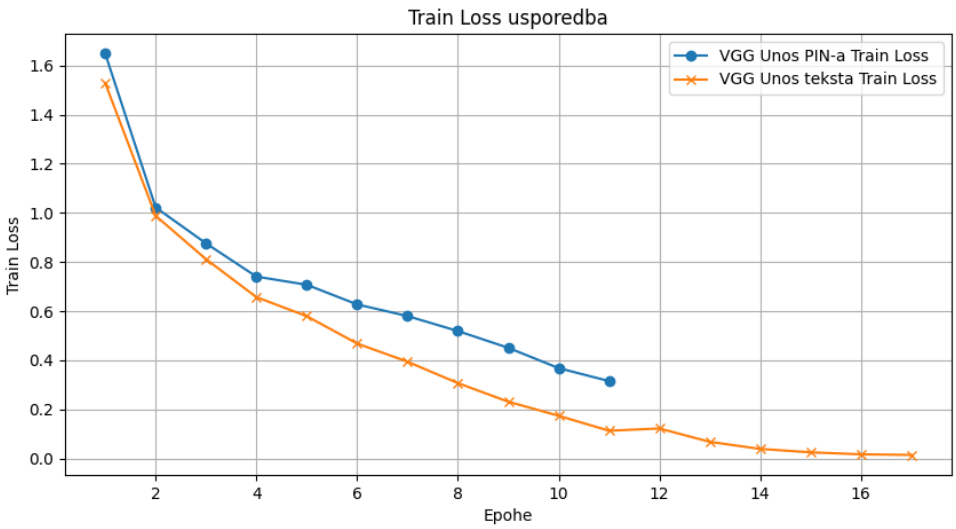
Na sljedećim fotografijama, bit će prikazana usporedba dvaju modaliteta za VGG16 model za os X. Vrijedno je spomenuti kako su podaci prikupljeni uz pomoć modaliteta unosa teksta omogućilo bolje predviđanje korisnika te će se na sljedećim fotografijama vidjeti usporedba.



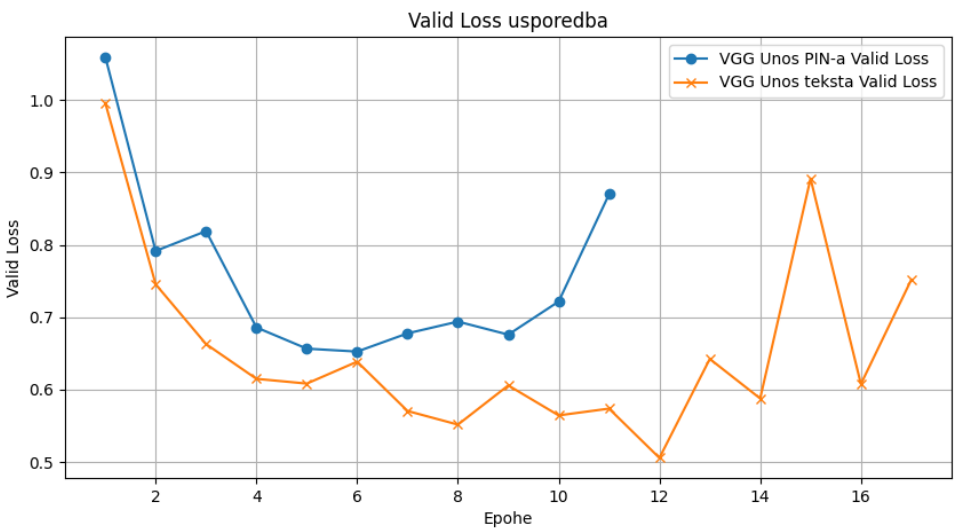
Slika 7.1 Usporedba preciznosti tijekom treniranja modela



