Universitatea POLITEHNICA din București

Facultatea de Automatică și Calculatoare,
Departamentul de Calculatoare

LUCRARE DE DIPLOMĂ

Instrument informatic pentru procesarea și vizualizarea electroencefalogramelor

Autor:
Dochian Alexandru Adrian

Conducători Științifici:
sl. dr. ing. Mihai Trăscău
sl. dr. ing. Alexandru Sorici
sl. dr. ing. Irina Andra Tache

București, Iulie 2022

Această lucrare nu ar fi fost posibilă fără ajutorul catedrei de fiziologie din cadrul Facultății de Medicină, UMF "Carol Davila", București care a achiziționat electroencefalograme intracraniene de calitate superioară pe regretați șoareci și le-a împărtășit cu mine și coordonatorii mei.

De asemenea adresez mulțumiri coordonatorilor mei, Domnului sl. dr. ing. Mihai Trăscău, Domnului sl. dr. ing. Alexandru Sorici și Doamnei sl. dr. ing. Irina Andra Tache pentru timpul, energia și cunoștiințele dedicate acestei lucrări.

Compendiu

Electroencefalogramele sunt semnale care poartă informații despre evoluția potențialui electric într-un punct ales din cortexul subiectului și sunt reprezentate ca simple serii de timp. Ele sunt utilizate în domeniul medical pentru diagnosticarea sau predicția de patologii. În lucrare de față este prezentat un sistem informatic de analiză a electroencefalogramelor. Analiza presupune procesarea informațiilor utilizând mai multe fluxuri configurabile formate la rândul lor dintr-un număr variabil de stagii de procesare. În vederea analizei, informațiile inițiale trec prin fluxuri configurabile de preprocesare și procesare. Fiecare flux este la rândul său format dintr-un număr variabil de stagii procesatoare. Setul de date constă în electrocorticograme din lobii frontali și parietali și electrocardiograme achiziționate pe șoareci. Pe parcursul colectării datelor, subiecții au fost stimulați cu impulsuri electrice, de asemenea înregistrate. Fluxul de preprocesare modifică semnalele în domeniul timp, iar fluxul de procesare extrag caracteristici din domeniile timp și frecvență. Caracteristicile extrase sunt analizate și vizualizate cu ajutorul algoritmului t-SNE(t-Distributed Stochastic Neighbor Embedding). Sistemul prezentat este un instrument informatic ce poate fi folosit pentru a analiza astfel de experimente. Arhitectura sistemului este flexibilă și permite extinderea facilă a fluxurilor existente.

Cuprins

Capitolul 1

Introducere

1.1 Electroencefalografia

Electroencefalografia este o modalitate prin care se relevă informații despre activitatea electrică a stratului de la suprafața creierului. Informația rezultată în urma procesului de achiziționare este evoluția in timp a potențialului electric în punctul de interes.

Electroencefalografia înregistrază date de pe scalpul subiecțiilor și de aceea este predispusă la zgomot. Electrocorticografia sau electroencefalografia invazivă a fost inventată și rafinată. Ea presupune achiziționarea datelor direct de pe cortexul subiectului.

1.1.1 Fundamente fizice

Electroencefalogramul este un dispozitiv care folosește electrozi plasați pe scalpul subiectului pentru a înregistra fluctuațiile de voltaj din punctul de interes. Existența acestui voltaj ce poate fi măsurat se datorează existenței unui curent ionic între neuroni.

Electrozii sunt conductori electrici folosiți pentru a face contact cu părți nemetalice ale unui circuit electric. Pot fi construiți din mai multe materiale dintre care menționăm: Litiu (Li), Mangan (Mn), Cupru (Cu) și Zinc (Zn)

Un ion este un atom care prezintă un dezechilibru de sarcină electrică. Anionul este un ion format din mai mulți electroni decât protoni, astfel având o sarcină electrică negativă. Cationul este oposul anionului, fiind format din mai mulți protoni decât electroni și implicit are o sarcină electrică pozitivă. Încheind această prezentare a fundamentelor fizice, curentul ionic este un flux de de electroni produs de grupări atomice de ioni și întâlnit la diverse nivele ale materiei.

1.1.2 Istorie

În anul 1875 fizicanul englez Richard Canton a prezentat în jurnalul British Medical Journal descoperile sale despre feneomenele electrice relevate pe cortexuri de iepuri și maimuțe.

