Beskontaktni unos teksta na mobilnim uređajima zasnovan na manipulaciji ambijentalnog magnetskog polja

Abdić, Lorena

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Sveučilišni diplomski studij računarstva

Diplomski rad

BESKONTAKTNI UNOS TEKSTA NA MOBILNIM UREĐAJIMA ZASNOVAN NA MANIPULACIJI AMBIJENTALNOG MAGNETSKOG POLJA

Rijeka, ožujak 2023.

Lorena Abdić

0069076313

SVEUČILIŠTE U RIJECI

TEHNIČKI FAKULTET

Sveučilišni diplomski studij računarstva

Diplomski rad

BESKONTAKTNI UNOS TEKSTA NA MOBILNIM UREĐAJIMA ZASNOVAN NA MANIPULACIJI AMBIJENTALNOG MAGNETSKOG POLJA

Mentor: izv. prof. dr. sc. Sandi Ljubić

Rijeka, ožujak 2023.

Lorena Abdić

0069076313

SVEUČILIŠTE U RIJECI
TEHNIČKI FAKULTET
POVJERENSTVO ZA DIPLOMSKE ISPITE

Rijeka, 3. ožujka 2022.

Zavod: **Zavod za računarstvo**
Predmet: **Interakcija čovjeka i računala**
Grana: **2.09.01 arhitektura računalnih sustava**

ZADATAK ZA DIPLOMSKI RAD

Pristupnik: **Lorena Abdić (0069076313)**
Studij: Diplomski sveučilišni studij računarstva
Modul: Računalni sustavi

Zadatak: **Beskontaktni unos teksta na mobilnim uređajima zasnovan na manipulaciji ambijentalnog magnetskog polja / Contactless Text Entry on Mobile Devices based on Ambient Magnetic Field Manipulation**

Opis zadatka:

U sklopu diplomskog rada potrebno je istražiti mogućnosti beskontaktnog upravljanja mobilnim uređajem, isključivo manipulacijom ambijentalnog magnetskog polja. Implementirati metodu unosa teksta koja koristi takav modalitet interakcije. Unos teksta mora se zasnivati na prepoznavanju gesti koje korisnik izvršava permanentnim magnetom u okolini mobilnog uređaja. Rješenje implementirati u formi tipkovničkog servisa za Android OS (engl. input method service) i empirijskim istraživanjem vrednovati njegovu učinkovitost.

Rad mora biti napisan prema Uputama za pisanje diplomskih / završnih radova koje su objavljene na mrežnim stranicama studija.

Zadatak uručen pristupniku: 21. ožujka 2022.

Mentor:

Doc. dr. sc. Sandi Ljubić

Predsjednik povjerenstva za diplomski ispit:

Prof. dr. sc. Kristijan Lenac

IZJAVA O AUTENTIČNOSTI DIPLOMSKOG RADA

kojom ja, Lorena Abdić (JMBAG: 0069076313), studentica Tehničkog fakulteta Sveučilišta u Rijeci, kao autorica diplomskog rada na temu:

BESKONTAKTNI UNOS TEKSTA NA MOBILNIM UREĐAJIMA ZASNOVAN NA MANIPULACIJI AMBIJENTALNOG MAGNETSKOG POLJA

odgovorno izjavljujem da sam rad izradila samostalno pod mentorstvom izv. prof. dr. sc. Sandija Ljubića. Korištena literatura na koju sam se referirala navedena je u popisu literature. Teorije, spoznaje, zaključke i radove drugih autora koje sam navela u svom radu sam u skladu s autorskim pravima ispravno referencirala.

Studentica

Lorena Abdić

ZAHVALA

Zahvaljujem se svojim roditeljima i sestri koji su mi bili neizmjerna podrška na mom akademskom putu, što su uvijek bili uz mene u svim sretnim, ali isto tako i teškim trenutcima. Bez njih sva moja postignuća ne bi bila moguća.

Zahvaljujem se svom mentoru izv. prof. dr. sc. Sandiju Ljubiću koji je uvijek bio dostupan za informacije i savjete, imao strpljenja za moje brojne upite, te pomogao organizirati prikupljanje podataka potrebnih za izradu rad.

Osobito se želim zahvaliti asist. dr. sc. Franku Hržiću na izdvojenom vremenu i pomoći tijekom izrade dijela rada vezanog za strojno učenje. Hvala Vam što ste svoje znanje podijelili sa mnom, što ste uvijek imali vremena i strpljenja za brojne mail-ove i konzultacije, te na kraju izdvojili još dodatnog vremena za testiranje konačne aplikacije.

SADRŽAJ

1. UVOD

Mobilni uređaji dio su ljudske svakodnevnice, a zahvaljujući njihovoj dostupnosti i malim dimenzijama većina ljudi ih u gotovo svakom trenutku ima uz sebe. Omogućavaju brz pristup informacijama iz cijeloga svijeta, stupanje u kontakt s drugim ljudima, ali i zabavu u obliku računalnih igara u mobilnoj domeni. Veliki udio u interakciji čovjeka s mobilnim uređajem odnosi se na unos tekstualnih podataka, bilo da se radi o pretraživanju Interneta, korištenju usluga za razmjenu poruka (engl. messaging) ili zapisivanju podsjetnika. Kako je mobilni internet postajao sve dostupniji širim masama ljudi, društvene mreže i razne SMS alternative u obliku aplikacija za besplatno slanje poruka, dovele su do povećanja tekstualnih unosa na mobilnim uređajima. Također su popularni i grupni razgovori koji pomažu prilikom dogovora i donošenja odluka, kao i za što brže slanje informacija većem broju ljudi.

Unos teksta na mobilnom uređaju uglavnom se odvija putem tipkovnice. Međutim, danas se rijetko vide uređaji s fizičkim tipkovnicama. Dolaskom pametnih telefona sa zaslonima osjetljivima na dodir tipkovnice su postale virtualne. Iako se u standardnim postavkama nalazi opcija za 3x4 tipkovnicu, korisnici se češće služe QWERTZ i QWERTY dizajnima. Njihova prednost je što, kao i tipkovnice računala, daju izravan pristup svim slovima, pa čak i brojevima. Nažalost, zbog malih dimenzija zaslona mobilnih uređaja, to je ujedno i njihova najveća mana jer se na tipkovnici u svakom trenutku nalazi veliki broj malih tipki. Njihova veličina također varira ovisno o jeziku koji se koristi. Primjerice, kada se na hrvatskoj tipkovnici nalaze i slova č, ć, š, ž i đ, tada sve tipke postanu još tanje, pa je teže pogoditi željenu. Iz želje za što preciznijim unosom teksta došlo je do rapidnog razvoja virtualnih tipkovnica.

Neki od alternativnih načina unosa teksta na mobilnim uređajima su pretvorba govora u tekst, *swype* tipkovnica i rukopis. Kod *swype* tipkovnice riječi se pišu tako da se prstom klizi po tipkovnici i povezuje slova, a podizanje prsta označava kraj riječi i razmak. Međutim, navedeni načini unosa teksta također su skloni greškama. Primjerice, ako se govor snima na lokaciji s dosta pozadinske buke, mikrofon ga neće uspjeti pravilno snimiti. Kod *swype* tipkovnice također treba biti precizan s odabirom tipki, a pisanje teksta prstom dosta je nespretno pogotovo u slučaju dužih riječi.

Na temu problematike preciznog unosa teksta odrađena su mnoga istraživanja te implementirana razna zanimljiva rješenja. Neka od njih se koriste interakcijom oko uređaja (engl. Around Device Interaction, ADI). U takvom pristupu uređaju se naredbe zadaju gestama koje se mogu raspoznati pomoću kamere ili raznih, unaprijed ugrađenih senzora. Budući da se djeluje u 3D prostoru, mogućnosti interakcije su velike.

U sklopu ovoga diplomskog rada izrađena je aplikacija za raspoznavanje gesti pisanja slova u 3D prostoru oko uređaja. Korisnik interakciju s uređajem vrši pomoću permanentnog magneta kojeg drži u ruci i njime utječe na magnetsko polje oko uređaja. Magnetometar ugrađen u uređaj mjeri nastale promjene u magnetnom polju te one služe za razlikovanje pojedinih gesti. Kako bi interakcija u potpunosti bila beskontaktna, početak i kraj pisanja slova te sve ostale potrebne funkcionalnosti (npr. razmak i brisanje slova) ostvaruju se kombinacijom magnetometra i senzora blizine. Geste pisanja slova izvode se iznad uređaja, a za klasifikaciju se koristi *scikit-learn* model nasumične šume (engl. Random Forest, RF). Odluke o lokaciji pisanja slova u odnosu na uređaj i klasifikacijskom modelu donesene su na temelju rezultata eksperimenata izvođenih tijekom razvoja aplikacije.

2. SRODNI RADOVI

Na temu interakcije oko uređaja odrađena su mnoga istraživanja i stručno-znanstveni radovi. Neki od njih se isključivo fokusiraju na, primjerice, upravljanje glazbom ili unos teksta, dok postoje i oni koji pružaju veće mogućnosti upravljanja uređajem. Osim različitih namjena, prisutni su i različiti načini implementacije rješenja. Uz razne vrste senzora, koji su unaprijed ugrađeni u današnje mobilne uređaje, za postizanje interakcije često je korištena i kamera uređaja. Međutim, kako je cilj rada bio implementirati unos teksta pomoću permanentnog magneta, proučeni su radovi koji se za implementaciju interakcije s mobilnim uređajem koriste isključivo magnetometrom. Za potrebe smještanja rada unutar šireg područja interakcije oko uređaja, izdvojeno je nekoliko srodnih radova koji se bave istom ili sličnom tematikom.

Sustav *MagiText* [1], namijenjen je isključivo za unos teksta pomoću magneta. Mobilna aplikacija koristi magnetometar kako bi tijekom pokreta pisanja slova u 3D prostoru ispred uređaja pratila promjene u magnetnom polju. Na temelju nastalih promjena raspoznaju se slova. Za pisanje slova potrebno je pratiti jedan od dva skupa gesti, *Graffiti* ili *EdgeWrite*. U oba skupa svako slovo ima točno određenu gestu, odnosno jedan potez kojim je potrebno napisati slovo. Za klasifikaciju podataka korišten je SVM (engl. Support Vector Machine) klasifikator, a rezultati su pokazali da preciznost kod Graffiti skupa iznosi 81.2%, dok kod EdgeWrite skupa postiže 89.4%. Navedeni rad je proširenje rješenja *MagiWrite* čija je zadaća unos znamenki pomoću magneta. *MagiWrite* [2] također koristi geste u 3D prostoru ispred uređaja, ali nema unaprijed definirane geste za svaku znamenku, već omogućava korisnicima da pohrane svoj predložak pisanja te ga kasnije učitaju i koriste.

Unos znamenki, uz još dodatne funkcionalnosti, omogućava i *MagiThings* [3]. Pomoću gesti i magnetometra moguće je vršiti interakciju s korisničkim sučeljem, provjeriti autentičnost korisnika, igrati videoigre te upravljati glazbom. Eksperimentom je dokazana vrlo visoka preciznost prepoznavanja jednostavnih i složenih gesti i to u iznosu od 91.4%. Slanje naredbi mobilnom uređaju također je implementirano i u rješenju *MagiTact* [4]. Gestama omogućava skaliranje sadržaja, okretanje stranica, prihvaćanje i odbijanje poziva, upravljanje glazbom te interakciju s videoigrama. Eksperiment je proveden sa šest gesti i pet korisnika, a rezultati su pokazali točnost prepoznavanja gesti veću od 90%.

Još dva vrlo zanimljiva rada su *MagiSign* i *MagNail*. *MagiSign* [5] pruža identifikaciju i autentifikaciju korisnika pomoću potpisa napisanog magnetom u 3D prostoru oko uređaja. U uređaj se prvo registriraju potpisi, a zatim se prilikom korištenja svaki novi potpis uspoređuje s modelima stvorenim na temelju već postojećih potpisa. Takve je potpise, u usporedbi s potpisima na papiru, teže kopirati zbog kretanja u tri dimenzije. Karakteristika po kojoj se *MagNail* [6] ističe je korištenje malenog magneta koji se dodaje kao ukras na oslikane nokte. Zatim se magnetom može upravljati unutar dvije mobilne aplikacije. Prva aplikacija obuhvaća kontekst crtanja i u njoj se, u njoj se zavisno od položaja prsta s magnetom, funkcionalnost mijenja između crtanja i brisanja. U drugoj aplikaciji se ikone premještaju na lijevu ili desnu stranu zaslona ovisno o tome drži li se uređaj u ruci na kojoj je prisutan magnet ili u ruci bez magneta.

3. PODACI

U ovom poglavlju opisani su podaci na kojima će se zasnivati implementirano rješenje, aplikacija za njihovo prikupljanje, kalibracija magnetskog senzora i eksperimenti provedeni za određivanje modaliteta pisanja slova.

3.1. Aplikacija za prikupljanje podataka

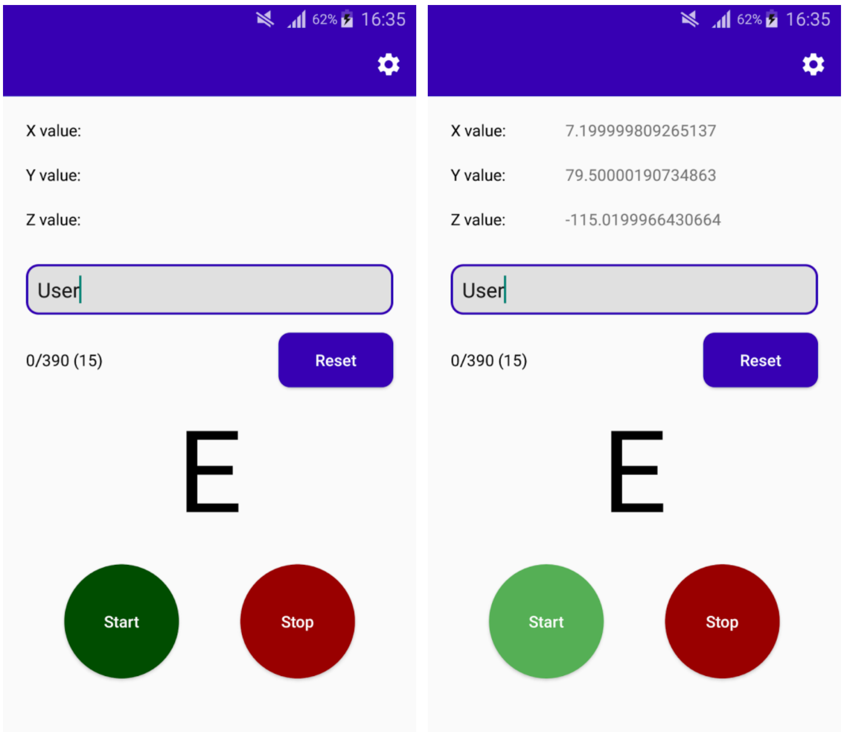
Za početak je bilo potrebno odrediti koji podaci mogu biti korisni prilikom treniranja modela za klasifikaciju. Promjene u magnetskom polju koje mjeri magnetometar (X, Y i Z vrijednosti), kao i slovo (klasa) koje korisnik u određenom trenutku upisuje odgovarajućim gestama odmah su se istaknuli kao najvažniji i najlogičniji. Uz njih, prikupljano je još nekoliko vrsta informacija. Krajnji popis podataka koji su prikupljani je sljedeći:

- slovo koje korisnik ostvaruje gestom
- identifikator korisnika
- kalibracijski podaci magnetskog polja
  - ΔX
  - ΔY
  - ΔZ
- t_0 - vrijeme pri početku pisanja
- t_N - vrijeme pri završetku pisanja
- t - vrijeme u sekundama potrebno za napisati slovo $\left(t = \frac{t_N - t_0}{1000}\right)$
- magnetski otisak slova, tj. očitani nizovi vrijednosti magnetskog polja (X, Y, Z)

Vrijednosti spremljene pod slovo kasnije će se tretirati kao klase, a također će imati i ulogu u filtriranju podataka prilikom izrade JSON datoteka. Korisnike se ne zapisuje pod punim imenom i prezimenom, već im se daje korisnička oznaka. Navedeni podatak se sprema radi filtriranja i lakše organizacije podataka. Tri vrijednosti koje se pohranjuju pod kalibraciju su vrijednosti magnetskog polja oko uređaja prije nego se na njega utječe permanentnim magnetom. Ti podaci su od velike važnosti jer nam omogućavaju da, neovisno o lokaciji na kojoj unosimo tekst u uređaj, rezultati uvijek budu jednaki. Podaci za početak, kraj i trajanje pisanja pojedinog slova daju ipak nešto manje informacija o kojem je slovu riječ. S obzirom da različiti korisnici mogu isto slovo pisati različitim redoslijedom poteza, vrijeme pisanja može uvelike varirati. Međutim, i ti su podaci ipak pohranjeni. Skupu podataka se na kraju dodaje i magnetski otisak slova, odnosno vektor (X, Y, Z) struktura (umanjenih za odgovarajuću delta vrijednost) svake pozicije u kojoj se magnet našao tijekom izvođenja odgovarajuće geste.

Aplikacija za prikupljanje podataka se sastoji od glavne aktivnosti u kojoj se snimaju promjene vrijednosti magnetskog polja tijekom pisanja slova te aktivnosti s postavkama aplikacije.