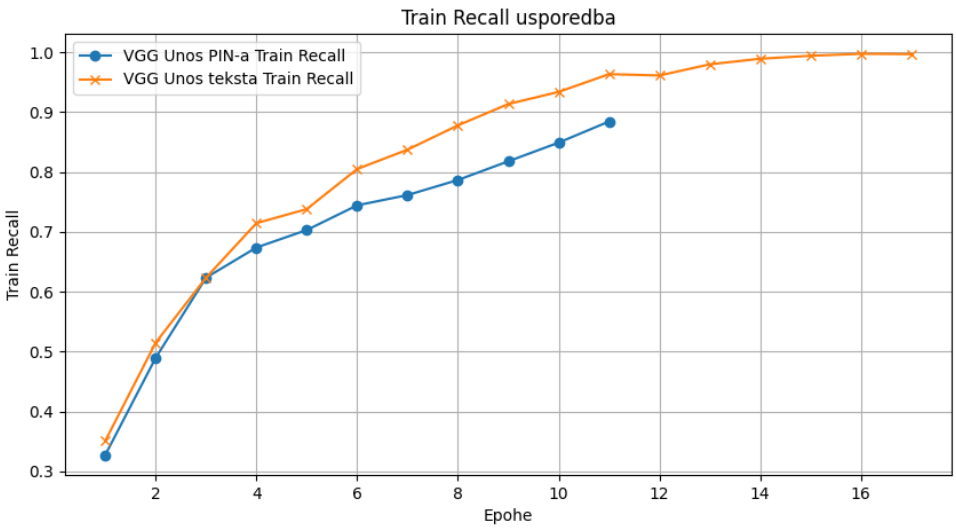
Slika 7.2 Usporedba preciznosti tijekom validacije modela



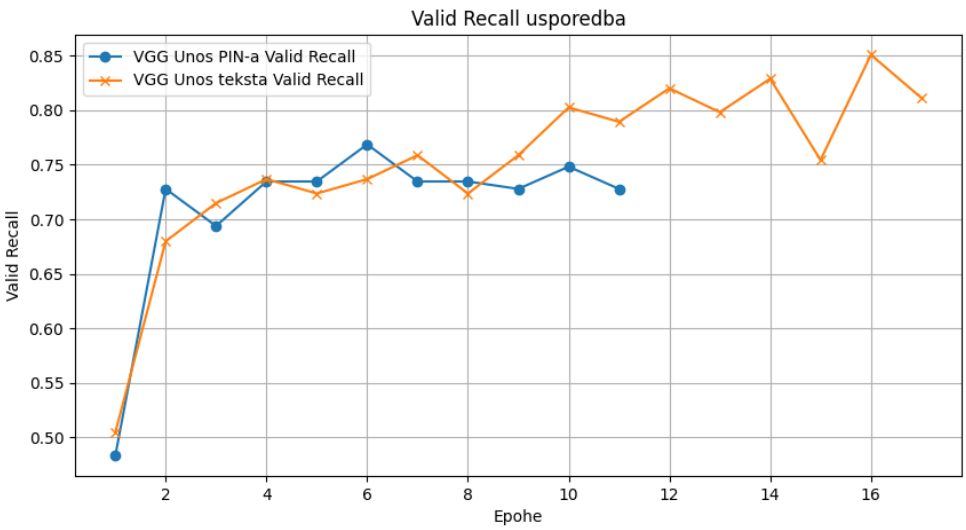
Slika 7.3 Usporedba funkcije gubitka tijekom treniranja modela



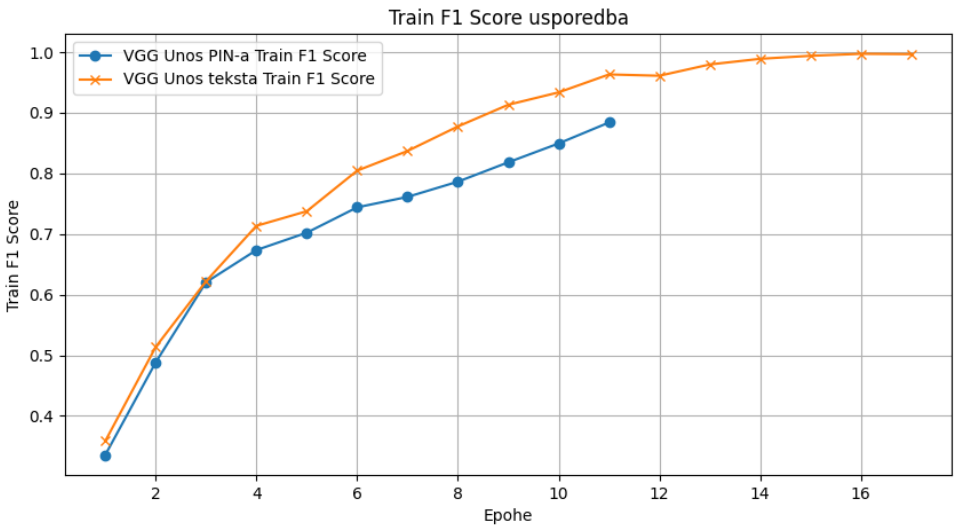
Slika 7.4 Usporedba funkcije gubitka tijekom validacije modela



