RAČUNALOM PODRŽANA MJERENJA

Izrada sučelja mozak računalo

Jan Strelec

Luka Pavlović

Mihael Bedi

Sven Čepić

Sadržaj:

[Sažetak 3](#_Toc190538251)

[1. Generiranje stimulacija 4](#_Toc190538252)

[1.1 Postavljanje grafičkog sučelja 4](#_Toc190538253)

[1.2 Generiranje nasumičnih slova 5](#_Toc190538254)

[1.3 Automatska izmjena slova u intervalima 6](#_Toc190538255)

[1.4 Pohrana podataka o prikazanim slovima (Markeri) 6](#_Toc190538256)

[2 Snimanje i generiranje EEG signala 7](#_Toc190538257)

[2.1 Generiranje signala 7](#_Toc190538258)

[2.2 Snimanje signala 8](#_Toc190538259)

[3 Obrada EEG podataka 9](#_Toc190538260)

[3.1 Usklađivanje slova i EEG vremenskih oznaka 9](#_Toc190538261)

[4 Klasifikacija EEG podataka 10](#_Toc190538263)

[4.1 Priprema podataka za rad s MNE 10](#_Toc190538264)

[4.2 Filtriranje signala 11](#_Toc190538265)

[4.3 Kreiranje događaja temeljenih na stimulansima 11](#_Toc190538266)

[4.4 Segmentacija podataka u epohe 12](#_Toc190538267)

[4.5 Priprema ciljne varijable 12](#_Toc190538268)

[4.6 Detekcija P300 odziva 13](#_Toc190538269)

[4.7 Predviđanje slova temeljenih na P300 odziva 13](#_Toc190538270)

[4.8 Rezultati i ispis predviđanja 14](#_Toc190538271)

[5 Zaključak 14](#_Toc190538272)

Sažetak

Ovaj seminar se bavi razvojem sučelja mozak-računalo (BCI), koje omogućuje korisniku da komunicira s računalom koristeći EEG signale. U radu su obrađeni osnovni koraci poput proučavanja literature, programiranja, generiranje signala te obrade i analize podataka.

Priprema okruženja Za rad je korišten Python i razvojno okruženje PyCharm. Osim toga, koristile su se knjižnice kao što su Expyriment, pylsl, MEDUSA I tkinter koje pomažu u radu s podražajima.

Za **generiranje stimulacija (main.py)**, napravljen je sustav koji korisniku prikazuje vizualne podražaje (slova) u određenim vremenskim intervalima. Cilj je testirati reakcije korisnika na određene stimulacije. Podaci o slovima se spremaju u datoteku **letter\_timestamps.csv** kako bi se poslije mogla napraviti sinhronizacija i otkriti koje slovo se pojavilo u danom vremenu.

Za **generiranje EEG signala (main.py)** korišten je Signal Generator u sustavu Meduze. Čitanje danih signala smo napravili uz pomoć pylsl knjižnice. Generator bilježi 16 x 512 uzoraka u sekundi. Testirano je snimanje i prikaz podataka u stvarnom vremenu te spremanje podataka u **(lsl\_data.csv)** za daljnju analizu.

**Obrada EEG podataka (preprocess.py)**, podaci se obrađuju kroz nekoliko koraka:

1. Razdvajanje signala na manje dijelove (epohe) i povezivanje s podražajima.
2. Uklanjanje srednje vrijednosti signala.
3. Normalizacija i filtriranje podataka radi boljih rezultata.
4. Analiza podataka u offline načinu rada.

Podaci se spremaju u **updated\_lsl\_data.csv**

Za **Klasifikacija EEG podataka (classify.py)** je cilj odrediti koje slovo korisnik promatra na temelju EEG podataka koristeći analizu P300 odziva. Kod analizira EEG signale kako bi prepoznao P300 odziv u P300 speller paradigmi. Nakon učitavanja podataka, signal se filtrira (1-30 Hz) i epohira oko stimulusa. P300 odziv detektira se analizom vrhova signala pomoću praga temeljenog na standardnoj devijaciji. Predikcija slova vrši se većinskim glasanjem unutar grupa od 22 stimulusa, čime se određuje koje je slovo korisnik zamislio.

**Zaključak** napravljen je osnovni sustav za rad sa sučeljem mozak-računalo. Omogućeno je generiranje stimulacija, snimanje EEG podataka, njihova obrada i analiza. Implementiran je pristup koji koristi detekciju P300 vrhova i većinsko glasanje za predikciju slova koje korisnik promatra.