U glavnoj aktivnosti (Slika 3.1.) se tijekom snimanja pokreta pisanja slova prikazuju iznosi X, Y i Z vrijednosti magnetskog polja. Ispod njih se nalazi polje za unos oznake korisnika. Brojač govori koliko je slova u svakom trenutku upisano te koji je konačan cilj, a brojka u zagradi označava koliko puta će se ponoviti cijela abeceda. Prije snimanja gesti potrebno je kliknuti na gumb "Reset" koji će osigurati da brojač krene od nule, prikazati prvo slovo koje korisnik mora unijeti, kao i gumbe za početak i kraj snimanja. Nakon što korisnik napiše zadano slovo i pritisne gumb za kraj snimanja, svi potrebni podaci se spremaju u lokalnu bazu uređaja (Slika 3.2.) te se korisniku zadaje novi zadatak upisivanja. Kako bi skup podataka bio što više raznolik, slova se ne zadaju abecednim redoslijedom, a isto slovo se može pojaviti maksimalno dva puta zaredom.



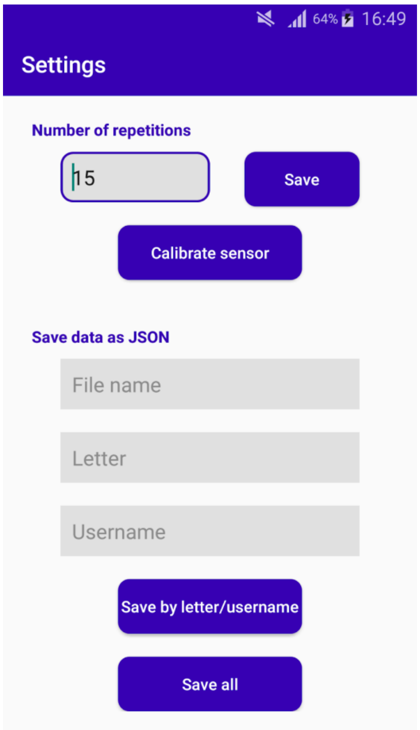
Slika 3.1. Glavna aktivnost aplikacije za prikupljanje podataka

id	letter	user	deltaX	deltaY	deltaZ	t0	tN	t	pointList
1	W	User2	-3.7799999	146.039993	-132.47999	1669129988451	1669129993199	4.748	{"point":[[-31.199999570846558,-30.77999
2	M	User2	-34.139999	146.279998	-126.59999	1669130013453	1669130017860	4.407	{"point":[[-6.239997863769531,-5.9399986
3	F	User2	-34.139999	146.279998	-126.59999	1669130065362	1669130068927	3.565	{"point":[[-5.220001220703125,-4.680000
4	I	User2	-34.139999	146.279998	-126.59999	1669130077729	1669130080321	2.592	{"point":[[-4.259998321533203,-4.8600006
5	L	User2	-34.139999	146.279998	-126.59999	1669130084225	1669130087167	2.942	{"point":[[3.780000686645508,3.180000305

Slika 3.2. Primjer zapisa u lokalnoj bazi uređaja

U postavkama (Slika 3.3.) se nudi mogućnost odabira broja ponavljanja abecede, kalibracije senzora te pohranjivanje podataka u obliku JSON datoteke. Kalibraciju senzora je potrebno uvijek napraviti prije nego se započne s prikupljanjem podataka. Pri tome, magnet treba držati što dalje od uređaja, kako bi se dobio iznos magnetskog polja bez utjecaja magneta. Delta vrijednosti se pohranjuju u *shared preferences* te ostaju nepromijenjene sve do sljedeće kalibracije. Tijekom pisanja slova, vrijednosti magnetskog polja se umanjuju za odgovarajuće delta vrijednosti. Za pohranjivanje svih podatka iz lokalne baze u jednu JSON datoteku upisuje se samo naziv datoteke koja će biti stvorena i potvrđuje se odgovarajućim gumbom. Za pohranjivanje određenog skupa podataka upisuje se:

- korisnik - sve instance svih slova definiranog korisnika,
- slovo - sve instance definiranog slova svih korisnika ili
- korisnik i slovo - sve instance definiranog slova i korisnika.



Slika 3.3. Postavke aplikacije za prikupljanje podataka

Stvorene JSON datoteke se na uređaj pohranjuju u mapu aplikacije. U prilogu A dostupna je funkcija za zapisivanje podataka u JSON format, a na Slici 3.4. prikazan je isječak iz generirane JSON datoteke.


```
[
  {
    "letter": "L",
    "user": "User1",
    "calibration": {
      "deltaX": -27.18000030517578,
      "deltaY": -9.119999885559082,
      "deltaZ": -71.76000213623047
    },
    "t0": 1670359635829,
    "tN": 1670359637167,
    "t": 1.338,
    "fingerprint": [
      [
        -37.019996643066406,
        -13.320000648498535,
        -66.5999984741211
      ],
      [
        -36.959999084472656,
        -13.380000114440918,
        -66.54000091552734
      ],
```

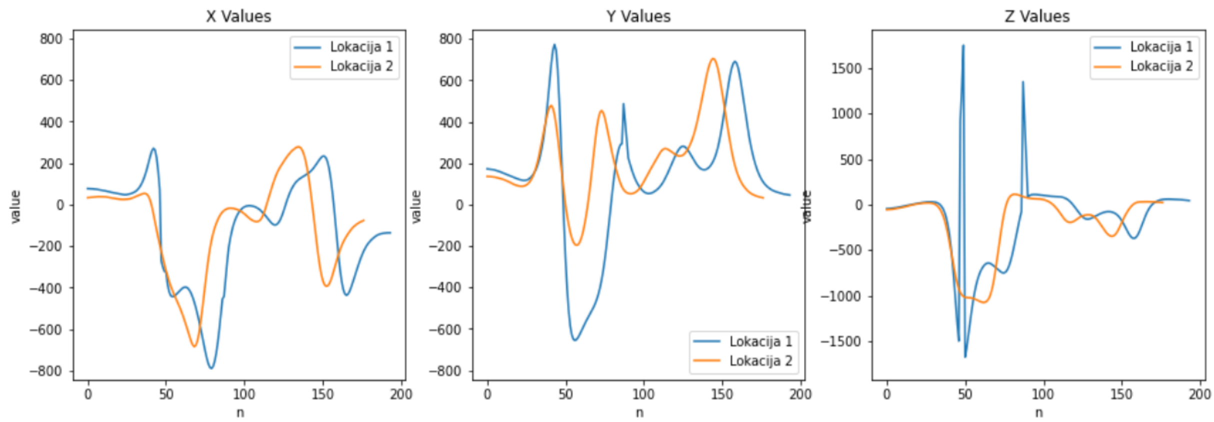

Slika 3.4. Primjer JSON datoteke s prikupljenim podacima

3.2. Kalibracija

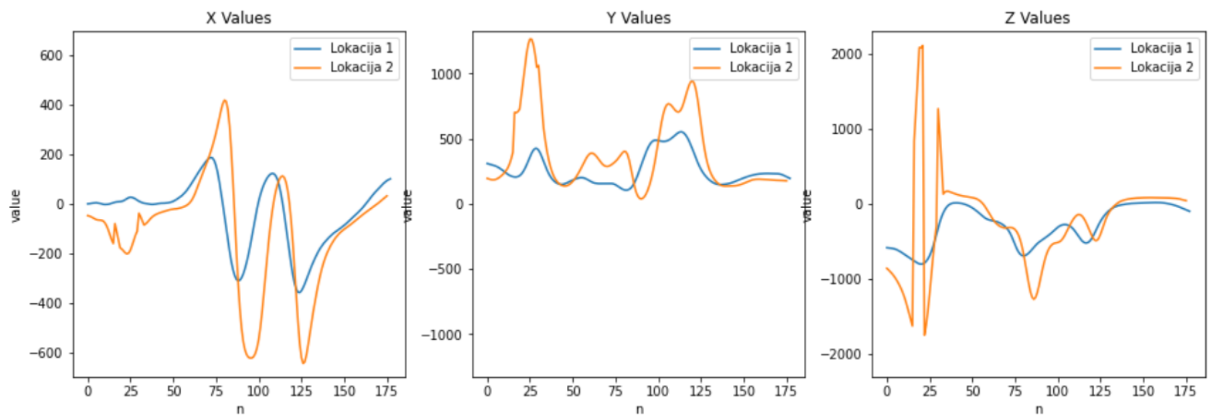
Ideja iza kalibracije je da se definiraju i izmjere tri varijable (ΔX, ΔY i ΔZ) u koje se pohranjuju početne vrijednosti magnetskog polja dok utjecaj permanentnog magneta nije prisutan, a zatim se sva iduća očitanja magnetskog polja umanjuju za te vrijednosti. Ovim postupkom se zapravo početni iznos magnetskog polja u kontekstu izvršavanja pojedinačne geste postavlja u nulu i dobiva točan iznos utjecaja permanentnog magneta.

Iz navedenoga slijedi pretpostavka da će se primjenom kalibracije, neovisno o lokaciji, uvijek dobiti jednaki rezultati ako se slovo piše istim potezima. Za dokazivanje navedene pretpostavke proveden je početni eksperiment. Odabrana su slova A i B radi razlike u potezima, a zatim su ona napisana na dvije različite lokacije. Pri tome su na svakoj od lokacija slova jednom napisana iznad mobilnog uređaja, a drugi put pored. Dobivene X, Y i Z vrijednosti prikazane su grafički radi lakšeg donošenja zaključka.

Na Slici 3.5. prikazana je usporedba kako se X, Y i Z vrijednosti s vremenom mijenjaju prilikom pisanja slova A iznad uređaja. Primjetno je da krivulje imaju vrlo slične oblike, iako su na prvoj lokaciji amplitude nešto veće. Kod rezultata za slovo B koji su prikazani na Slici 3.6. može se doći do istog zaključka, jedino su u ovom slučaju amplitude veće na drugoj lokaciji. Ovakav rezultat je bio i očekivan jer kod pisanja slova u zraku permanentni magnet je vrlo teško neprekidno držati na jednakoj udaljenosti od zaslona mobilnog uređaja. Također, kod oba slova najveći nesklad je u vrijednosti Z na prvoj polovici pisanja slova.

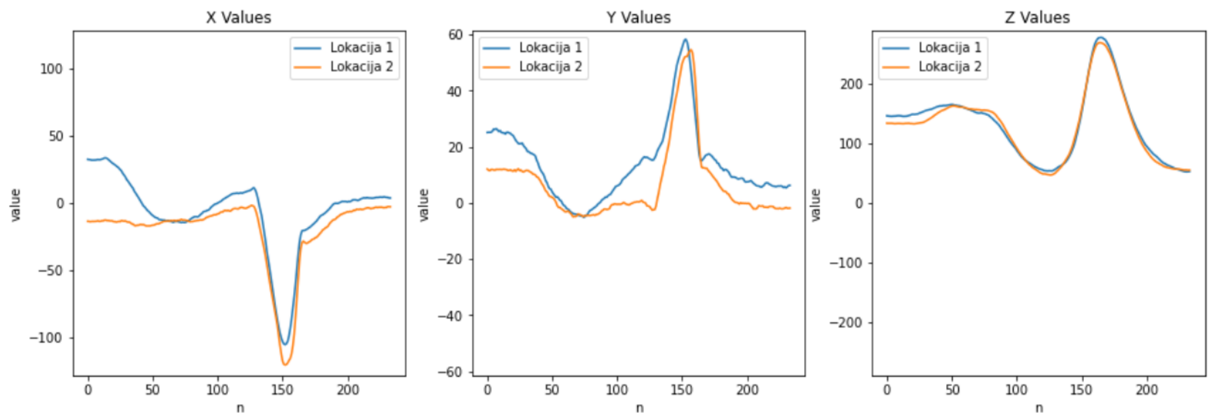


Slika 3.5. Promjena X, Y i Z vrijednosti tijekom pisanja slova A iznad uređaja

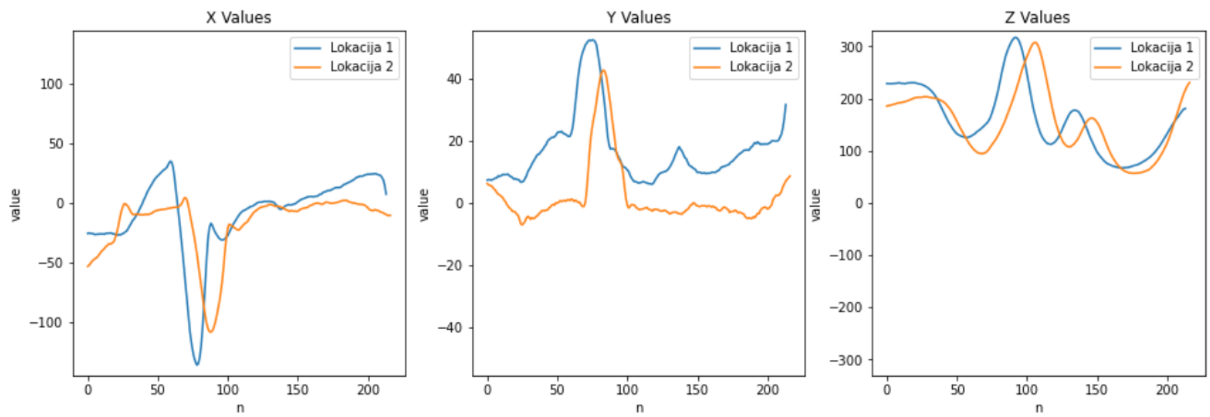


Slika 3.6. Promjena X, Y i Z vrijednosti tijekom pisanja slova B iznad uređaja

Prije pisanja slova pored mobilnog uređaja, na listu papira naznačeno je gdje treba odložiti uređaj te je nacrtan okvir unutar kojega će se pisati slova. U ovom slučaju slova će se uvijek pisati na jednakoj visini jer će se magnet pomicati po papiru koji se nalazi na ravnoj podlozi (stolu) isto kao i mobilni uređaj. Tako dobivene X, Y i Z vrijednosti magnetskog polja za slovo A prikazane su na Slici 3.7. iz koje je vidljivo da, kada se eliminira visina držanja magneta, dobiveni rezultati su gotovo identični. Isto je vidljivo i na Slici 3.8. koja prikazuje rezultate za slovo B. Usprkos nešto većoj razlici amplituda kod Y vrijednosti, ostale dvije vrijednosti se uglavnom podudaraju.

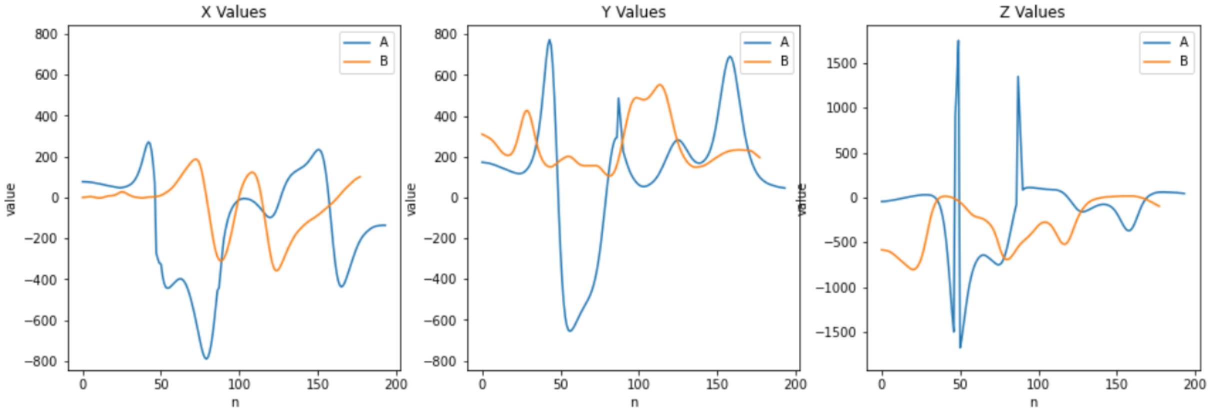


Slika 3.7. Promjena X, Y i Z vrijednosti tijekom pisanja slova A pored uređaja.

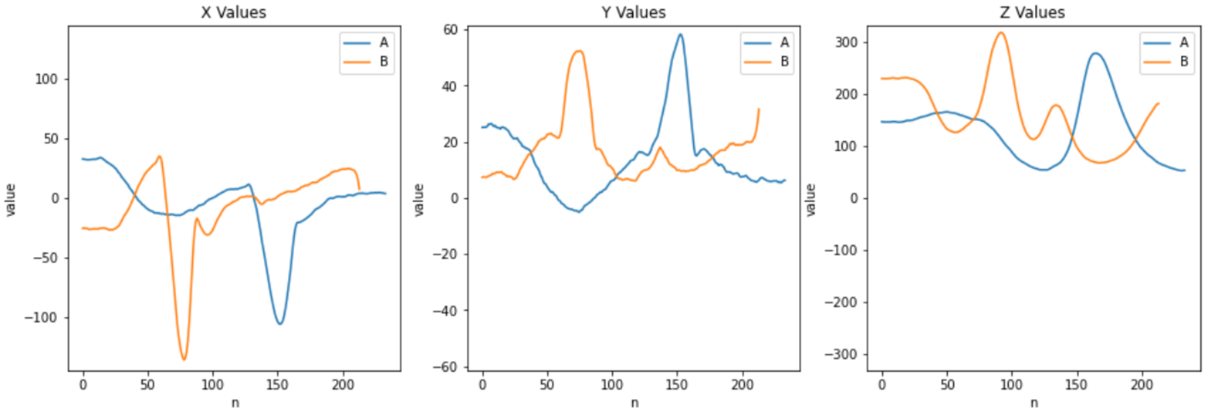


Slika 3.8. Promjena X, Y i Z vrijednosti tijekom pisanja slova B pored uređaja

Iz prikazanih grafova donesen je zaključak da kalibracija zaista pomaže pri dobivanju jednakih rezultata neovisno o iznosu ambijentalnog magnetskog polja, tj. lokaciji na kojoj su slova napisana. Također je zanimljivo usporediti i koliko se međusobno razlikuju grafovi slova A i B, pa je napravljena usporedba vrijednosti izmjerenih na prvoj lokaciji. Iako kod obje usporedbe postoje lako uočljive razlike, ovi rezultati dovode do pretpostavke da bi pisanje slova iznad mobilnog uređaja (Slika 3.9.) moglo dati bolje podatke za klasifikaciju od pisanja slova pored uređaja (Slika 3.10.).