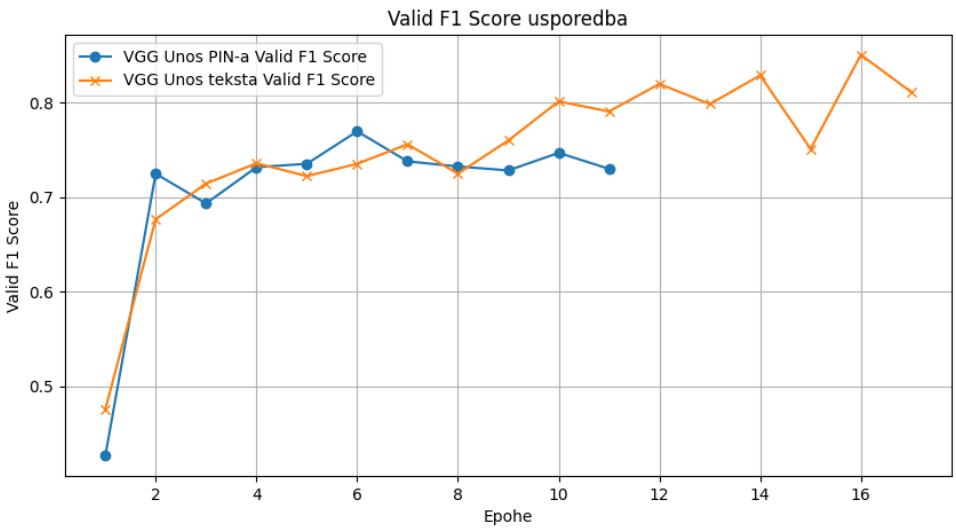
Slika 7.5 Usporedba odziva tijekom treniranja modela



Slika 7.6 Usporedba odziva tijekom validacije modela



Slika 7.7 Usporedba F1 vrijednosti tijekom treniranja modela



Slika 7.8 Usporedba F1 vrijednosti tijekom validacije modela

Poglavlje 8

Zaključak

U ovom diplomskom radu uspješno je demonstrirana primjena modela umjetne inteligencije u području bihevioralne autentifikacije uz pomoć Android mobilne aplikacije, koja prikuplja podatke iz raznih funkcionalnosti mobilnog bankarstva, poput upisa teksta i PIN-a te vrijednosti senzora žiroskopa, akcelerometra.

Kao rezultat ovog projekta, zaključeno je da podaci prikupljeni putem žiroskopskih senzora, u usporedbi s podacima akcelerometra, pokazuju statistički nesignifikantne razlike, što omogućuje njihovu primjenu tijekom treniranja modela dubokog učenja. Korišteni su modeli VGG16 i ResNet u kombinaciji s modalitetima unosa teksta i PIN-a, pri čemu je modalitet unosa teksta u kombinaciji s modelom VGG16 dao najbolje rezultate na testnom skupu podataka, što je prikazano u tablici 6.13.

Ovaj projekt potvrđuje da je integracija umjetne inteligencije i bihevioralne autentifikacije perspektivan smjer razvoja sigurnosnih rješenja te otvara mogućnosti za daljnje istraživanje i primjenu u stvarnim sustavima. Dodatna optimizacija modela i prilagodba na različite scenarije korištenja može dodatno povećati njihovu učinkovitost i prihvaćenost u širokoj primjeni.

Bibliografija

[1] A. Sharma, S. K. Singh, S. Kumar, A. Chhabra, and S. Gupta, "Security of android banking mobile apps: Challenges and opportunities," in *International Conference on Cyber Security, Privacy and Networking (ICSPN 2022)*, N. Nedjah, G. Martínez Pérez, and B. B. Gupta, Eds. Cham: Springer International Publishing, 2023, pp. 406–416.

[2] The Future Of User Authentication: A Guide To Behavioral Biometrics. , s Interneta, https://expertinsights.com/insights/a-guide-to-behavioral-biometrics/#:~:text=While%20physiological%20biometric%20authentication%20relies, machine%20learning%20(ML)%20technologies.