De-a lungul vremii oameni de știință ca fiziologul polonez Adolf Beck sau fiziologul ucraineen Vladimir Vladimirovich Pravdich-Neminsky au studiat aceste fenomene electrice ale creierului. În experimentele lor subiecții erau animale supuse la diverși stimuli exteriori.

Un eveniment de o importanță majoră în istoria electroencefalografiei este produs de fizologul și psihiatrul german Hans Berger care reușește să achiziționeze primele date pe un subiect uman. Numește dispozitivul propus de el "Electroencefalogram" și astfel netezește și potențează această calea științifică ce și-a dovedit până în zilele noastre potențialul.

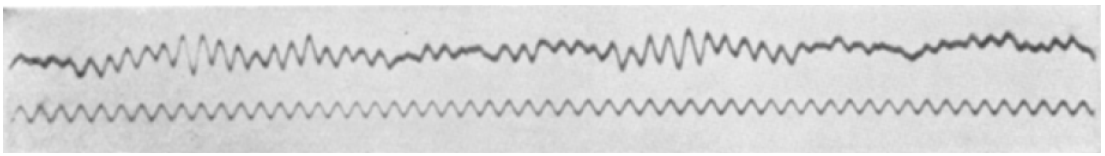

Figura 1.1: Primul EEG uman achiziționat de către Hans Berger în 1924. Semnalele din imagine sunt eșantionate la o frecvență de 10 Hz.

1.1.3 Aplicații

Activitatea creierului observată cu ajutorul acestei metode s-a dovedit prolifică în domeniul medical. Electroencefalogramele sunt utile în diagnosticarea și tratarea unor boli ca epilepsia, tumorile pe creier sau accidentele vasculare cerebrale. Unele boli ca apoplexia pot fi chiar și prezise uzitând activitatea cerebrală relevată.

1.2 Prezentarea sistemului

1.2.1 Datele folosite

Catedra de fiziologie de la Facultatea de Medicină , UMF "Carol Davila", București ne-a onorat cu împărtășirea unor electroencefalograme invazive (electrocorticograme) obținute din experimente efectuate pe șoareci.

Pentru achiziționarea electrocorticogramelor, părțile anatomice care projeau cortexul subiecților au fost eliminate și anestezice au fost administrate direct în creier prin injecție. Electrozi au fost plasați direct pe cortex în zonele corespunzătoare lobilor frontali și parietali astfel prelevându-se date. În timpul experimentului subiecții au fost supuși unor stimuli electrici, stimuli a căror activitate a fost de asemenea înregistrată. În cadrul unora dintre experimentele respective s-a înregistrat și activitatea EKG. Frecvența de eșantionare folosită pentru înregistrarea acestor semnale a fost de 1000 Hz. Prin aceste experimente catedra de fiziologie încearcă să găsească efectele pe care anumite substanțe anestezice le au asupra activității cerebrale.

Patru experimente au fost efectuate pe șobolani diferiți astfel obținându-se electrocorticograme. De-a lungul acestei lucrări experimentele vor fi recunoscute prin fișierele rezultate corespunzătoare: **mouse1.acq**, **mouse2.acq**, **mouse3.acq** și **mouse4.acq**. Setul de date conține următoarele informații:

- **mouse1.acq (Șoarecele 1)**
  - ECoG F (lobul frontal)

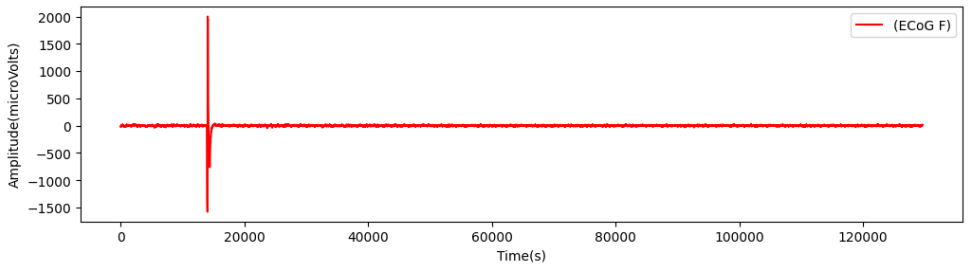

Figura 1.2: Lobul frontal al șoarecelui 1

  - ECoG P (lobul parietal)