Slika 3.9. Usporedba slova A i B pisanih iznad uređaja



Slika 3.10. Usporedba slova A i B pisanih pored uređaja

3.3. Određivanje položaja pisanja slova

Nakon dokazivanja pretpostavke o kalibraciji, te prije nego se započne s prikupljanjem većeg skupa podataka za treniranje modela, potrebno je odrediti hoće li se u konačnoj aplikaciji slova pisati iznad ili pored mobilnog uređaja.

Za tu je potrebu, uz pomoć dvoje ispitnih korisnika, prikupljen manji skup podataka. Podaci se sastoje od trideset ponavljanja abecede iznad uređaja te još trideset ponavljanja pored uređaja. Kako dobiveni podaci imaju tri dimenzije (svaka "točka" geste ima svoju X, Y i Z vrijednost) teško ih je međusobno uspoređivati u njihovom izvornom obliku. Stoga je provodena redukcija dimenzionalnosti, odnosno dimenzionalnost podataka je smanjena sa tri dimenzije na dvije. S obzirom da se redukcija provodi nad skupom podataka, potrebno je prvo osigurati da sva slova imaju jednak broj "točaka". Potom su podaci normalizirani i na kraju je provedena redukcija dimenzionalnosti koristeći sljedeće metode:

- t-SNE
- ICA
- PCA

Dobiveni rezultati su uspoređeni putem mapiranja u 2D prostoru. Iz dobivenih 2D prikaza donesen je zaključak da je klase, tj. slova, najlakše grupirati i međusobno razlikovati kada se redukciju izvede t-SNE metodom. Kako bi se takva odluka dodatno potvrdila, konstruirana je i prosječna 2D slika svakog slova. U nastavku poglavlja opisana je priprema podataka za redukciju dimenzionalnosti, predstavljene su navedene metode te prikazani grozdovi i prosječne slike slova dobivene redukcijom.

3.3.1. Priprema podataka za redukciju dimenzionalnosti

Prilikom učitavanja podataka iz JSON datoteka zapisan je i broj točaka (X, Y, Z uređenih trojki) koje svako slovo sadrži. Za određivanje broja točaka koje svako slovo morati sadržavati izračunata je aritmetička sredina, odnosno zbroj točaka svih slova podijeljen je s brojem slova u skupu podataka.

Programskom petljom se uzima svako slovo iz skupa podataka, računa se razlika između njegovog i zadanog broja točaka te se, ovisno o rezultatu, brišu postojeće ili dodaju nove točke, sve dok razlika ne postane jednaka nuli (prilog B). U slučaju kada je razlika pozitivan broj znači da slovo sadrži više od željenog broja točaka. Uzima se jedna nasumična točka, briše se iz polja, a zatim se vrijednost varijable u kojoj je pohranjen broj točaka trenutnog slova smanjuje za jedan. Ukoliko je razlika broj manji od nule, uzima se indeks jedne nasumične točke i njoj prateće, za svaku od X, Y i Z vrijednosti se računa aritmetička sredina te se tako dobivena nova točka ubacuje između ranije odabranih. Također se vrijednost varijable u kojoj je pohranjen broj točaka povećava za jedan.

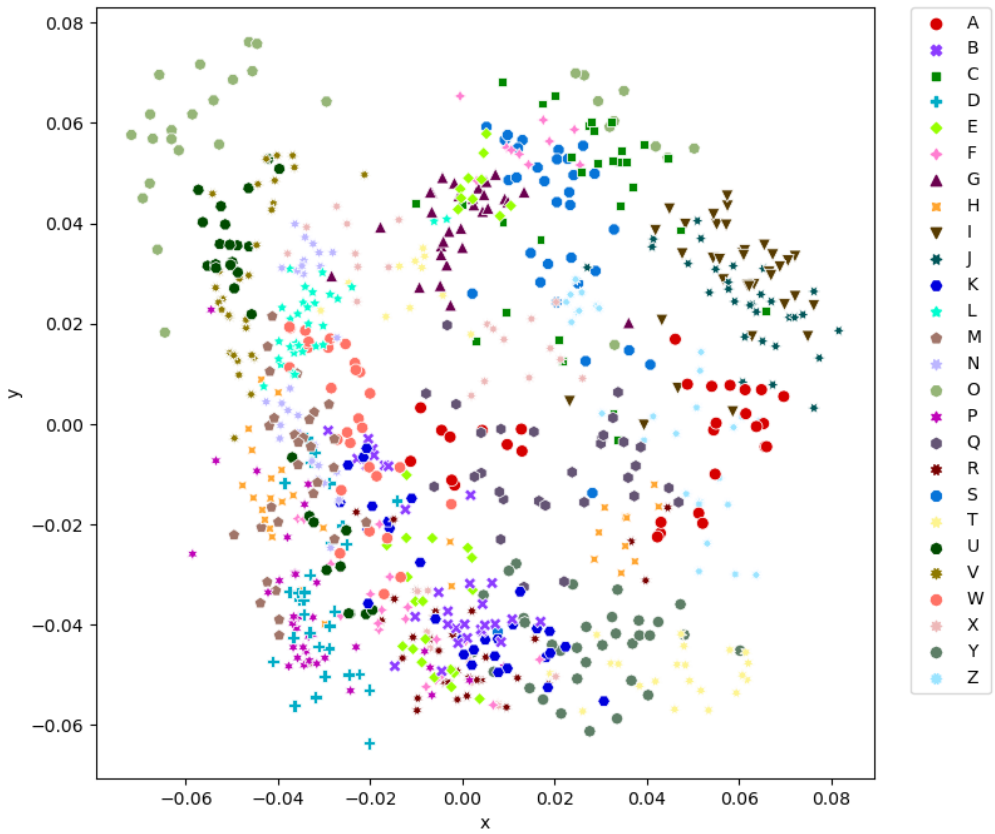
Prije same redukcije dimenzionalnosti potrebno je još provesti normalizaciju podataka. Normalizacijom se podaci skaliraju na vrijednosti u rasponu između nula i jedan (Slika 3.11.). Korišten je *MinMaxScaler* koji je dio paketa za predobradbu podataka unutar knjižnice *scikit-learn*.

```
[[0.92307691 0.57499973 0.28211116]
 [0.91941388 0.58571449 0.27884167]
 [0.92673988 0.61785697 0.27650629]
 ...
 [0.91208787 0.23928523 0.04950958]
 [0.93406594 0.23214266 0.04950958]
 [0.91941388 0.22857138 0.04764125]]
```

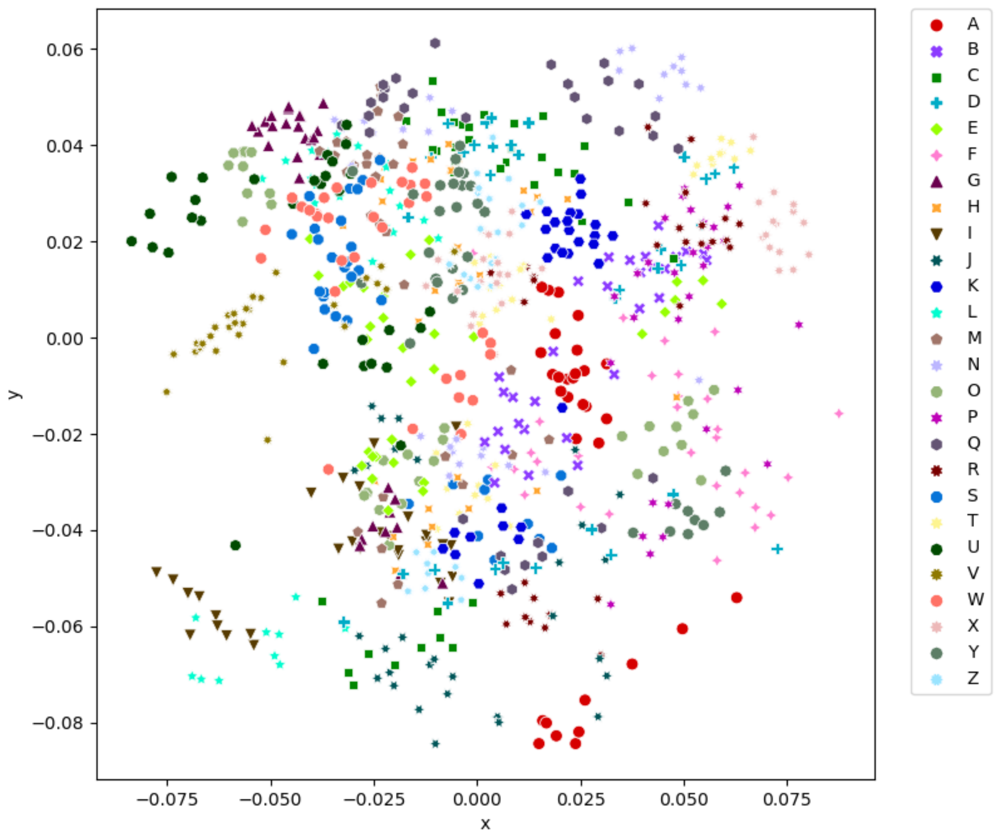
Slika 3.11. Primjer normaliziranih podataka vrijednosti magnetskog polja

3.3.2. ICA i PCA

ICA (*Independent Component Analysis*) je računalna metoda za razdvajanje multivarijantnog signala u aditivne podkomponente [7]. To se postiže pretpostavkama da je najviše jedna podkomponenta Gaussova krivulja te da su sve podkomponente međusobno nezavisne. Uporaba ICA je vrlo učestala kod obrade zvuka jer je zvuk signal kojeg u svakom trenutku t čini zbroj više signala koji potječu iz različitih izvora. Na Slici 3.12. prikazan je rezultat ICA metode redukcije dimenzionalnosti podataka dobivenih pisanjem slova iznad mobilnog uređaja, a na Slici 3.13. podataka dobivenih pisanjem slova pored uređaja. Uočljivo je da slova pisana iznad uređaja daju nešto bolje rezultate, prepoznatljivi su pojedini grozdovi, međutim njihovo međusobno preklapanje je i dalje preveliko.

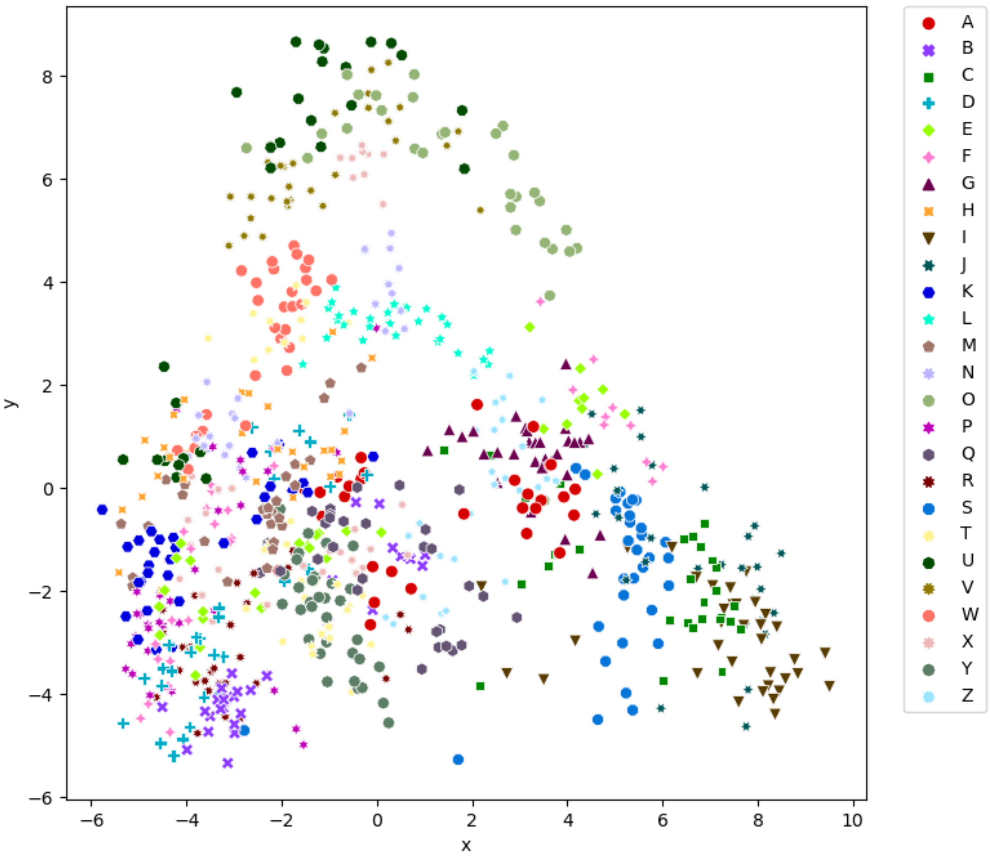


Slika 3.12. ICA redukcija dimenzionalnosti podataka za slova pisana iznad uređaja

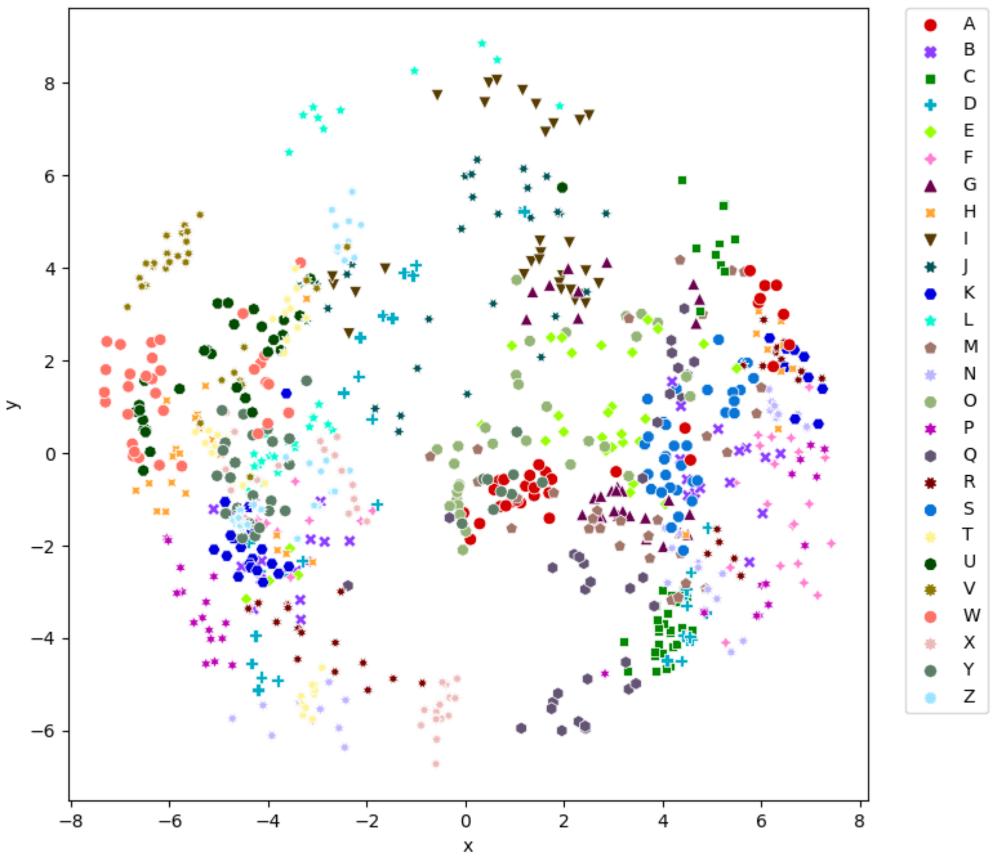


Slika 3.13. ICA redukcija dimenzionalnosti podataka za slova pisana pored uređaja

PCA (*Principal Component Analysis*) je statistička metoda za reduciranje dimenzionalnosti skupa podataka na način da se podaci linearno transformiraju u novi koordinatni sustav [8]. U takvom koordinatnom sustavu treba biti moguće što veću količinu podataka opisati sa manje dimenzija u odnosu na originalne podatke. Rezultat redukcije dimenzionalnosti PCA metodom za podatke dobivene pisanjem slova iznad mobilnog uređaja prikazan je na Slici 3.14., a za podatke dobivene pisanjem pored uređaja na Slici 3.15. U oba slučaja grozdovi su teško prepoznatljivi jer je previše međusobnih preklapanja.



Slika 3.14. PCA redukcija dimenzionalnosti podataka za slova pisana iznad uređaja



Slika 3.15. PCA redukcija dimenzionalnosti podataka za slova pisana pored uređaja

3.3.3. t-SNE

t-SNE (*T-distributed Stochastic Neighbor Embedding*) je nelinearna metoda redukcije dimenzionalnosti koja je vrlo pogodna za preoblikovanje visokodimenzionalnih podataka u niskodimenzionalne kako bi ih se moglo jednostavno vizualizirati pomoću dvije ili tri dimenzije [9]. Svaki visokodimenzionalni objekt se modelira na način da mu se što sličniji objekti nalaze na što manjoj udaljenosti, a objekti sa što više različitosti na što većoj udaljenosti.