# Generiranje stimulacija

## Postavljanje grafičkog sučelja

Generiranje vizualnih stimulacija je implementirano uz pomoć Tkinter knjižnice, koja omogućava prikaz grafičkog sučelja. Glavni cilj ovog dijela programa je prikazivanje nasumično odabranih slova korisniku u određenim vremenskim intervalima, čime se omogućuje mjerenje EEG signala tijekom vizualne stimulacije.

Na početku programa kreira se glavni prozor aplikacije koristeći Tkinter gdje stavljamo naslov, veličinu fonta i poziciju:

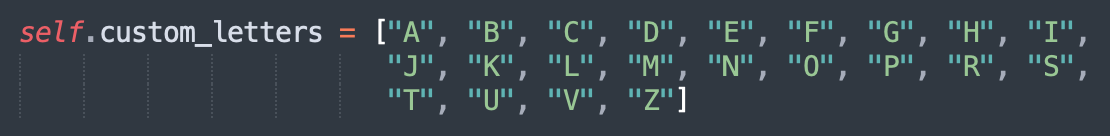
A screenshot of a computer screen

AI-generated content may be incorrect.

Cilj ovog dijela koda je postaviti jednostavno sučelje koje korisniku jasno prikazuje trenutno odabrano slovo.

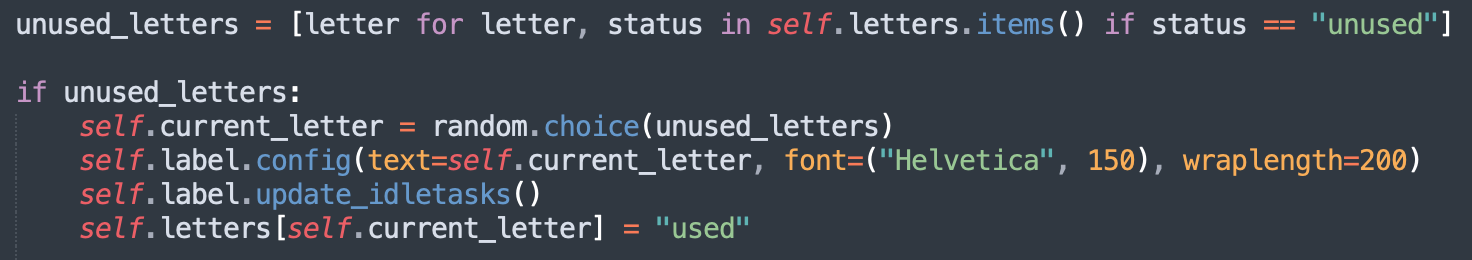
## Generiranje nasumičnih slova

Aplikacija koristi unaprijed definiranu listu slova koja će se nasumično prikazivati korisniku:

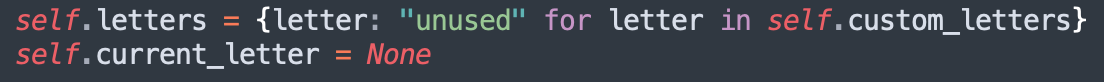


Ova lista sadrži slova abecede koja će se prikazivati korisniku kao vizualni stimulusi. Svakom slovu inicijalno je pridružen status "unused", što znači da još nije prikazano.

Kod zatim bira nasumično slovo koje još nije prikazano:



Ključevi slova (od "A" do "Z"), a vrijednosti su inicijalno "unused". Ovo omogućava praćenje koji su znakovi još uvijek "neiskorišteni" i koji su već "iskorišteni". Letters lista sadrži sva slova sa vrijednosti unused ili use, dok unused\_letters sadrži samo unused slova zbog lakšeg nasumičnog odabira slova.



Ako su sva slova prikazana, program resetira listu i kreće ispočetka:

A computer screen shot of text

AI-generated content may be incorrect. Ovim načinom osiguravamo da se stimulacija odvija u ponavljajućim ciklusima.

## Automatska izmjena slova u intervalima

Kako bi program samostalno mijenjao slova bez ručne intervencije, koristi se metoda **after()**, koja omogućuje ponavljanje funkcije u određenim vremenskim intervalima:



Svake sekunde se poziva funkcija update\_letter(), koja mijenja prikazano slovo. Na ovaj način osigurava se ritmična izmjena stimulacija.

Nakon što program prikaže maksimalno definirani broj slova (npr. 4 iteracije), zaustavlja se:

A computer code with text

AI-generated content may be incorrect.

Ovo osigurava da aplikacija ne radi beskonačno, već prestaje nakon određenog broja ciklusa.

## Pohrana podataka o prikazanim slovima (Markeri)

Funkcija add\_marker() zapisuje ID prikazanog slova, njegovu vrijednost i vrijeme prikaza. Svaki put kada se korisniku prikaže novo slovo, ono se zapisuje u CSV datoteku:

A computer screen shot of a computer code

AI-generated content may be incorrect.