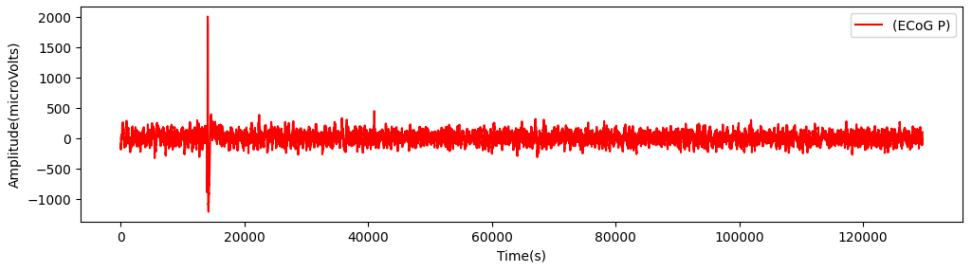

Figura 1.3: Lobul parietal al șoarecelui 1

– STIM (stimularea)

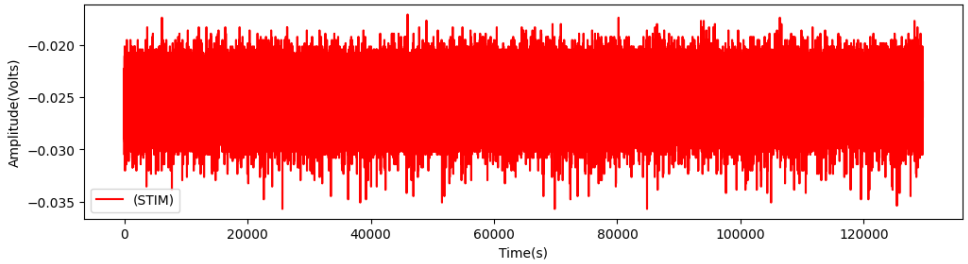

Figura 1.4: Stimularea șoarecelui 1

Proprietățile semnalelor achiziționate pe șoarecele 1							
Canal	Unitate de măsură	Frecvență de eșantionare	Secunde	Medie	Deviație	Minim	Maxim
Lobul Frontal	microvolți	1000	129.59	0.5917	61.9097	-1575.7446	1999.9390
Lobul Pari-etal	microvolți	1000	129.59	6.4214	104.1361	-1214.1113	1999.9390
Stimulare	volți	1000	129.59	-0.0248	0.0023	-0.0357	-0.0171

- **mouse2.acq (Șoarecele 2)**

– ECoG F (lobul frontal)

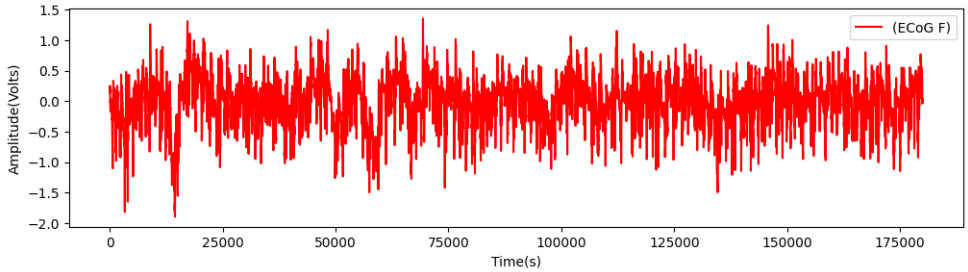

Figura 1.5: Lobul frontal al șoarecelui 2

– ECoG P (lobul parietal)

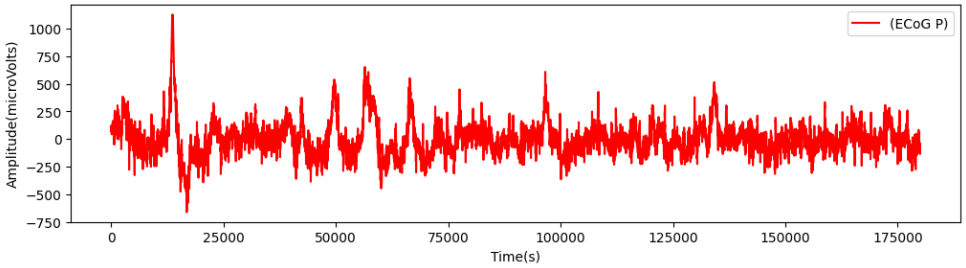

Figura 1.6: Lobul parietal al șoarecelui 2

– STIM (stimularea)