Metodu t-SNE je razvio Laurens van der Maaten predloživši t-distribuiranu varijantu SNE metode koju su razvili Sam Roweis i Geoffrey Hinton. t-distribucija, poznata još kao i Studentova t-distribucija, opisuje standardizirane udaljenosti srednjih vrijednosti uzoraka od srednje vrijednosti populacije kada standardna devijacija populacije nije poznata i opažanja dolaze iz normalno distribuirane populacije. t-SNE algoritam ima dva glavna koraka:

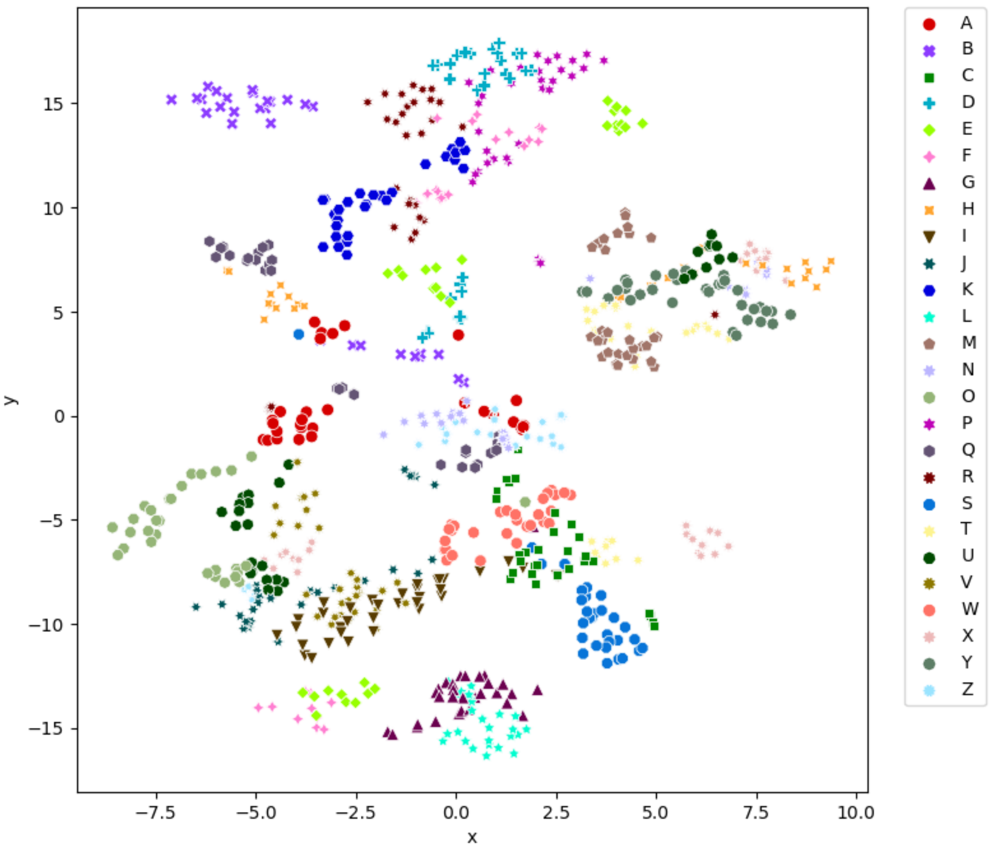
1. preko parova visokodimenzionalnih objekata konstruira se distribucija vjerojatnosti tako što se sličnim objektima dodjeljuje veća vjerojatnost, a različitim objektima manja vjerojatnost;

2. definira se slična distribucija vjerojatnosti preko točaka na niskodimenzionalnoj karti te se minimizira Kullback-Leiblerova divergencija između dviju distribucija.

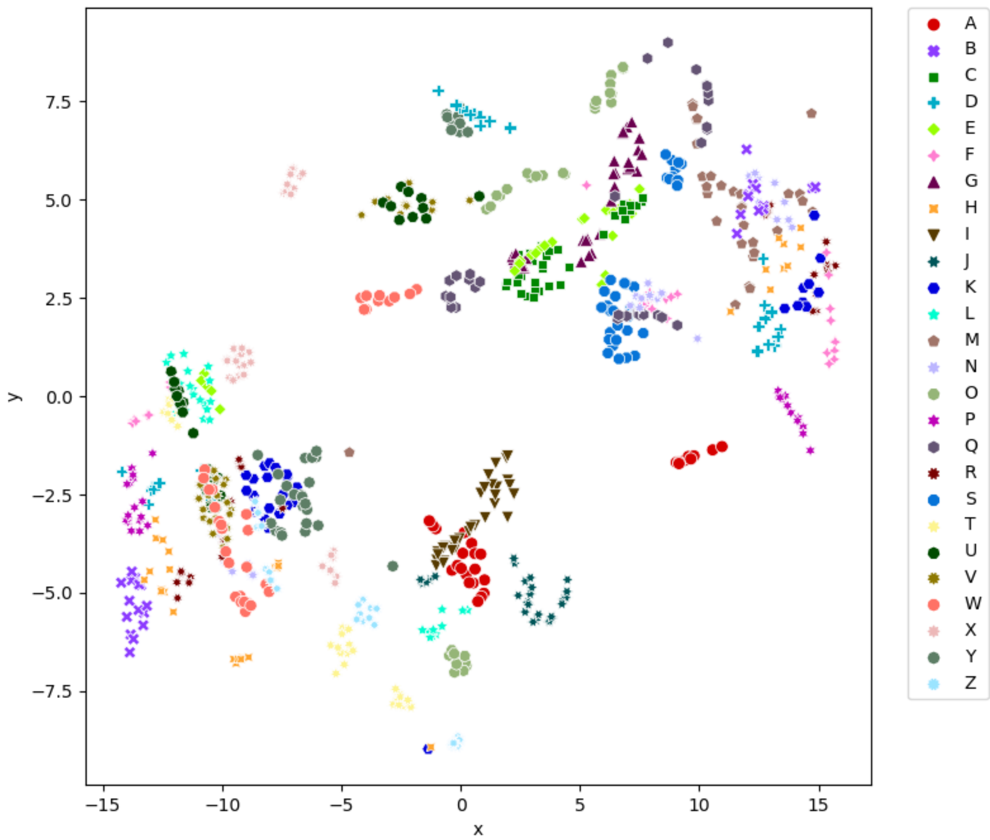
Kullback-Leiblerova divergencija, također poznata kao relativna entropija ili l-divergencija, je mjera koja nam govori koliko se jedna distribucija vjerojatnosti P razlikuje od druge distribucije vjerojatnosti Q. Označava se na sljedeći način:

$$D_{KL} = (P \mid\mid Q) \tag{3.1}$$

Na Slici 3.16. prikazani su rezultati redukcije dimenzionalnosti pomoću t-SNE metode za podatke dobivene pisanjem slova iznad uređaja. Rezultati redukcije dimenzionalnosti podataka dobivenih pisanjem slova pored uređaja prikazani su na Slici 3.17. Kod oba slučaja grozdovi su prepoznatljivi, međutim kod podataka slova pisanih iznad uređaja bolje su razdvojeni.



Slika 3.16. t-SNE redukcija dimenzionalnosti podataka za slova pisana iznad uređaja

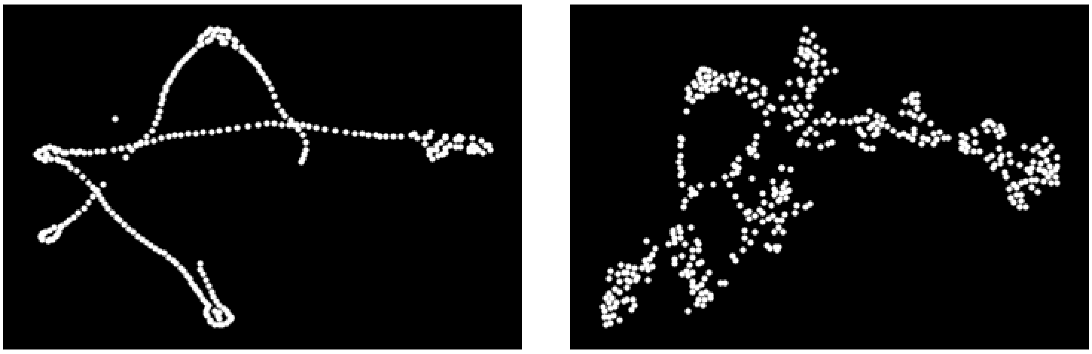


Slika 3.17. t-SNE redukcija dimenzionalnosti podataka za slova pisana pored uređaja

3.3.4. Usporedba položaja pisanja slova pomoću 2D slika

Drugi pristup kojim je dodatno potvrđeno da je pisanje slova iznad mobilnog uređaja pogodnije je konstruiranje prosječne slike svakog slova. Na taj način je moguće vidjeti koliko su slova međusobno slična te po kojim obilježjima ih se može razlikovati.

Prvo je potrebno za svako slovo izraditi pripadajući 2D prikaz. Korišten je isti skup podataka i *MinMaxScaler* kao u prethodnom postupku, međutim u ovom slučaju nije potrebno sva slova svesti na jednak broj točaka pošto se redukcija dimenzionalnosti provodi za svako slovo posebno. Za redukciju dimenzionalnosti korištena je samo t-SNE metoda, budući da se ona u prethodnom postupku pokazala najboljom. Dobiveni podaci su potom pomoću Matplotlib knjižnice prikazani *scatter* plotom te pohranjeni kao slika. Primjer rezultata za slovo A napisano iznad i pored mobilnog uređaja prikazan je na Slici 3.18. Na njoj se jasno vidi da je u slučaju pisanja slova pored uređaja prisutan značajan šum.



Slika 3.18. Primjer slova A pisanog iznad (lijevo) i pored uređaja (desno)

Nakon što su sva slova iz skupa podataka dobila pripadajući 2D prikaz, konstruirana je prosječna slika pojedinog slova. Postupak se izvodi u nekoliko koraka:

1. uzima se jedna klasa (tj. slovo) i učitaju se sve njene slike
2. definira se novo 2D polje sa dimenzijama jednakim dimenzijama slika i svim elementima jednakim nuli
3. prolazi se kroz sve retke i stupce slike, a za svaku vrijednost veću od nule se u novom polju element na istim indeksima uveća za jedan
4. traži se maksimalna vrijednost u novom polju
5. prolazi se kroz sve retke i stupce novog polja, svaki element se podijeli sa maksimalnom vrijednošću te pomnoži sa 255
6. ukoliko je dobivena vrijednost manja od 50, vrijednost elementa se postavi u nulu
7. pomoću Matplotlib knjižnice prikaže se i pohrani dobivena slika

Svim elementima novog 2D polja se za početak pridaje vrijednost jednaka nuli kako bi ono predstavljalo "praznu" sliku s crnom pozadinom. Svaki element polja predstavlja jedan piksel slike. Ukoliko na slici koja sadrži slovo određeni piksel ima vrijednost veću od nule, to znači da je taj piksel dio slova. Zato se u novom polju, elementima koji se nalaze na istom indeksu kao i pikseli koji su dio slova, vrijednost poveća za jedan (Ispis 3.1.). Novo polje će tako imati označene sve piksele koji su dijelovi slova. Kako bi se dobila prosječna slika slova, odnosno slika koja će prikazati na kojim pozicijama se najčešće nalaze dijelovi slova, elementi polja su dodatno skalirani na vrijednosti između 0 i 255. Ukoliko je vrijednost elementa manja od 50, tada ju se prebacuje u nulu kako bi se ignorirale manje učestale pozicije (Ispis 3.2.).

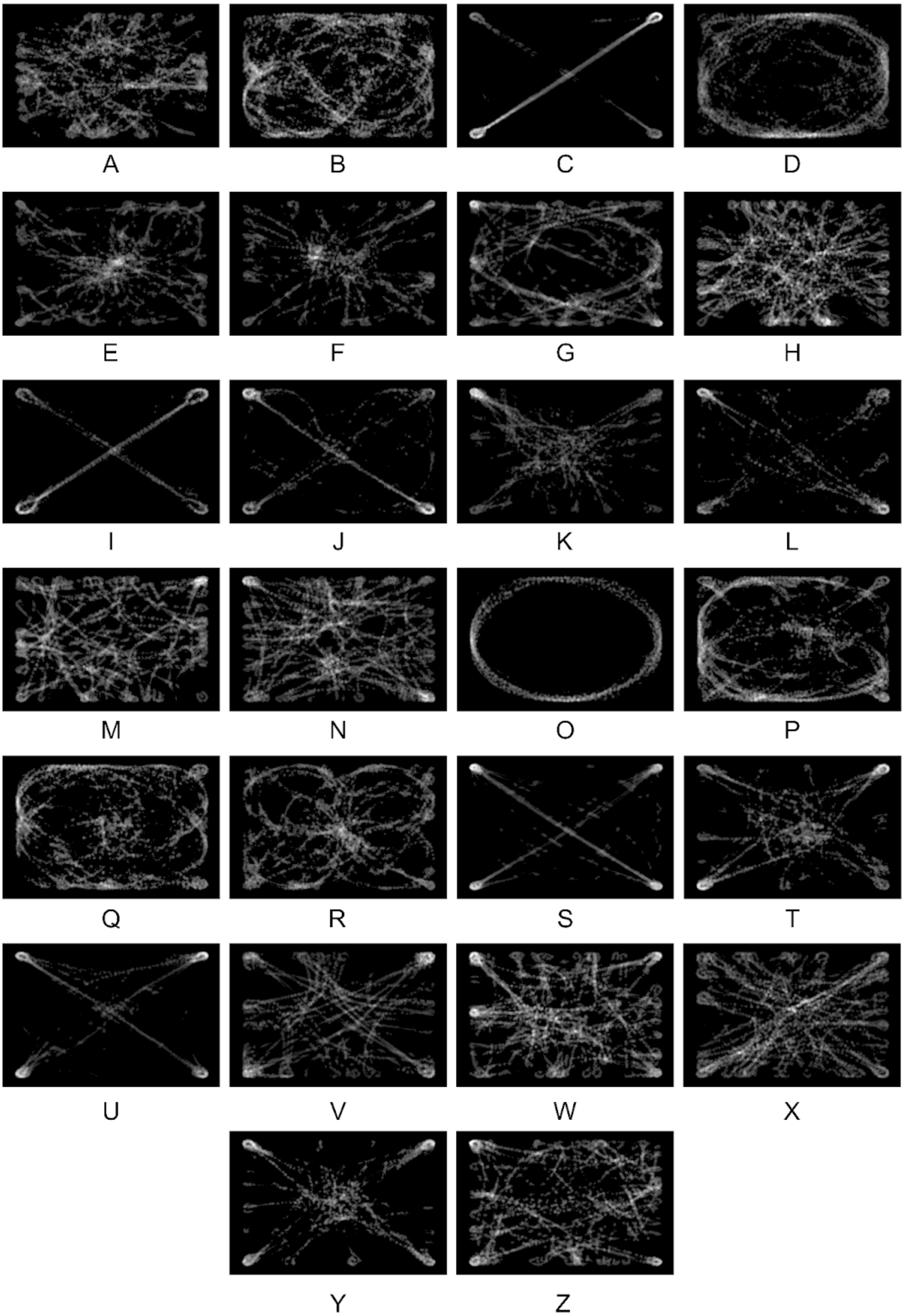
Ispis 3.1. Funkcija za zbrajanje piksela koji su dio slova

```
def get_image_sum(dataset, rows_num, colls_num):
    image = np.zeros((rows_num, colls_num))

    for image in dataset:
        for i in range(rows_num):
            for j in range(colls_num):
                if image[i][j] > 0:
                    image_sum[i][j] += 1

    return image_sum
```

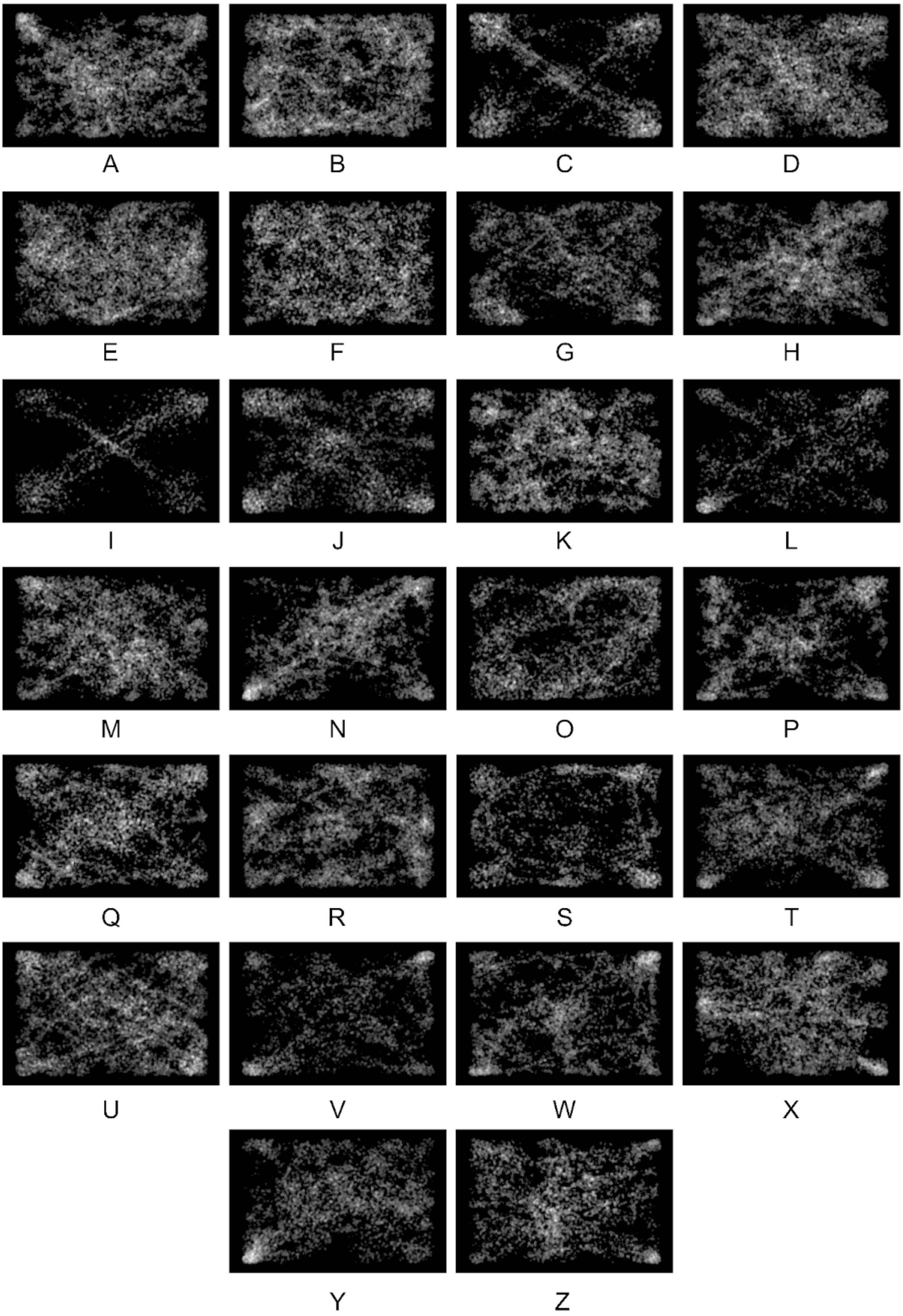
Ispis 3.2. Skaliranje vrijednosti piksela nove slike

```
def get_reconstruction(average_image, max_num):
    rec_image = np.zeros((rows_num, colls_num))

    for i in range(rows_num):
        for j in range(colls_num):
            rec_image[i][j] = average_image[i][j] / max_num
            rec_image[i][j] = round(rec_image[i][j] * 255)

        if rec_image[i][j] < 50:
            rec_image[i][j] = 0

    return rec_image
```

Nakon uspoređivanja rezultata za slova pisana iznad mobilnog uređaja (Slika 3.19.) i slova pisanih pored uređaja (Slika 3.20.) donesen je konačni zaključak da pisanje slova iznad uređaja uistinu daje prikladnije podatke za klasifikaciju. Iako je nemoguće s potpunom preciznošću razlikovati sva slova (npr. C i I), ipak su uočljivi pojedini detalji (npr. razlika u zaobljenosti D i O). Kod slova pisanih pored uređaja postoji dosta šuma, a pretpostavka je da je to posljedica većeg udaljavanja magneta od mobilnog uređaja. Prilikom pisanja iznad uređaja visina na kojoj se magnet nalazi nije uvijek jednaka, ali variranje je puno manje u usporedbi s postavljanjem magneta pored ruba uređaja te njegovim daljnjim odmicanjem tijekom pisanja.