[3] Simon Eberz, Vincent Lenders, Kasper B. Rasmussen, Ivan Martinovic. Evaluating Behavioral Biometrics for Continuous Authentication: Challenges and Metrics. , s Interneta, https://lenders.ch/publications/conferences/asiaccs17.pdf

[4] Behavioral Biometrics Market by Component (Software Services), Application (Identity Access Management, Risk Compliance Management, Fraud Detection Prevention management), Deployment Model, Organization Size, and Vertical - Global forecast to 2023. , s Interneta, https://www.marketsandmarkets.com/Market-Reports/behavioral-biometrics-market-64844371.html

[5] C. Tam and T. Oliveira, "Literature review of mobile banking and individual performance," *International Journal of Bank Marketing*, vol. 35, no. 7, pp. 1044–1067, 2017.

[6] T. U. L. of Mobile Banking Trends and Statistics, 2024, accessed: 2024-06-24. , s Interneta, https://www.mx.com/blog/mobile-banking-stats/

[7] Python (programski jezik). , s Interneta, https://hr.wikipedia.org/wiki/Python_(programski_jezik)

[8] T. pandas development team, "pandas: Python data analysis library," 2024, accessed: 2024-05-21. , s Interneta, https://pandas.pydata.org

[9] T. N. development team, "Numpy: The fundamental package for scientific computing with python," 2024, accessed: 2024-05-21. , s Interneta, https://numpy.org

[10] T. P. development team, "Pytorch: An open source machine learning framework," 2024, accessed: 2024-05-21. , s Interneta, https://pytorch.org

[11] T. scikit-learn development team, "scikit-learn: Machine learning in python," 2024, accessed: 2024-05-21. , s Interneta, https://scikit-learn.org

[12] T. P. development team, "Torchvision: Datasets, transforms and models specific to computer vision," 2024, accessed: 2024-05-21. , s Interneta, https://pytorch.org/vision/stable/index.html

[13] ——, "Pillow: The friendly pil fork," 2024, accessed: 2024-05-21. , s Interneta, https://python-pillow.org

[14] ——, "Plotly: The interactive graphing library for python," 2024, accessed: 2024-05-21. , s Interneta, https://plotly.com/python

[15] T. M. development team, "Matplotlib: Visualization with python," 2024, accessed: 2024-05-21. , s Interneta, https://matplotlib.org

[16] P. S. Foundation, "logging — logging facility for python," 2024, accessed: 2024-05-21. , s Interneta, https://docs.python.org/3/library/logging.html

[17] L. St»hle and S. Wold, "Analysis of variance (anova)," *Chemometrics and Intelligent Laboratory Systems*, vol. 6, no. 4, pp. 259–272, 1989. , s Interneta, https://www.sciencedirect.com/science/article/pii/0169743989800954

[18] E. S. ANOVA Test: Definition, Types, 2024, accessed: 2024-06-24. , s Interneta, https://www.statisticshowto.com/probability-and-statistics/hypothesis-testing/anova/

[19] L. Durak and O. Arikan, "Short-time fourier transform: two fundamental properties and an optimal implementation," *IEEE Transactions on Signal Processing*, vol. 51, no. 5, pp. 1231–1242, 2003.

[20] "Short-time fourier transform," 2024, accessed: 2024-07-01. , s Interneta, https://en.wikipedia.org/wiki/Short-time_Fourier_transform

[21] A. Satt, S. Rozenberg, R. Hoory *et al.*, "Efficient emotion recognition from speech using deep learning on spectrograms." in *Interspeech*, 2017, pp. 1089–1093.

[22] Y. M. Costa, L. S. Oliveira, and C. N. Silla Jr, "An evaluation of convolutional neural networks for music classification using spectrograms," *Applied soft computing*, vol. 52, pp. 28–38, 2017.

[23] G. Ruffini, D. Ibañez, M. Castellano, L. Dubreuil-Vall, A. Soria-Frisch, R. Postuma, J.-F. Gagnon, and J. Montplaisir, "Deep learning with eeg spectrograms in rapid eye movement behavior disorder," *Frontiers in neurology*, vol. 10, p. 806, 2019.