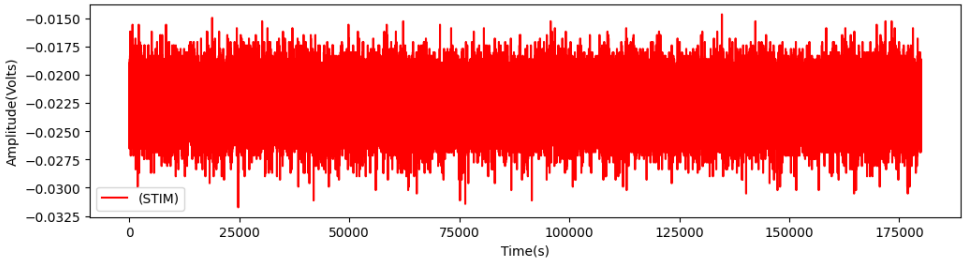

Figura 1.7: Stimularea șoarecelui 2

– EKG (electrocardiograma)

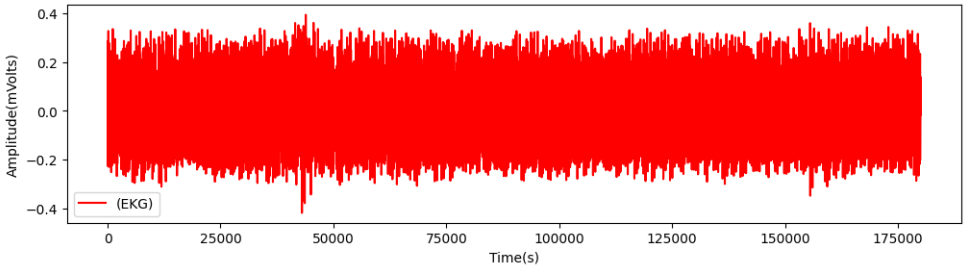

Figura 1.8: Electrocardiograma șoarecelui 2

Proprietățile semnalelor achiziționate pe șoarecele 2							
Canal	Unitate de măsură	Frecvență de eșantionare	Secunde	Medie	Deviație	Minim	Maxim
Lobul Frontal	volți	1000	179.99	-0.0017	0.3940	-1.8958	1.3522
Lobul Pari-etal	microvolți	1000	179.99	-3.2591	152.6158	-660.9497	1124.4507
Stimulare	volți	1000	179.99	-0.0225	0.0018	-0.0317	-0.0146
EKG	milivolți	1000	179.99	0.0147	0.1034	-0.4187	0.3937

- **mouse3.acq (Șoarecele 3)**

– ECoG F (lobul frontal)

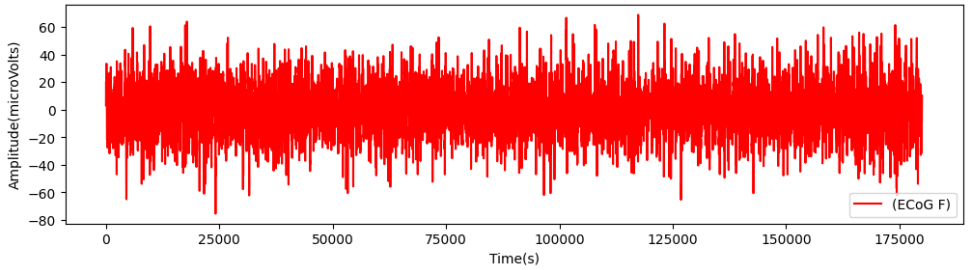

Figura 1.9: Lobul frontal al șoarecelui 3

– ECoG P (lobul parietal)

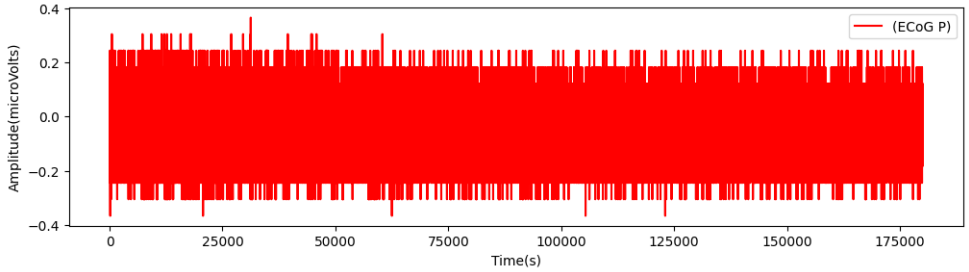

Figura 1.10: Lobul frontal al șoarecelui 3

– STIM (stimularea)