Slika 3.19. Prosječni izgled slova pisanih iznad uređaja



Slika 3.20. Prosječni izgled slova pisanih pored uređaja

4. KLASIFIKACIJA I IMPLEMENTACIJA

Skup podataka kojim će se trenirati model za klasifikaciju prikupljen je pomoću petnaest korisnika, koristeći istu aplikaciju kao i kod ranije opisanih eksperimenata. Korisnicima je zadatak bio magnetom pisati velika tiskana slova engleske abecede iznad uređaja, ponoviti cijelu abecedu petnaest puta i to redoslijedom koji zadaje aplikacija. Nije određen poseban redoslijed pokreta za pisanje pojedinog slova. Cilj je bio prikupiti što raznolikiji skup podataka kako bi konačno rješenje bilo što fleksibilnije u pogledu raspoznavanja gesti tijekom pisanja.

4.1. Određivanje klasifikacijskog modela

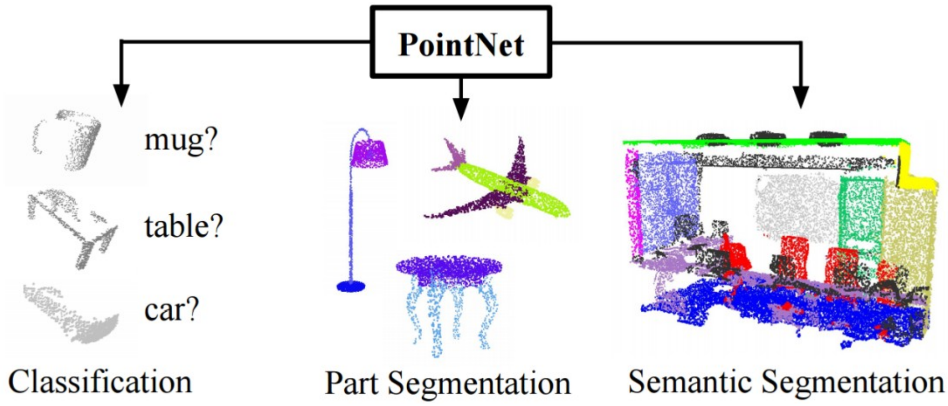
Nakon što su podaci prikupljeni bilo je potrebno pronaći odgovarajući model strojnog učenja koji će što kvalitetnije vršiti klasifikaciju. Problemu je pristupljeno na nekoliko načina. Za početak su podaci tretirani kao 3D oblaci točaka (engl. point clouds), a kasnije pretvoreni u 2D slike. Na kraju su najbolje rezultate postigli modeli implementirani u *scikit-learn* knjižnici kojima su dani i 2D i 3D oblici podataka. Osobito se istaknuo RF model koji je nakon treniranja s 3D podacima postigao preciznost klasifikacije od 76%. Povećanjem predikcije sa jednog slova s najvećom vjerojatnošću na četiri, navedeni model postiže preciznost od 93%. Daljnje povećanje broja slova ne daje značajnije poboljšanje. Stoga je za klasifikaciju odabran RF model koji će korisniku ponuditi četiri najvjerojatnija slova, a korisnik će potom odabrati ono koje želi upisati.

U nastavku rada ukratko su predstavljene sve metode kojima se pokušala postići klasifikacija, a zatim detaljnije prikazan način rada RF modela te dobiveni rezultati.

4.1.1. PointNet i 3D CNN

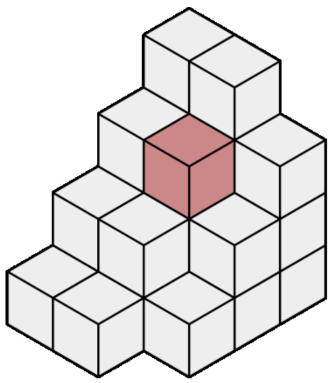
S obzirom da skup podataka korišten u radu sadrži točke u 3D prostoru, za početak su istražene metode posebno prilagođene za rad s takvom vrstom podataka.

Prva takva metoda je PointNet, neuronska mreža prilagođena za rad s oblacima točaka, koja osim klasifikacije pruža i mogućnost segmentacije (Slika 4.1.) [10].



Slika 4.1. Mogućnosti PointNet neuronske mreže [10]

Pri radu s oblacima točaka podaci se uobičajeno formatiraju u 3D mreže voksela ili zbirku slika. Vokseli su 3D ekvivalenti pikselima, a na Slici 4.2. prikazan je primjer voksela. Međutim, tako dobiveni podaci su velikog obujma i mogu izazvati probleme tijekom treniranja modela. PointNet je stoga razvijen na način da izravno učita oblake točaka te koristi *max pooling* - operaciju koja mapu značajki podijeli na nekoliko manjih segmenata, traži maksimalnu vrijednost svakog segmenta i pomoću njih stvara novu, manju mapu značajki. Tako neuronska mreža nauči skup optimizacijskih kriterija koji odabiru zanimljive ili informativne točke iz oblaka i enkodiraju razlog odabira. Na kraju, jedan potpuno povezani sloj mreže spaja naučene optimalne vrijednosti u globalni deskriptor za cijeli oblak i njime dodjeljuje klasu [11].



Slika 4.2. Primjer skupa voksela [12]

Druga istražena metoda za klasifikaciju 3D podataka je 3D konvolucijska neuronska mreža (engl. Convolutional Neural Network, CNN). Način na koji radi je isti kao i 2D CNN, samo se konvolucija pomoću filtera vrši nad višestrukim 2D matricama [13]. U usporedbi s 2D CNN, broj operacija se množi s veličinom filtera i veličinom samog ulaznog skupa podataka. Iz navedenoga se može odmah zaključiti da treniranje ovakvom metodom zahtijeva više vremena i memorije.

Tijekom implementiranja dviju navedenih metoda, pojavljivali su se problemi s formatiranjem podataka. Na temelju testiranja u procesu modeliranja donesen je zaključak da one nisu primjenjive na podatke korištene u radu i metodu kojom su podaci prikupljeni. PointNet se primarno koristi nad podacima pohranjenim u datoteke *.off* formata dobivenih snimanjem stvarnih 3D objekata pomoću senzora dubine (npr. LIDAR i RGB-D kamera). Kod 3D CNN korištene su slike kojima je dodana dubina, primjerice MNIST skup podataka pisanih znamenki [13] ili podaci dobiveni slaganjem slojeva slika kao što su CT snimke pluća [14].

4.1.2. Klasifikacija 2D slika pomoću *scikit-learn* modela

Nakon što se prethodno navedene metode namijenjene radu s 3D podacima nisu pokazale uspješnima, klasifikaciji je pristupljeno s 2D podacima. Korištene su slike napravljene postupkom redukcije dimenzionalnosti. Slike prikazuju podatke koji su normalizirani i reducirani na dvije dimenzije, a dodatno im je veličina promijenjena na 50x50 piksela. Slike su klasificirane pomoću nekoliko modela strojnog učenja implementiranih u *scikit-learn* knjižnici te su dobiveni rezultati preciznosti prikazani u Tablici 4.1.

Tablica 4.1. Rezultati klasifikacije slika slova pomoću scikit-learn modela

Model	**Preciznost klasifikacije**
Random forest (RF)	22%
K-nearest neighbours	5%
Decision tree	16%
Gaussian naive bayes	14%

Skup podataka kojim su dobiveni navedeni rezultati sadrži 5849 slika, a omjer za treniranje i testiranje je 80% - 20%, respektivno. Prije podjele podaci su izmiješani. Potom je skup podataka povećan morfološkim operacijama erozijom i dilatacijom, te rotiranjem slika ulijevo (45, 90, 135 i 180 stupnjeva) i udesno (-45, -90 i -135 stupnjeva). Novi skup podataka posjeduje 76037 slika, a omjer je ponovno 80% - 20%. Modeli su ponovno istrenirani, a dobiveni rezultati su prikazani u Tablici 4.2.

Tablica 4.2. Rezultati klasifikacije slika slova nakon augmentacije skupa podataka

Model	Preciznost klasifikacije
Random forest (RF)	43%
K-nearest neighbours	32%
Decision tree	23%
Gaussian naive bayes	10%

4.1.3. Klasifikacija 3D podataka pomoću *scikit-learn* modela

S obzirom da su *scikit-learn* modeli dali bolje rezultate od prijašnjih metoda, provedeno je treniranje i sa 3D podacima. Nakon učitavanja podataka iz JSON datoteka bilo je potrebno osigurati da svako slovo ima jednak broj točaka. Naime, isto kao i kod redukcije dimenzionalnosti, i modeli za klasifikaciju zahtijevaju da svi podaci imaju jednak broj značajki. Podaci su zatim izmiješani te podijeljeni na 80% za treniranje i 20% za testiranje. Istrenirani modeli su dali rezultate prikazane u Tablici 4.3.

Tablica 4.3. Rezultati klasifikacije 3D podataka pomoću scikit-learn modela

Model	Preciznost klasifikacije
Random forest (RF)	76%
K-nearest neighbours	71%
Decision tree	48%
Gaussian naive bayes	24%

Rezultate se zatim dodatno poboljšalo uzimanjem u obzir prve četiri klase s najvećom vjerojatnošću. Tako formalizirane preciznosti klasifikacije nalaze se u Tablici 4.4.

Tablica 4.4. Rezultati klasifikacije 3D podataka za top-4 klase

Model	Preciznost klasifikacije
Random forest (RF)	94%
K-nearest neighbours	91%
Decision tree	53%
Gaussian naive bayes	49%

Za prvih pet klasa s najvećom vjerojatnošću rezultati dobivaju minimalno poboljšanje u odnosu na prethodne. To je prikazano u Tablici 4.5.

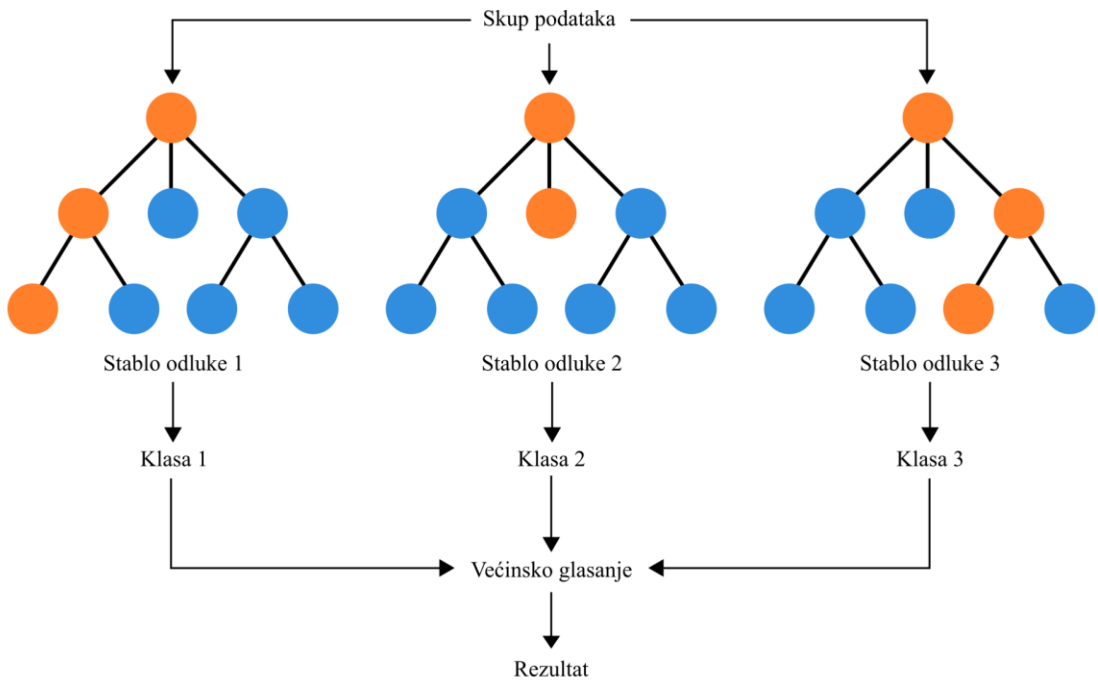
Tablica 4.5. Rezultati klasifikacije 3D podataka za top-5 klasa

Model	Preciznost klasifikacije
Random forest (RF)	95%
K-nearest neighbours	91%
Decision tree	55%
Gaussian naive bayes	56%

S obzirom na dobivene rezultate donesena je konačna odluka da se u aplikaciju za unos teksta integrira upravo model RF.

4.1.4. RF model

RF (engl. Random Forest) je algoritam strojnog učenja koji do rezultata dolazi na osnovu kombiniranja izlaza više stabala odluke (engl. decision trees) [15]. Primjer RF modela prikazan je na Slici 4.3. Radi svoje jednostavnosti i fleksibilnosti postao je jedan od često korištenih modela koji istovremeno nudi mogućnost klasifikacije i regresije. Kod klasifikacije, izlaz RF-a je klasa koju je izabrala većina stabala, dok kod regresije vraća srednju ili prosječnu vrijednost svih stabala. RF također ispravlja problem pretjerane prilagodbe (engl. overfitting) skupu podataka za treniranje koji se javlja kod stabla odluke. Algoritam koji se danas uvelike koristi razvili su Leo Breiman i Adele Cutler. Njihov rad je proširenje prvog RF algoritma kojeg je napravila Tin Kam Ho koristeći metodu slučajnog potprostora (engl. random subspace method) [16].



Slika 4.3. Princip rada modela RF

S obzirom da se RF sastoji od višestrukih stabala odluke, otkud i dolazi riječ šuma (engl. forest) u nazivu algoritma, potrebno je prvo razumjeti kako pojedino stablo djeluje. Svako stablo odluke ima hijerarhijsku strukturu koja se sastoji od korijena, grana, čvorova i listova [17]. Algoritam započinje od korijena koji je granama povezan s čvorovima na sljedećoj razini dubine stabla. Ovisno o odluci koja se donese u korijenu, odgovarajućom granom se prelazi na jedan od navedenih čvorova. Svaki idući čvor se također dijeli na dvije ili više grana, te ovisno o odlukama koje se u njima donose, algoritam prolazi kroz stablo sve dok ne dođe do lista. List je čvor iz kojeg ne izlaze grane te sadrži klasu koja se dodjeljuje ulaznom podatku. Broj listova koje sadrži stablo odluke jednako je broju klasa koje sadrži skup podataka.