Uz svako slovo se zapisuje timestamp kada se prikazalo, to omogućava povezivanje prikazanog stimulusa s EEG podacima. Podaci se spremaju u datoteku **letter\_timestamps.csv**, koja će kasnije biti korištena za analizu EEG signala.

A table with numbers and letters

AI-generated content may be incorrect.

# Snimanje i generiranje EEG signala

## Generiranje signala

Za testiranje koristili smo Medusa signal generator. Generator bilježi 16 x 512 uzoraka u sekundi. Testirano je snimanje i prikaz podataka u stvarnom vremenu te spremanje podataka za daljnju analizu.

A screenshot of a computer

AI-generated content may be incorrect.

## Snimanje signala

Čitanje danih signala smo napravili uz pomoć pylsl knjižnice. Dok se slova prikazuju, paralelno se pokreće obrada EEG podataka pomoću Lab Streaming Layer (LSL), čime se osigurava da se točno zna kada je prikazano određeno slovo

A blue and white text

AI-generated content may be incorrect.

EEG podaci se kontinuirano prikupljaju i zapisuju u zasebnu CSV datoteku zajedno s informacijama o stimulacijama **(lsl\_data.csv):**

A screen shot of a computer

AI-generated content may be incorrect.

A grey rectangular sign with black text

AI-generated content may be incorrect. A grey and black text

AI-generated content may be incorrect.

Svaka linija u CSV datoteci sadrži: Vrijeme zapisa 16 EEG kanala, vrijednost stimulansa (ako se u danom trenutku prikazalo slovo Stimulus = 1 - koristi se za klasifikaciju) i slovo koje se prikazalo u tom trenutku. Stimulus i Letter vrijednosti se dodaju tokom obrade EEG podataka.

# Obrada EEG podataka

## Usklađivanje slova i EEG vremenskih oznaka

Za svako slovo iz vremenskih oznaka slova, traži se najbliži trenutak u vremenskim oznakama EEG signala, a zatim se označava koji EEG signal je povezan s kojim slovom. Ovaj korak omogućuje povezivanje stimulansa sa stvarnim EEG signalom koji je zabilježen u trenutku stimulacije:

A screen shot of a computer code

AI-generated content may be incorrect.

Zbog šuma i nepoželjnih frekvencija u sirovim EEG podacima, signal se filtrira kako bi se zadržale samo korisne informacije. To se postiže korištenjem bandpass filtra koji propušta samo određeni raspon frekvencija (npr. između 1 Hz i 30 Hz). Na taj način se uklanjaju neželjeni niskofrekventni i visokofrekventni šumovi koji nisu relevantni za analizu.

A screen shot of a computer code

AI-generated content may be incorrect.

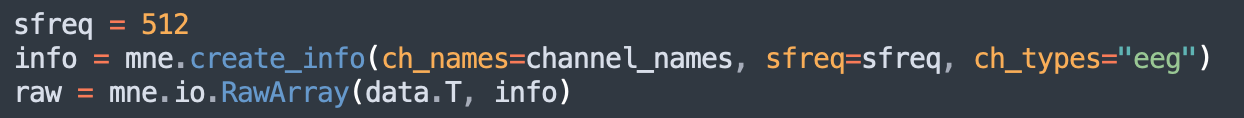
Spremanje procesuiranog signala:



# Klasifikacija EEG podataka

## Priprema podataka za rad s MNE

Iz DataFramea se izdvajaju samo EEG podaci (kanali), dok se zanemaruju informacije o vremenskim oznakama, slovima i stimulansima. Koristi se MNE biblioteka za rad s EEG podacima. Izrađuje se Raw objekt, koji je specifičan format za rad s EEG podacima u MNE-u. Ovdje se koristi informacija o kanalima i uzorkovnoj frekvenciji (512 Hz).



## Filtriranje signala

P300 signal se obično nalazi u opsegu frekvencija između 1 i 30 Hz. Podaci se filtriraju kako bi se uklonile neželjene frekvencije (npr. šum ili spori signali) i kako bi se izdvojile samo relevantne informacije koje uključuju P300 odziv.



## Kreiranje događaja temeljenih na stimulansima

Ovdje smo identificirali trenutak kada je stimulans bio prikazan sudioniku (gdje je "Stimulus" jednak 1). Ti trenuci označavaju početak potencijalnih P300 odziva Na temelju tih stimulansa, generira se niz događaja koji se koriste za segmentiranje podataka u epohe.

A computer code with text

AI-generated content may be incorrect.