[24] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. Pearson, 2016. , s Interneta, https://www.pearson.com/us/higher-education/program/Russell-Artificial-Intelligence-A-Modern-Approach-3rd-Edition/PGM263722.html

[25] H.-P. Chan, L. M. Hadjiiski, and R. K. Samala, "Computer-aided diagnosis in the era of deep learning," *Medical Physics*, vol. 47, no. 5, pp. e218–e227, 2020. , s Interneta, https://aapm.onlinelibrary.wiley.com/doi/abs/10.1002/mp.13764

[26] S. Aziz, M. Dowling, H. Hammami, and A. Piepenbrink, "Machine learning in finance: A topic modeling approach," *European Financial Management*, vol. 28, no. 3, pp. 744–770, 2022.

[27] T. T. Nguyen and G. Armitage, "A survey of techniques for internet traffic classification using machine learning," *IEEE communications surveys & tutorials*, vol. 10, no. 4, pp. 56–76, 2008.

[28] S. Kokal, M. Vanamala, and R. Dave, "Deep learning and machine learning, better together than apart: A review on biometrics mobile authentication," *Journal of Cybersecurity and Privacy*, vol. 3, no. 2, pp. 227–258, 2023.

[29] K. Gurney, *An introduction to neural networks*. CRC press, 2018.

[30] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[31] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016. , s Interneta, https://www.cv-foundation.org/openaccess/content_cvpr_2016/html/He_Deep_Residual_Learning_CVPR_2016_paper.html

[32] "Variants of resnet: A comparative analysis nayan chaure," 2024, accessed: 2024-07-01. , s Interneta, https://medium.com/@nayanchaure601/variants-of-resnet-a-comparative-analysis-63fdc1573b34

[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. , s Interneta, https://arxiv.org/abs/1409.1556

[34] S. Mascarenhas and M. Agarwal, "A comparison between vgg16, vgg19 and resnet50 architecture frameworks for image classification," in *2021 International Conference on Disruptive Technologies for Multi-Disciplinary Research and Applications (CENTCON)*, vol. 1, 2021, pp. 96–99.

[35] Wikipedia, *VGG16 Model Architecture*, srpanj 2024. , s Interneta, https://commons.wikimedia.org/wiki/File:VGG16.png

[36] R. C. Littell, "Statistical analysis of experiments with repeated measurements," *HortScience*, vol. 24, no. 1, pp. 37–40, 1989.

[37] Z. Yao, A. Gholami, S. Shen, M. Mustafa, K. Keutzer, and M. Mahoney, "Adahessian: An adaptive second order optimizer for machine learning," in *proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 12, 2021, pp. 10 665–10 673.

[38] Y. Ho and S. Wookey, "The real-world-weight cross-entropy loss function: Modeling the costs of mislabeling," *IEEE access*, vol. 8, pp. 4806–4813, 2019.

[39] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE conference on computer vision and pattern recognition*. Ieee, 2009, pp. 248–255.

[40] "Multi-class neural networks: Softmax," 2024, accessed: 2024-07-01. , s Interneta, https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/softmax

[41] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," 2019. , s Interneta, https://arxiv.org/abs/1711.05101

Popis slika

Popis tablica

Sažetak

Motivacija ovog znanstveno-istraživačkog projekta je ispitivanje hipoteze kako je moguće, uz pomoć senzora na mobilnom uređaju, prepoznati korisnika mobilne aplikacije koja simulira razne modalitete mobilnog bankarstva. Za prepoznavanje korisnika, korišten je programski jezik Python te algoritmi dubokog učenja VGG16 i ResNet. Ovim projektom možemo zaključiti kako je moguće napraviti algoritam za prepoznavanje korisnika uz pomoć senzora žiroskopa i akcelerometra, no potrebna je dodatna optimizacija već spomenutih algoritama, kako bi se mogao prepoznati korisnik sa što većom preciznosti.

Ključne riječi **— Bihevioralna autentikacija, Umjetna inteligencija, Duboko učenje**

Abstract

The motivation behind this scientific research project is to examine the hypothesis that it is possible, with the help of sensors on a mobile device, to recognize the user of a mobile application that simulates various modalities of mobile banking. For user recognition, the Python programming language and deep learning algorithms VGG16 and ResNet were used. This project concludes that it is possible to create an algorithm for user recognition using gyroscope and accelerometer sensors, but further optimization of the mentioned algorithms is needed to achieve higher accuracy in recognizing the user.

Keywords **— Behavioral authentication, Artificial Intelligence, Deep learning**