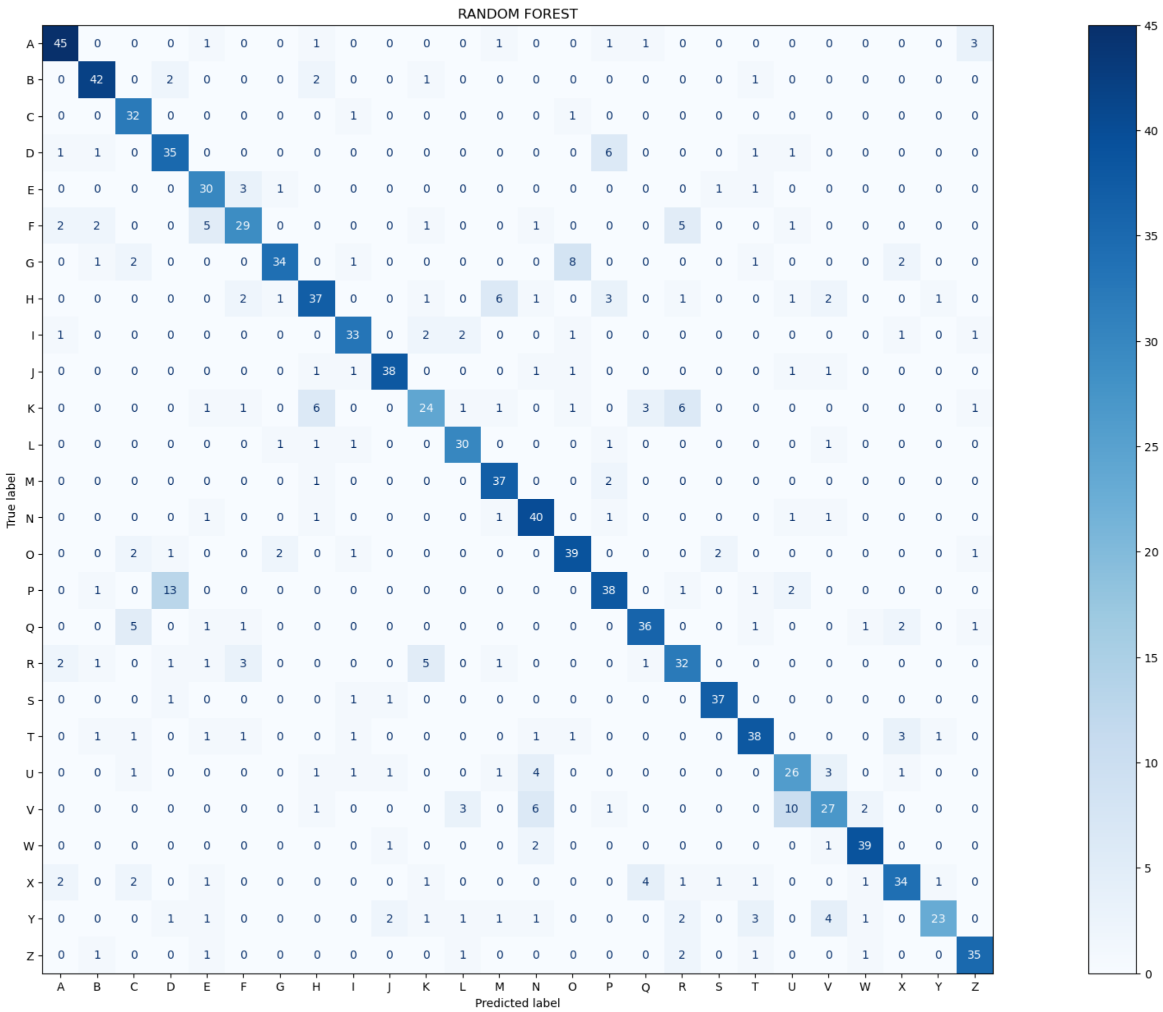
U nazivu RF algoritma bitan je i pojam nasumičnosti (engl. random). Naime, on dolazi iz nasumičnog odabira značajki prilikom donošenja odluka. Svako stablo odluke traži najbolju značajku pomoću koje će ulazni skup podataka razdvojiti na dva podskupa. RF za svako stablo odabire skup nasumičnih značajki, a zatim stablo odabire najbolju među njima. Ovime se doprinosi većoj raznolikosti stabala, a što je korelacija među stablima manja to je generalizacija bolja i klasifikacija preciznija.

U radu je korišten RF algoritam dostupan u sklopu *scikit-learn* knjižnice. *Scikit-learn* je besplatna knjižnica za programski jezik Python koja sadrži brojne algoritme klasifikacije, regresije i grupiranja. Također je dizajnirana za usklađen rad s *NumPy* i *SciPy* knjižnicama.

Kako je ranije navedeno, RF model je postigao točnost klasifikacije od 76% na validacijskom skupu, a na Slici 4.4. prikazana je matrica zabune. Iz nje se može iščitati da se najviše zabuna javlja između slova D i P (13) te između slova U i V (10). Sve ostale zabune se javljaju manje od deset puta. Slovo A ima najviše točno predviđenih primjeraka (45). Kada se u obzir uzme četiri slova s najvećom vjerojatnošću, tada preciznost klasifikacije na validacijskom skupu raste na 94%. Ako se rezultat raščlani po indeksima na kojima se nalazi točno slovo, dobivaju se sljedeći postotci prisutnosti točnog slova:

1. 76%
2. 12%
3. 4%
4. 2%

Istrenirani model je pohranjen pomoću *pickle* knjižnice kao *.pickle* tip datoteke.



Slika 4.4. Matrica zabune modela RF

4.2. Integracija RF modela u Android IME

Nakon odabira i treniranja modela za klasifikaciju, istražene su mogućnosti integracije *scikit-learn* modela u Android IME. Radi prethodnog znanja o integraciji *TensorFlow* modela s Android aplikacijama, problemu se pokušalo pristupiti izradom *TensorFlow Lite* RF modela. Nažalost, za vrijeme pisanja rada *TensorFlow Lite* još nije imao razvijenu podršku za RF.

Daljnjim istraživanjem pronađeni su Open Neural Network Exchange (ONNX) i Predictive Model Markup Language (PMML). ONNX je ekosustav otvorenog koda razvijen od strane velikih tehnoloških kompanija i istraživačkih organizacija sa ciljem lakše integracije modela strojnog učenja neovisno o arhitekturi sustava [18]. PMML je također format koji omogućava dijeljenje modela između različitih aplikacija i sustava. Koristi XML (Extensible Markup Language) format pomoću kojega opisuje modele, uključujući ulazne podatke, transformacije potrebne za pripremu podataka kao i parametre koji definiraju same modele [19]. Tijekom rada s navedenim metodama javljale su se poteškoće prilikom instalacije paketa, kao i tijekom pokušaja njihove implementacije. U konačnici su ONNX i PMML metode odbačene. Problem integracije *scikit-learn* modela s Android aplikacijom je na kraju riješen koristeći Chaquopy.

4.2.1. Chaquopy

Chaquopy je komplet za razvoj softvera (engl. Software Development Kit, SDK) koji omogućava korištenje programskog jezika Python u Android aplikacijama [21]. Pomoću jednostavnih sučelja za programiranje aplikacija (engl. Application Programming Interface, API) poziva se Python kôd iz Jave/Kotlina i obratno. Chaquopy također pruža podršku i za brojne Python knjižnice poput *SciPy*, *OpenCV*, *TensorFlow*, *scikit-learn* te mnoge druge. Chaquopy je razvio Malcolm Smith, a zahvaljujući podršci Anaconde, u srpnju 2022. godine SDK je postao potpuno besplatan i otvorenog koda [22]. Izvorni kod je u potpunosti dostupan putem GitHub repozitorija [23] te je zaštićen MIT licencom. Službena Chaquopy dokumentacija ima detaljno opisan proces integracije s Android aplikacijom, instaliranje dodatnih knjižnica, čitanje datoteka te još mnoštvo korisnih informacija.

Za početak je na mobilni uređaj prebačen RF model istreniran na računalu. Python skripta, ranije korištena na računalu za učitavanje spremljenog modela i klasifikaciju podataka iz JSON datoteka, kopirana je u izvorni kôd Android aplikacije. Zatim je skripta malo izmijenjena kako bi bila prilagođena klasifikaciji jednog slova i to za izravno proslijeđene vrijednosti koje je izmjerio magnetometar. Android IME podatke ne pohranjuju u lokalnu bazu niti u JSON datoteke. Međutim, prilikom učitavanja RF modela iz *.pickle* datoteke dolazilo je do pogreške u tipu podataka. Problem je proizlazio iz razlike u arhitekturi uređaja (računala i pametnog telefona). Pomoću Anaconde je izrađeno 32-bitno virtualno okruženje (engl. virtual environment) sa istom inačicom Pythona i *scikit-learn* knjižnice. RF model je ponovno treniran u novom virtualnom okruženju, ali pogreška u tipu podataka se i dalje javljala. Model je zatim treniran i na 32-bitnom Raspberry Pi uređaju. Napravljena je potpuno nova instalacija operacijskog sustava, kako prethodno instalirane knjižnice ne bi dovele do nekompatibilnosti, ali problem se javio prilikom instaliranja *scikit-learn* knjižnice. Naime, najnovija stabilna verzija *scikit-learn* knjižnice u vrijeme pisanja ovoga rada je 1.2.1, dok je najnovija verzija koju podržava Chaquopy 0.24.1.

S obzirom na to da se treniranje RF modela odvija vrlo brzo, slijedio je pokušaj implementacije treniranja modela na samom mobilnom uređaju. Ideja je da trening bude dio inicijalnog postavljanja Android IME-a. Dakle, nakon instalacije aplikacije korisnik bi pokrenuo treniranje modela za klasifikaciju, model bi se zatim spremio na uređaj u mapu aplikacije i u svakom trenutku bio dostupan za korištenje. JSON datoteke u kojima je pohranjen skup podataka za treniranje modela dodane su u *assets* mapu Android projekta, a one se prilikom pokretanja treninga kopiraju u mapu aplikacije te se put (engl. path) do navedene mape prosljeđuje Python skripti. Trening se na mobilnom uređaju izvršava vrlo brzo, rezultat preciznosti jednak je kao i na računalu te je model uspješno spremljen na uređaj. Prilikom korištenja Android IME-a model se uspješno učita i vraća predviđene klase za magnetom napisano slovo. Preciznost klasificiranja je vrlo dobra, a napisano slovo se u većini slučajeva nalazi u četiri ponuđena kojima je model dodijelio najveću vjerojatnost.

Međutim, uočeno je da kada se Android IME tek pokrene i završi se pisanje prvog slova, dođe do kraćeg zastoja aplikacije prilikom čekanja na rezultat klasifikacije. Kod svih sljedećih slova rezultat se dobiva puno brže. Do zastoja je dolazilo tijekom prvog učitavanja skripte za predikciju slova. Naime, kako bi se Python skripta mogla koristiti, prvo je potrebno pokrenuti Python i instancu spremiti u varijablu tipa *Python*, a potom u varijablu tipa *PyObject* treba učitati skriptu (Ispis 4.1.). Nakon što je to napravljeno, putem *PyObject* varijable u kojoj je pohranjena skripta, mogu se pozivati funkcije iz skripte i slati im argumente. Stoga je u Android IME dodana tipka koja služi za učitavanje skripte kako ne bi dolazilo do zastoja prilikom unosa prvog slova. Skriptu je potrebno učitati samo kada se u postavkama mobilnog uređaja promijeni tipkovnica, a prije nego se započne s pisanjem.

Ispis 4.1. Kôd za pokretanje Pythona i učitavanje skripte

```
private void startScript() {
      if (!Python.isStarted()) {
            Python.start(new
            AndroidPlatform(this.getApplicationContext()));
      }
      py = Python.getInstance();
      pyModule = py.getModule("Predict");

      String dirPath = getExternalFilesDir(null).getAbsolutePath();
      PyObject object = pyModule.callAttr("start", dirPath);

      Toast.makeText(this, object.toString(),
      Toast.LENGTH_SHORT).show();
}
```

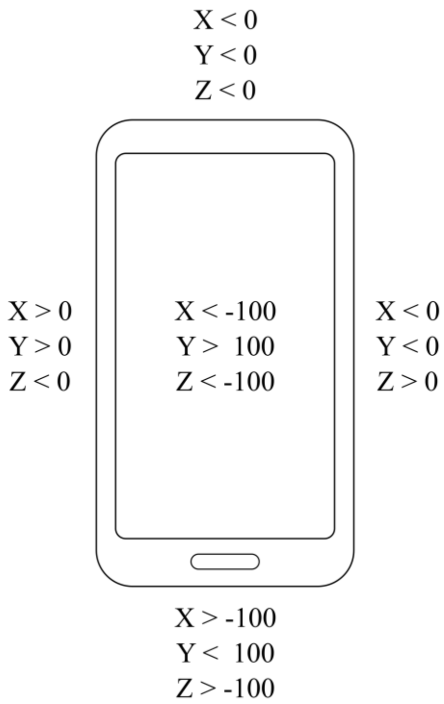
5. INTERAKCIJA

U ovom poglavlju opisuje se dizajn modaliteta interakcije, dizajn korisničkog sučelja za Android IME, te se demonstrira korištenje implementiranog rješenja.

5.1. Modalitet interakcije

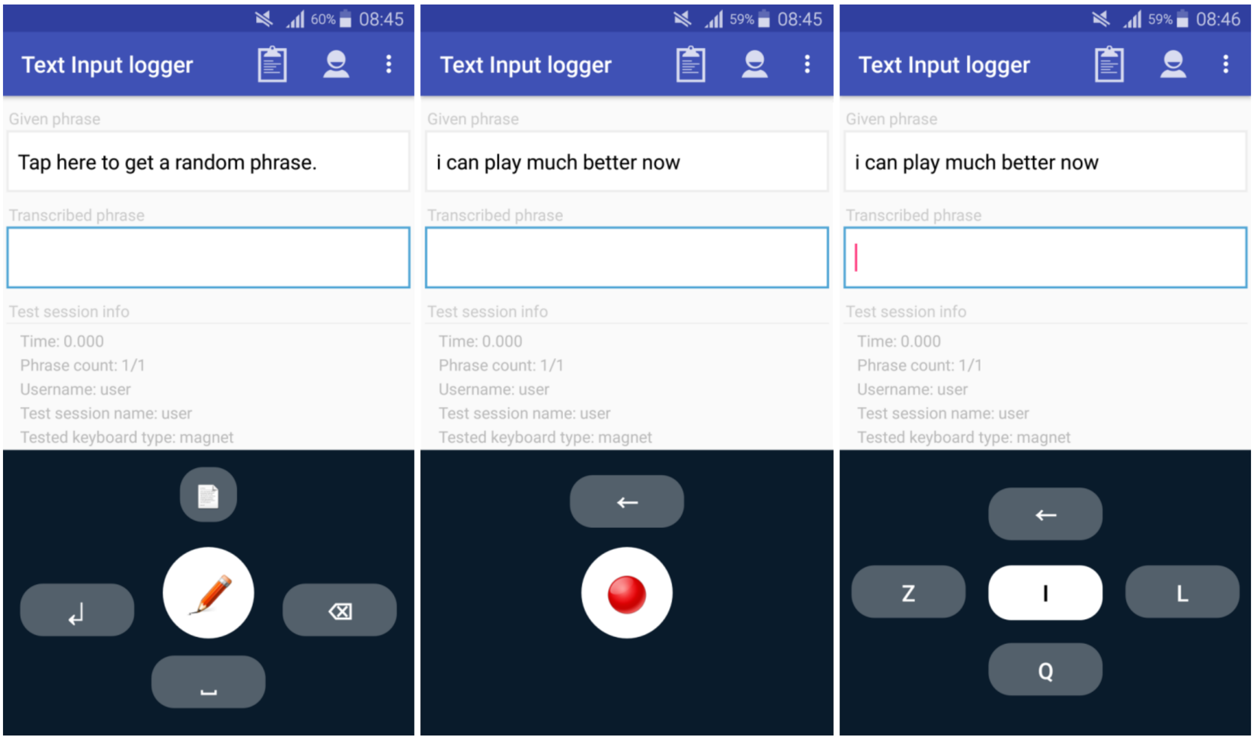
Interakcija s Android IME-om se u početku odvijala pomoću pritiska tipki na zaslonu mobilnog uređaja te utjecajem permanentnog magneta na magnetski senzor. Pomoću tipki se učitavala skripta za klasifikaciju, pokretalo i zaustavljalo snimanje vrijednosti magnetskog polja te odabirao razmak, brisanje, potvrda, povratak i slovo koje se želi upisati. S obzirom da je dio interakcije pomoću magneta beskontaktan, odlučeno je da kompletna interkacija s aplikacijom bude u potpunosti beskontaktna. Konačna interakcija se odvija na način da se u jednoj ruci drži permanentni magnet kojim se utječe na magnetometar uređaja, odnosno mijenja vrijednosti magnetskog polja oko uređaja, a drugom rukom se aktivira senzor blizine.

Tipke koje se ranije pritiskalo na zaslonu osjetljivom na dodir implementirane su na način da se, ovisno o poziciji magneta u odnosu na mobilni uređaj, po tipkama pomiče "pokazivač", a odabir ciljane tipke se vrši senzorom na blizinu. Trenutno odabrana tipka označena je svjetlijom pozadinom. Za odrediti poziciju magneta u odnosu na uređaj promatraju se predznaci i iznosi X, Y i Z vrijednosti kako je prikazano na Slici 5.1. Sva četiri ruba uređaja imaju drugačije predznake, dok centar uređaja i donji rub imaju jednake predznake, ali se razlikuju po iznosima navedenih triju vrijednosti. Kod odabira tipke u centru uređaja magnet je potrebno malo približiti uređaju kako bi se postigle željene X, Y i Z vrijednosti. Nakon što je "pokazivač" postavljen na željenu tipku, aktiviranjem senzora blizine potvrđuje se odabir. Za aktiviranje senzora je potrebno da očitanje bude vrijednost manja od jedan, to jest potrebno je ruku približiti na manje od jednog centimetra do zaslona uređaja. Za interakciju je odabran senzor blizine, a ne senzor svjetla, jer je tijekom pisanja ruka često pozicionirana iznad senzora i blokira svjetlo, što može dovesti do pogrešnih očitanja.