## Segmentacija podataka u epohe

Na temelju događaja (stimulansa), podaci se segmentiraju u epohe koje sadrže EEG podatke oko trenutka stimulacije. Svaka epoha traje od 0 ms do 800 ms nakon stimulansa, kako bi se obuhvatio P300 odziv.

Svaka epoha se zatim pretvara u 2D niz, gdje su podaci za sve kanale i vrijeme spojeni u jedan dugi vektor. Ovaj korak omogućava analizu svih podataka u svakom trenutku kao jedinstven vektor za klasifikaciju

A black background with white text and orange and white letters

AI-generated content may be incorrect.

## Priprema ciljne varijable

Povezivanje slova i epoha: Svakom stimuliranju (koje je vezano uz slovo) dodaje se jedinstveni identifikator temeljen na redoslijedu slova koja su prikazana. Ovo omogućuje da se za svaku epohu zna koje slovo je stimulirano.

A computer screen shot of a program code

AI-generated content may be incorrect.

## Detekcija P300 odziva

Za detekciju P300 odziva koristi se metoda detekcije vrhova, koja koristi standardnu devijaciju podataka unutar svake epohe kao prag za detekciju P300 odziva.

Ako je apsolutna vrijednost maksimalnog vrha u epohi veća od praga (koji je postavljen na višestruku standardnu devijaciju), tada se označava da je P300 odziv detektiran.

A computer screen shot of a code

AI-generated content may be incorrect.

## Predviđanje slova temeljenih na P300 odzivu

Nakon što su epohe označene kao sadrže li P300 odziv ili ne, koristi se glasanje većine za grupiranje predviđanja slova.

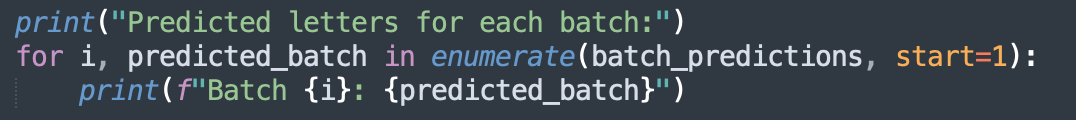
Predviđanja se grupiraju u serije od 22 predviđena slova (koja odgovaraju seriji stimulacija). Za svaku grupu se koristi većinsko glasanje, gdje je najvjerojatniji rezultat slovo koje se najviše puta predvidjelo za tu grupu.

A computer screen shot of a program code

AI-generated content may be incorrect.

## Rezultati i ispis predviđanja

Na kraju, za svaku grupu predviđanja slova, ispisuju se rezultati temeljem glasanja većine. Ako nije detektiran nijedan P300 odziv u nekoj grupi, tada neće biti predviđeno slovo za tu grupu.



Slika na kojoj se prikazuje tekst, snimka zaslona, Font

Opis je automatski generiran

# Zaključak

Ovaj projekt bio je usmjeren na razvoj sučelja mozak-računalo (BCI), koristeći generatore signala za stimulaciju i prepoznavanje P300 odziva. Kroz različite faze projekta, od inicijalnog istraživanja do implementacije ključnih metoda za obradu i klasifikaciju signala, uspjeli smo razviti sustav koji omogućuje prepoznavanje korisničkih namjera temeljenih na P300 odzivu.

U početnim fazama projekta, fokusirali smo se na razvoj podsustava za generiranje stimulacija korisniku. Korištenjem vizualnih podražaja, stimulirali smo korisnika kako bi inducirali P300 odziv. Nakon toga, signal koji smo prikupljali pomoću generatora signala bio je podvrgnut obradi koja je uključivala segmentaciju u epohe, uklanjanje srednje vrijednosti, normiranje i filtriranje, što je omogućilo da se podaci pripreme za klasifikaciju.

Za klasifikaciju smo koristili P300 signal, prepoznajući odgovore korisnika prema njegovom fokusu na određeno slovo. Ovaj proces uključuje dinamičko praćenje P300 odziva u stvarnom vremenu, što je ključno za točnost prepoznavanja slova sa signalom.

Iako su glavni dijelovi sustava uspješno implementirani, budući rad mogao bi se fokusirati na optimizaciju klasifikacije, smanjenje vremena prepoznavanja slova te unapređenje sustava za još preciznije prepoznavanje korisničkih akcija. Također, proširenje sustava na dodatne vrste stimulacija ili unapređenje učenja i prepoznavanja specifičnih obrazaca u P300 odziv moglo bi poboljšati cijeli sustav. Ovaj projekt predstavlja solidan temelj za daljnji razvoj BCI sustava koji omogućuju korisnicima bržu i učinkovitiju interakciju s računalima i tehnologijama, posebno u kontekstu osoba s invaliditetom ili za razvoj naprednih interaktivnih aplikacija.