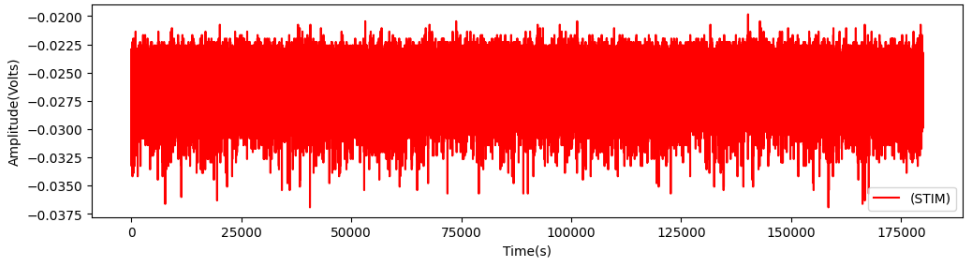

Figura 1.11: Stimularea șoarecelui 3

Proprietățile semnalelor achiziționate pe șoarecele 3							
Canal	Unitate de măsură	Frecvență de eșantionare	Secunde	Medie	Deviație	Minim	Maxim
Lobul Frontal	microvolți	1000	180.00	-0.0014	16.4453	-75.4837	68.8124
Lobul Pari-etal	microvolți	1000	180.00	-0.0359	0.0849	-0.3662	0.3662
Stimulare	volți	1000	180.00	-0.0264	0.0017	-0.0369	-0.0198

- **mouse4.acq (Șoarecele 4)**

– ECoG F (lobul frontal)

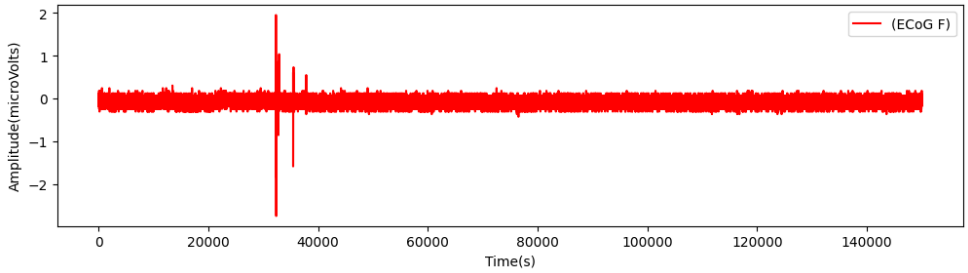

Figura 1.12: Lobul frontal al șoarecelui 4

– ECoG P (lobul parietal)

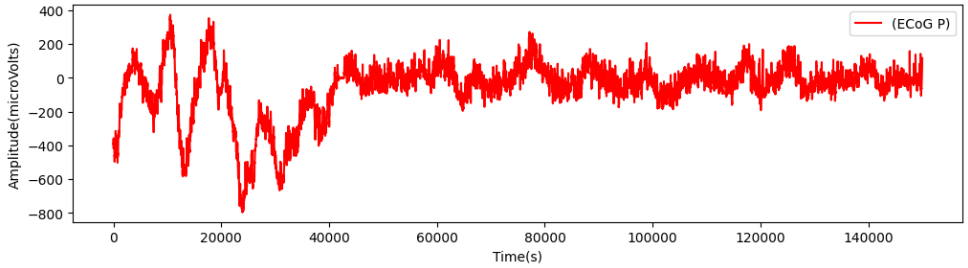

Figura 1.13: Lobul parietal al șoarecelui 4

– STIM (stimularea)