Slika 5.1. Vrijednosti magnetskog polja koje se koriste u postupku pomicanja pokazivača

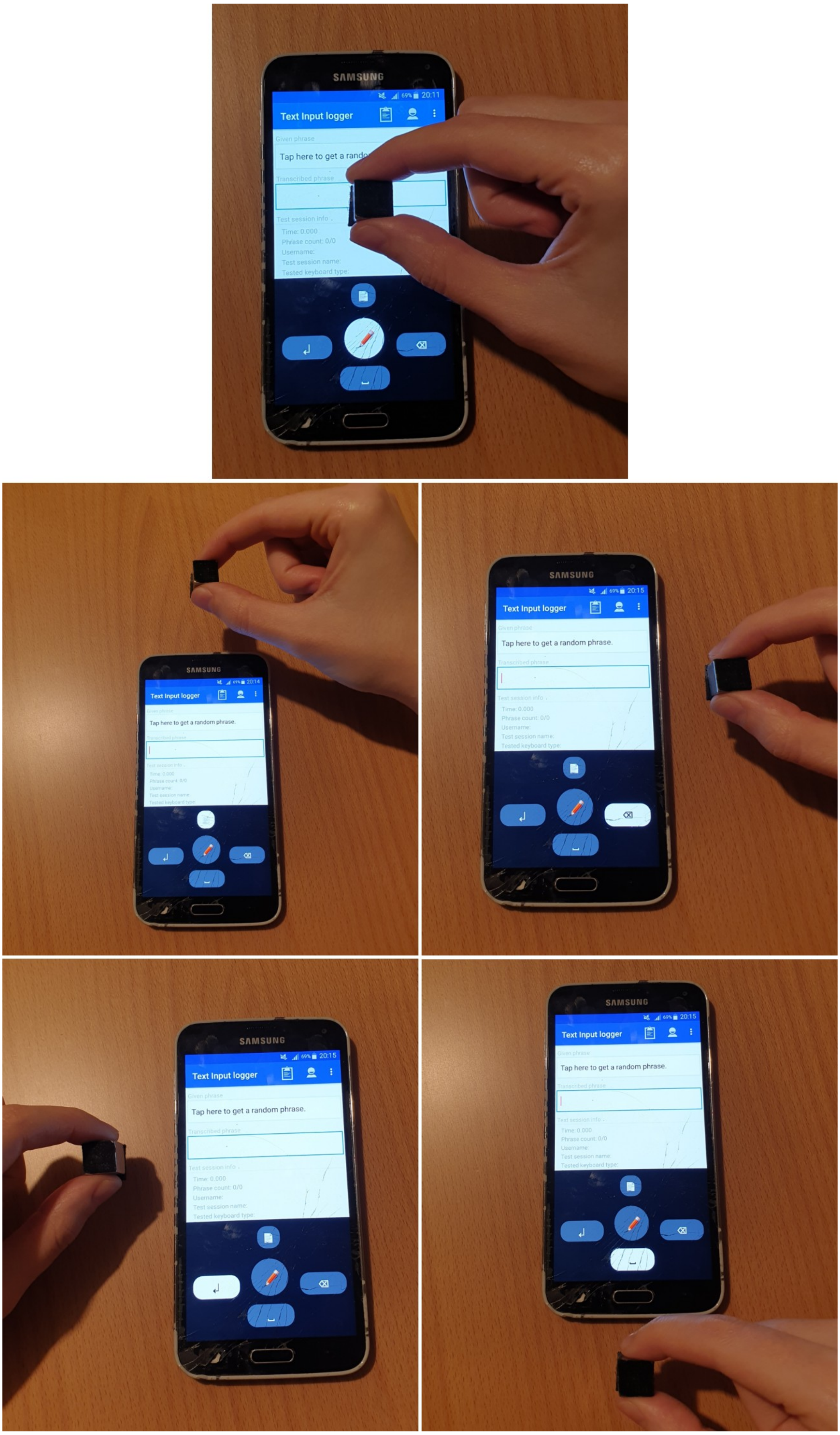
Korisničko sučelje Android IME-a se sastoji od tri dijela prikazanih na Slici 5.2. Prilikom pokretanja vidljiv je raspored (engl. layout) sa pet tipki. Gornja tipka sadrži ikonu lista papira koja predstavlja skriptu, a služi za učitavanje Python skripte za klasificiranje slova. Slijeva se nalazi potvrda unosa, zdesna tipka za brisanje, a na dnu razmak. U centru, označena ikonom olovke, je tipka kojom se prelazi na dio za pisanje. U drugom dijelu se sa gornje strane nalazi tipka za povratak, a u centru tipka za pokretanje snimanja označena crvenim krugom. Prije početka pisanja slova potrebno je magnet postaviti na poziciju s koje će se početi pisati slovo. Ukoliko bi se snimanje pokrenulo prije pomicanja magneta na početnu poziciju, tada bi se pomak iz centra uređaja prema početnoj poziciji računao kao dio slova, što bi moglo dovesti do pogrešne klasifikacije. Nakon pokretanja snimanja, tipka za povratak nestaje, a tipka za snimanje poprima oznaku crnog kvadrata te sada ima ulogu zaustavljanja snimanja. Odabrane su navedene oznake i boje za pokretanje i zaustavljanje snimanja jer se u takvom obliku često pojavljuju u raznim programskim rješenjima. Tijekom snimanja, bez obzira kako se magnet pomakne u odnosu na uređaj, pokazivač će uvijek stajati na tipci za zaustavljanje snimanja. Nakon što se snimanje prekine, Python skripta će učitati RF model za klasifikaciju te mu predati snimljene vrijednosti. Model će zatim odraditi proces klasifikacije i vratiti četiri klase, tj. slova, s najvećom vjerojatnošću. Android IME zatim prelazi na treći raspored koji vizualizira najvjerojatnija slova u zasebnim tipkama. Slovo s najvećom vjerojatnošću se upisuje u srednju tipku, a zatim, kako se vjerojatnost smanjuje, upisuje ih se u desnu, pa donju i na kraju lijevu tipku. Za slučaj da model nije uspješno odradio klasifikaciju, gornja tipka nudi mogućnost povratka na ponovno pisanje slova. Ukoliko je željeno slovo ponuđeno, tada se „pokazivač“ magnetom pomakne na odgovarajuću tipku i odabir potvrdi senzorom blizine. Nakon što je slovo upisano, sučelje IME-a se vraća na dio za pisanje. Time je povećana brzina unosa teksta jer nije potrebno svaki put ulaziti u način za pisanje, a tipke na početnom rasporedu (primjerice razmak) se u zadanom tekstu javljaju rjeđe od slova.



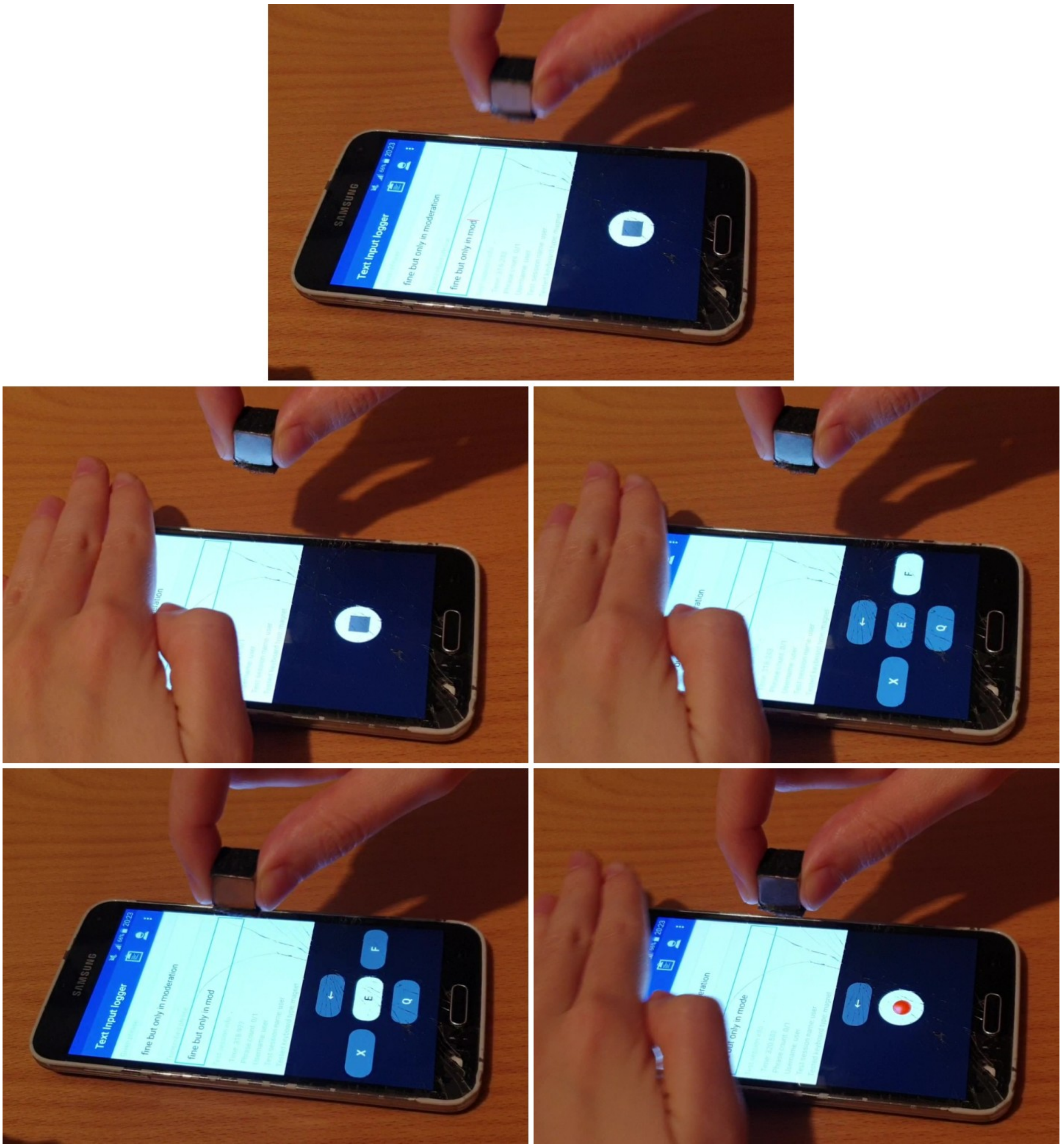
Slika 5.2. Korisničko sučelje Android IME-a vizualizirano tijekom rada u testnoj aplikaciji

5.2. Demonstracija interakcije

Na Slici 5.3. prikazan je način pomicanja virtualnog pokazivača pomoću permanentnog magneta, a na Slici 5.4. postupak unosa slova E u testnoj aplikaciji tijekom pisanja fraze.



Slika 5.3. Pomicanje virtualnog pokazivača permanentnim magnetom



Slika 5.4. Primjer unosa slova E u testnoj aplikaciji

6. HCI EKSPERIMENT

Nakon uspješno izrađene metode za unos teksta i integracije modela za klasifikaciju, pripremljeno je sve potrebno za provođenje odgovarajućeg HCI (engl. Human-Computer Interaction) eksperimenta. Eksperimentom se željelo utvrditi koliko kvalitetno aplikacija prepoznaje slova te koliko će korisnici pomoću nje biti brzi i precizni pri unosu teksta.

6.1. Korisnici

U eksperimentu je sudjelovalo osam korisnika. Među njima su tri žene (37,5%) i pet muškaraca (62,5%). Najmlađi korisnik ima 25 godina, najstariji 60, a medijan iznosi 33 godine. Svim korisnicima je dominantna desna ruka.

6.2. Oprema

Svi korisnici su metodu unosa teksta testirali na istom Samsung Galaxy S5 uređaju koji je također korišten i tijekom razvoja aplikacije. Broj modela uređaja je SM-G900F, visina uređaja iznosi 14.2, a širina 7.25 centimetra. Dimenzije zaslona su 11.5 x 6.5 centimetara. Mobilni uređaj ima unaprijed ugrađen magnetometar i senzor blizine. Magnetometar je model AK09911C Magnetic field Sensor proizvođača Asahi Kasei Microdevices, a senzor blizine je model TMG399X Proximity Sensor koji proizvodi AMS, Inc.

Korisnici su tekst unosili putem Android IME-a izrađenog u sklopu diplomskog rada. Interakcija je postignuta utjecanjem na magnetometar, tj. magnetsko polje oko uređaja čije vrijednosti magnetometar mjeri. Promjenom ambijentalnog magnetskog polja utjecalo se pomoću permanentnog magneta u obliku kocke koji je korišten i tijekom razvoja rješenja. Magnet je uvijek potrebno držati u istoj orijentaciji radi smjera kretanja magnetnih silnica.

Brzinu unosa teksta se izražava brojem riječi u minuti (engl. Words Per Minute, WPM), a točnost ukupnom stopom greške (engl. Total Error Rate, TER). Za mjerenje navedenih metrika korišten je Text Input Logger, aplikacija pomoću koje se provode eksperimenti u domeni unosa teksta. Aplikacija korisniku omogućava odabir rečenice koju će prepisati, a tijekom procesa pisanja mjeri i računa potrebne metrike. U postavkama aplikacije mijenjano je samo korisničko ime, te ime sesije i broj rečenica, ovisno odrađuje li korisnik u datom trenutku treniranje ili testiranje.

6.3. Procedura

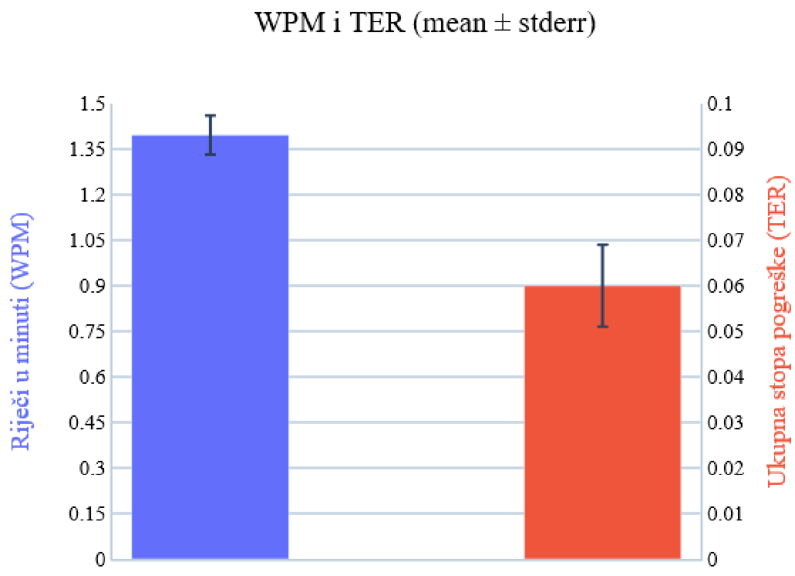
Svakom korisniku je prvo objašnjeno kako se koristi metoda unosa teksta, ukazano je na važnost orijentacije magneta te je napravljena kratka demonstracija. Zatim im je ponuđeno da pokušaju napisati pet rečenica za trening. Cilj treninga je bio uvježbavanje koordinacije rada s dvije ruke, dobivanje osjećaja za visinu na kojoj je najučinkovitije držati magnet, ali i općenito navikavanje na dostupni modalitet interakcije.

Nakon treninga magnetni senzor je ponovno kalibriran, a zatim je slijedilo pravo testiranje u kojem je korisnik trebao prepisati deset rečenica. Tijekom trajanja eksperimenta korisnicima je dozvoljeno uzeti kraći odmor. Također, ako bi tijekom pisanja teksta došlo do poremećaja u radu senzora, pri završetku rečenice vršila se dodatna kalibracija magnetometra. Poremećaj u radu senzora najlakše je vidljiv po neusklađenom pomicanju pokazivača u odnosu na poziciju magneta, ali i po većim greškama modela prilikom klasifikacije slova.

Kada je korisnik završio s unosom svih deset rečenica slijedilo je ispunjavanje upitnika TLX kojim se ocjenjuje radno opterećenje interakcije. Na ljestvici od 1 do 21 treba ocijeniti koliko je unos teksta bio mentalno zahtjevan, fizički zahtjevan, kolika je bila frustracija tijekom rada s metodom unosa teksta, procijeniti vlastite performanse te ocijeniti ukupan napor potreban za izvršavanje zadataka pisanja.

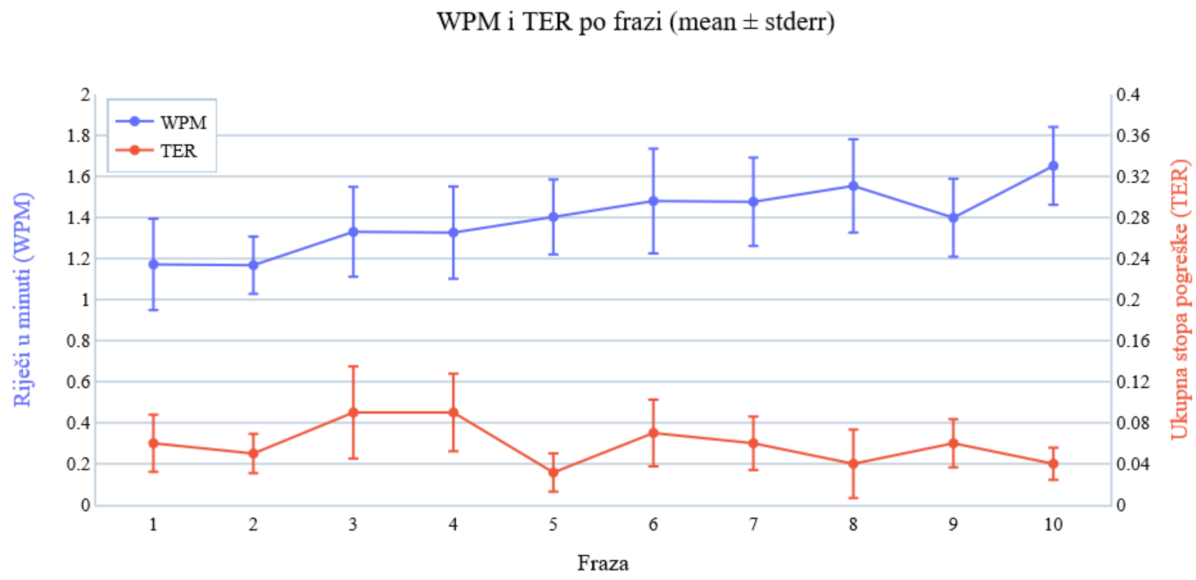
6.4. Rezultati

Na Slici 6.1. prikazani su dobiveni rezultati (metrike WPM i TER) koji direktno određuju učinkovitost implementirane metode za beskontaktni unos teksta. Brzina unosa teksta iznosi 1.396 ± 0.064 WPM, dok je ukupna stopa pogreške 6 ± 0.9 %.