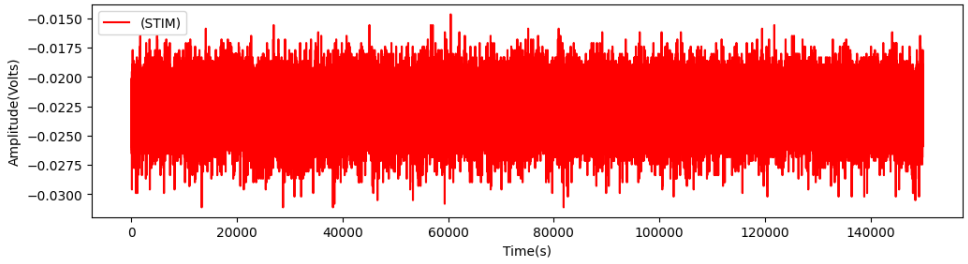

Figura 1.14: Stimularea șoarecelui 4

– EKG (electrocardiograma)

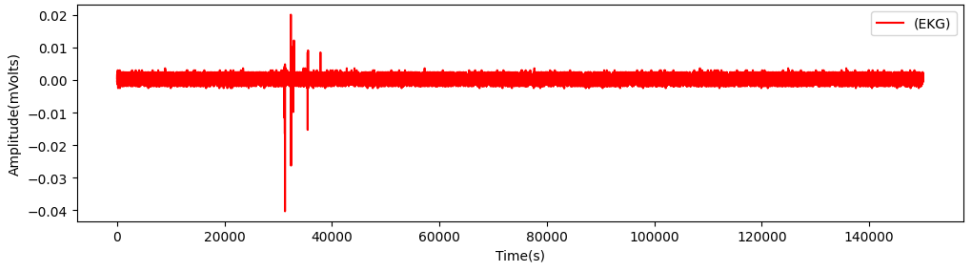

Figura 1.15: Electrocardiograma șoarecelui 4

Proprietățile semnalelor achiziționate pe șoarecele 4							
Canal	Unitate de măsură	Frecvență de eșantionare	Secunde	Medie	Deviație	Minim	Maxim
Lobul Frontal	microvolți	1000	149.99	-0.0802	0.0776	-2.7466	1.9531
Lobul Pari-etal	microvolți	1000	149.99	-59.2345	165.2851	-794.4336	372.8027
Stimulare	volți	1000	149.99	-0.0227	0.0018	-0.0311	-0.0146
EKG	milivolți	1000	149.99	0.0003	0.0008	-0.0403	0.0201

1.2.2 Analiza clasică a electroencefalogramelor

În mod normal astfel de informații se analizează ochiometric de către oameni specializați în acest sens. Un neurolog se antrenează timp îndelungat să dezvolte abilitățile necesare interpretării acestor semnale. Fără un sistem de calcul, analiza umană se poate face doar prin observarea unor lungi secvențe de semnale.

1.2.3 Analiza electroencefalogramelor cu ajutorul sistemului

Cu această lucrare, încercăm să oferim o nouă perspectivă pentru analiza semnalelor electroencefalografice ca Lobul Frontal sau Lobul Parietal ale șobolanului 2.

Sistemul informatic prezentat în această lucrare procesează electroencefalograme cu ajutorul învățării automate nesupervizate și le proiectează într-un spațiu bidimensional.