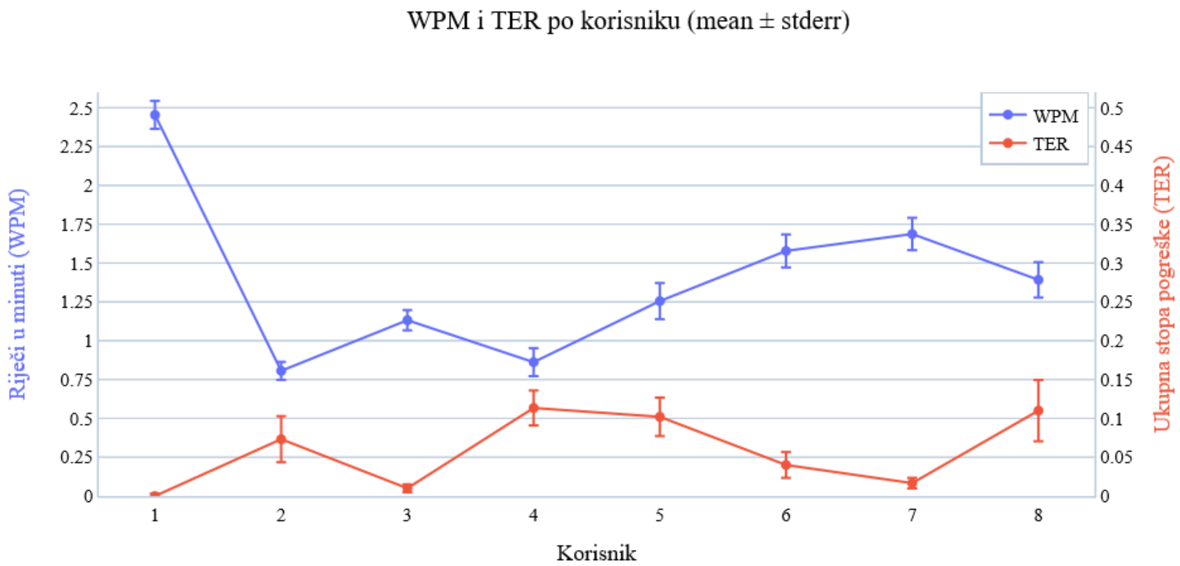
Slika 6.1. Metrike WPM i TER za implementiranu metodu unosa teksta

Rezultati postignuti po pojedinačnim frazama prikazani su na Slici 6.2. Vidljivo je da brzina unosa teksta prati blago uzlazni trend. Najbolji rezultat je postignut u desetoj rečenici kod koje brzina unosa teksta iznosi 1.651 WPM, dok je rezultat najslabiji u drugoj rečenici i iznosi 1.168 WPM. Najveća varijacija u iznosu od ± 0.255 WPM postignuta je u šestoj rečenici. Za razliku od brzine unosa, ukupna stopa pogreške ne pokazuje primjetan trend promjene. Najniža ukupna stopa pogreške postignuta je u petoj rečenici i iznosi 3.15%, dok je najviša (i jednaka) stopa pogreške od 9% u trećoj i četvrtoj rečenici.



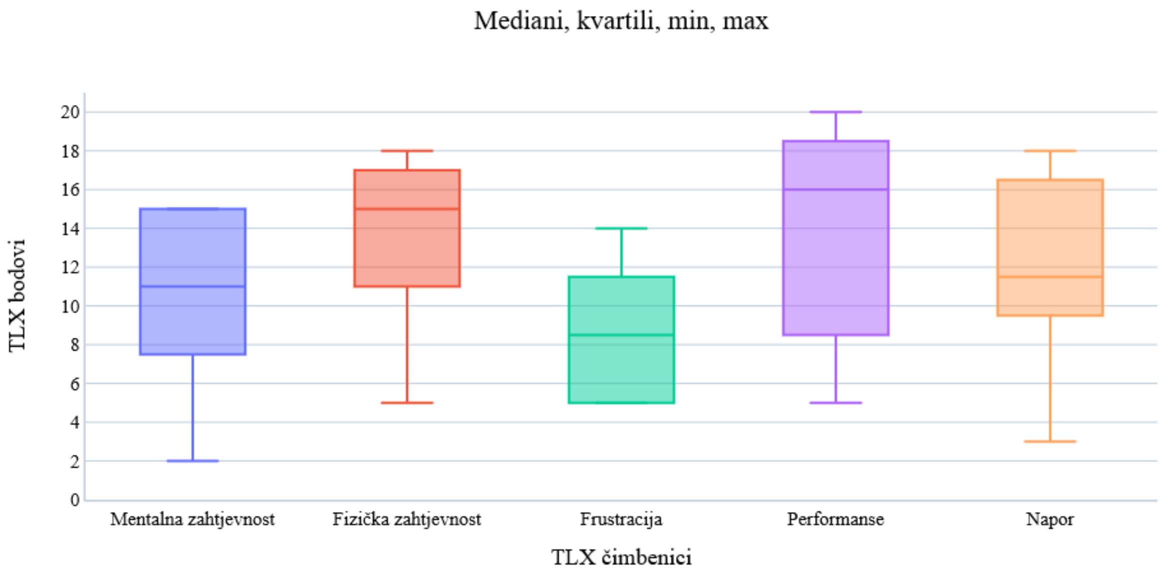
Slika 6.2. Metrike WPM i TER po pojedinačnim frazama

Na Slici 6.3. prikazani su rezultati unosa teksta po pojedinom korisniku. Najveća brzina unosa teksta pojedinca iznosi 2.454 WPM, dok je najniža brzina unosa 0.806 WPM. Najniža ukupna stopa pogreške pojedinca iznosi 0%, a najviša stopa pogreške je 11.3%.



Slika 6.3. Metrike WPM i TER po pojedinom korisniku

Rezultati TLX upitnika prikazani su na Slici 6.4. Kod mentalne i fizičke zahtjevnosti, frustracije i napora manja vrijednost ukazuje na bolji rezultat, dok za performanse manja vrijednost ukazuje na lošiji rezultat. Primjetno je da rezultati dosta variraju, osobito kod procjene vlastitih performansi i uloženog napora.



Slika 6.4. Rezultati TLX upitnika

7. ZAKLJUČAK

U radu je predstavljena implementacija metode unosa teksta beskontaktnim putem, utjecanjem na magnetsko polje u okolini uređaja pomoću permanentnog magneta. Opisani su glavni podaci i predstavljena je aplikacija razvijena za njihovo prikupljanje. Pretpostavku da će se postupkom kalibracije dobivati jednaki rezultati, neovisno o lokaciji na kojoj se magnetom utječe na magnetno polje, dokazala se pomoću eksperimenta u kojem su različita slova pisana na dvije lokacije. Koristeći t-SNE redukciju dimenzionalnosti i mapiranje dobivenih 2D vrijednosti, a zatim i izradu prosječne 2D slike svakog slova, donesen je zaključak da se slova međusobno bolje razlikuju kada su napisana iznad mobilnog uređaja.

Tijekom traženja najboljeg modela za klasifikaciju podataka, proučavane su neuronske mreže za rad s 3D podacima i podacima u obliku oblaka točaka. Ipak, učinkovitijim su se pokazali jednostavniji i često korišteni *scikit-learn* modeli. Među nekoliko odabranih modela najboljim se pokazao RF kojim je postignuta preciznost klasifikacije od 76%, a zatim je dodatno povećana na 94% uzimajući u obzir četiri slova s najvećom vjerojatnošću.

Kod integriranja istreniranog RF modela u Android IME javljale su se razne poteškoće, koje su razriješene omogućavanjem treniranja na samom mobilnom uređaju. Za pokretanje Python skripti na Android uređaju korišten je Chaquopy. U konačnici je razvijena u potpunosti beskontaktna metoda za unos teksta koja omogućava prepoznavanje gesti i pomicanje pomoćnog virtualnog pokazivača.

Za utvrđivanje učinkovitosti unosa teksta pomoću izrađenog Android IME-a, proveden je HCI eksperiment. Aplikaciju je testiralo osam korisnika od kojih je svaki dobio zadatak prepisati deset rečenica, te po završetku testiranja ispuniti TLX upitnik. Brzina unosa teksta je niska, što je i očekivano. Iako su u rečenicama prisutne pogreške, tijekom testiranja je primijećeno da one uglavnom nastaju kada korisnici slučajno senzor blizine aktiviraju dva puta. Za slučaj pogrešne klasifikacije postoji tipka za povratak na pisanje, ali klasifikacija je u većini slučajeva bila vrlo precizna. Klasifikaciju bi vjerojatno povećao točno zadani način pisanja svakog slova, ali ideja je bila istražiti kolika preciznost se može postići kada je korisnicima dana potpuna fleksibilnost prilikom izvođenja gesti. Drugim riječima, svaki je korisnik mogao napisati slovo na različit način, pri čemu odgovarajuća gesta mora „konstruirati“ ispravni grafem (oblik slova).

8. LITERATURA

[1] Darbar R.; Samanta D.: „MagiText : Around Device Magnetic Interaction for 3D Space Text Entry in Smartphone“, IEEE International Conference on Electronics, Computing and Communication Technologies (CONECCT), 2015.

[2] Ketabdar H.; Roshandel M.; Yüksel K.A.: „MagiWrite: Towards Touchless Digit Entry Using 3D Space Around Mobile Devices“, MobileHCI '10: Proceedings of the 12th international conference on Human computer interaction with mobile devices and services, pp. 443–446, 2010.

[3] Ketabdar, H.; Haji-Abolhassani A.; Roshandel M.: „MagiThings: Gestural Interaction with Mobile Devices Based on Using Embedded Compass (Magnetic Field) Sensor“, International Journal of Mobile Human Computer Interaction, Vol. 5, No. 3, pp. 23–41, 2013.

[4] Ketabdar, H.; Yüksel K.A.; Roshandel M.: „MagiTact: Interaction with Mobile Devices Based on Compass (Magnetic) Sensor“, IUI '10: Proceedings of the 15th international conference on Intelligent user interfaces, pp. 413–414, 2010.

[5] Ketabdar, H.; Yüksel K.A.; Jahnbekam A.; Roshandel M.; Skripko D.: “MagiSign: User Identification/Authentication”, UBICOMM 2010 : The Fourth International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, pp. 31-34, 2010.

[6] Siio I.; Kadomura A.: “MagNail: User Interaction with Smart Device through Magnet Attached to Fingernail”, UbiComp/ISWC'15 Adjunct: Adjunct Proceedings of the 2015 ACM International Joint Conference on Pervasive and Ubiquitous Computing and Proceedings of the 2015 ACM International Symposium on Wearable Computers, pp. 309–312, 2015.

[7] “Independent component analysis”, s Interneta, https://en.wikipedia.org/wiki/Independent_component_analysis, 13. veljače 2023.

[8] “Principal component analysis”, s Interneta, https://en.wikipedia.org/wiki/Principal_component_analysis, 13. veljače 2023.

[9] “t-distributed stochastic neighbor embedding”, s Interneta, https://en.wikipedia.org/wiki/T-distributed_stochastic_neighbor_embedding, 14. veljače 2023.

[10] “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”, s Interneta, https://github.com/charlesq34/pointnet, 18. veljače 2023.

[11] Qi C.R.; Su H.; Mo K.; Guibas L.J.: “PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation”, s Interneta, https://stanford.edu/~rqi/pointnet/, 18. veljače 2023.

[12] “Voxel”, s Interneta, https://en.wikipedia.org/wiki/Voxel, 18. veljače 2023.

[13] Chan M.: “Step by Step Implementation: 3D Convolutional Neural Network in Keras”, s Interneta, https://towardsdatascience.com/step-by-step-implementation-3d-convolutional-neural-network-in-keras-12efbdd7b130, 22. veljače 2023.

[14] “Using a 3D Convolutional Neural Network on medical imaging data (CT Scans) for Kaggle”, s Interneta, https://pythonprogramming.net/3d-convolutional-neural-network-machine-learning-tutorial/, 22. veljače 2023.

[15] “What is random forest?”, s Interneta, https://www.ibm.com/topics/random-forest, 24. veljače 2023.

[16] “Random Forest”, s Interneta, https://en.wikipedia.org/wiki/Random_forest, 24. veljače 2023.

[17] “Decision Trees”, s Interneta, https://www.ibm.com/topics/decision-trees, 24. veljače 2023.

[18] “ONNX” , s Interneta, https://github.com/onnx/onnx, 28. veljače 2023.

[19] “Predictive Model Markup Language”, s Interneta, https://en.wikipedia.org/wiki/Predictive_Model_Markup_Language, 28. veljače 2023.

[20] “SkLearn2PMML”, s Interneta, https://github.com/jpmml/sklearn2pmml, 28. veljače 2023.

[21] “Chaquopy”, s Interneta, https://chaquo.com/chaquopy/, 1. ožujka 2023.

[22] “Chaquopy is now open-source”, s Interneta, https://chaquo.com/chaquopy/chaquopy-is-now-open-source/, 1. ožujka 2023.

[23] “Chaquopy: the Python SDK for Android”, s Interneta, https://github.com/chaquo/chaquopy, 1. ožujka 2023.

9. SAŽETAK

U ovome radu istražena je mogućnost beskontaktnog upravljanja mobilnim uređajem manipulacijom ambijentalnog magnetskog polja. Implementirana je metoda unosa teksta koja se zasniva na prepoznavanju gesti koje korisnik ostvaruje permanentnim magnetom iznad zaslona mobilnog uređaja. Implementirano rješenje koristi magnetometar i senzor blizine, a problem klasifikacije slova s obzirom na odgovarajući otisak magnetskog polja riješen je modelom nasumične šume (RF). Opisani su postupci testiranja koji su rezultirali argumentiranim odlukama vezanima za dizajn modaliteta interakcije i za odabir klasifikacijskog modela. Realizirana metoda unosa teksta vrednovana je provedbom HCI eksperimenta s osam ispitnih korisnika, s ciljem izlučivanja metrika učinkovitosti i radnog opterećenja interakcije.

Ključne riječi — interakcija čovjeka i računala, interakcija oko uređaja, unos teksta, magnetsko polje, klasifikacija, redukcija dimenzionalnosti, model nasumične šume

10. ABSTRACT

In this thesis, the possibility of contactless control of a mobile device by manipulating the surrounding magnetic field is investigated. A text entry method has been implemented based on the recognition of gestures performed by the user with a permanent magnet over the mobile device screen. The implemented solution uses a magnetometer and a proximity sensor, and the problem of letter classification with respect to the corresponding magnetic field fingerprint is solved by a Random Forest (RF) model. Test procedures are described that led to reasoned decisions regarding the design of the interaction modality and the selection of the classification model. The implemented text entry method was evaluated by conducting an HCI experiment with eight participants, with the goal of extracting metrics about efficiency and interaction workload.

Keywords — Human-Computer Interaction (HCI), Around-Device Interaction (ADI), text entry, magnetic field, classification, dimensionality reduction, Random Forest model

11. POPIS SLIKA

12. POPIS TABLICA

PRILOG A: Spremanje podataka u JSON datoteku

```
private void saveJSON(String JSON_NAME, List<Letter> list) throws
Exception {
      File dir = getExternalFilesDir(null);
      File file = new File(dir, JSON_NAME);

      OutputStream fos = new FileOutputStream(file);
      JsonWriter writer = new JsonWriter(new OutputStreamWriter(fos,
      StandardCharsets.UTF_8));

      writer.setIndent(„  “);
      writer.beginArray();

      for (int i = 0; i < list.size(); i++) {
            Point pointList = list.get(i).getPointList();

            writer.beginObject();
            writer.name("letter").value(list.get(i).getLetter());
            writer.name("user").value(list.get(i).getUser());
            writer.name("calibration");
            writer.beginObject();
            writer.name("deltaX").value(list.get(i).getDeltaX());
            writer.name("deltaY").value(list.get(i).getDeltaY());
            writer.name("deltaZ").value(list.get(i).getDeltaZ());
            writer.endObject();
            writer.name("t0").value(list.get(i).getT0());
            writer.name("tN").value(list.get(i).getTN());
            writer.name("t").value(list.get(i).getT());
            writer.name("fingerprint");

            writer.beginArray();
            for (int n = 0; n < pointList.point.get(0).size(); n++) {
                  writer.beginArray();
                  for (int m = 0; m < 3; m++) {
                        writer.value(pointList.point.get(m).get(n));
                  }
                  writer.endArray();
            }
            writer.endArray();

            writer.endObject();
      }

      writer.endArray();

      writer.close();
      fos.close();
}
```

PRILOG B: Dodavanje i brisanje točaka u postupku redukcije dimenzionalnosti

```
for d in data:
    letter_points = d[3]
    difference = letter_points - average_points

    if difference > 0:
        while difference > 0:
            random_index = random.randint(0, d[3]-1)

            d[4].pop(random_index)
            d[3] -= 1

            difference -= 1

    elif difference < 0:
        while difference < 0:
            before_index = random.randint(0, d[3]-2)
            after_index = before_index + 1

            new_x = (d[4][before_index][0] + d[4][after_index][0]) / 2
            new_y = (d[4][before_index][1] + d[4][after_index][1]) / 2
            new_z = (d[4][before_index][2] + d[4][after_index][2]) / 2

            new_point = [new_x, new_y, new_z]

            d[4].insert(after_index, new_point)
            d[3] += 1

            difference += 1
```