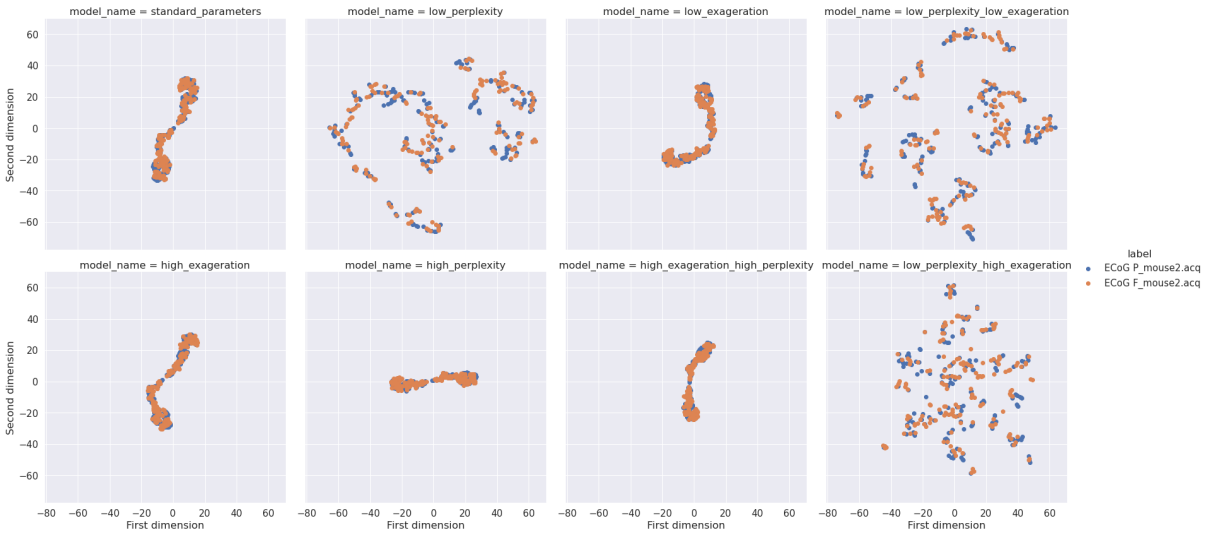

Figura 1.16: Excuția sistemului pe electroencefalogramele prezentate în Secțiunea 1.2.1

Modul în care sistemul produce aceste rezultate va fi explicitat în următoarele secțiuni ale lucrării.

Capitolul 2

Arhitectură

În acest capitol se va prezenta și explicita aplicația din diverse unghiuri și nivele de adâncime.

2.1 Arhitectura codului

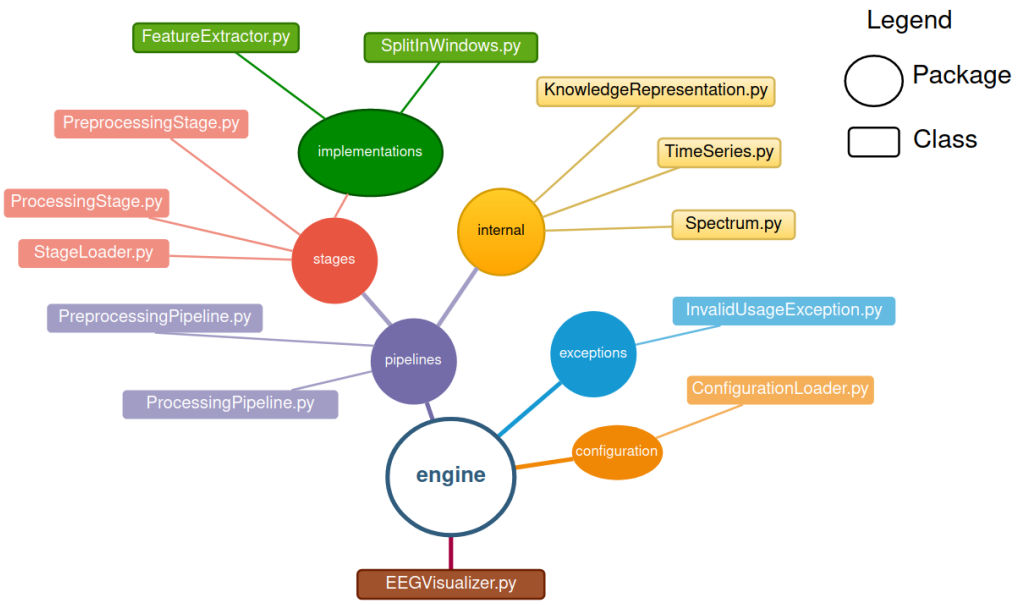

Figura 2.1: Structura codului

Sistemul este implementat folosind limbajul de programare Python.

Fiecare fișier din ierarhia prezentată în Structura codului conține doar o clasă de python cu același nume.

2.1.1 Pachetul engine

Pentru inițierea sistemului trebuie creat un obiect din clasa **EEGVisualizer**. Obiectul expune metodele necesare producerii de rezultate și creează la nevoie obiectele necesare din restul sistemului.

În mod implicit electroencefalogramele prezentate în Secțiunea 1.2.1 sunt încărcate în memorie la instanțierea clasei **EEGVisualizer**.

În continuare o să exemplific modul în care s-a folosit sistemul pentru generarea rezultatelor prezentate în Figura 1.2.3.

```
1
2 from engine.EEGVisualizer import EEGVisualizer
3
4 eeg_visualizer = EEGVisualizer()
5 eeg_visualizer.build(files_name=["mouse2.acq"], eeg_channels=["ECoG␣
    F", "ECoG␣P"])
6 eeg_visualizer.execute()
7 eeg_visualizer.plot()
```

Listarea 2.1: Execution

2.1.2 Pachetul pipelines

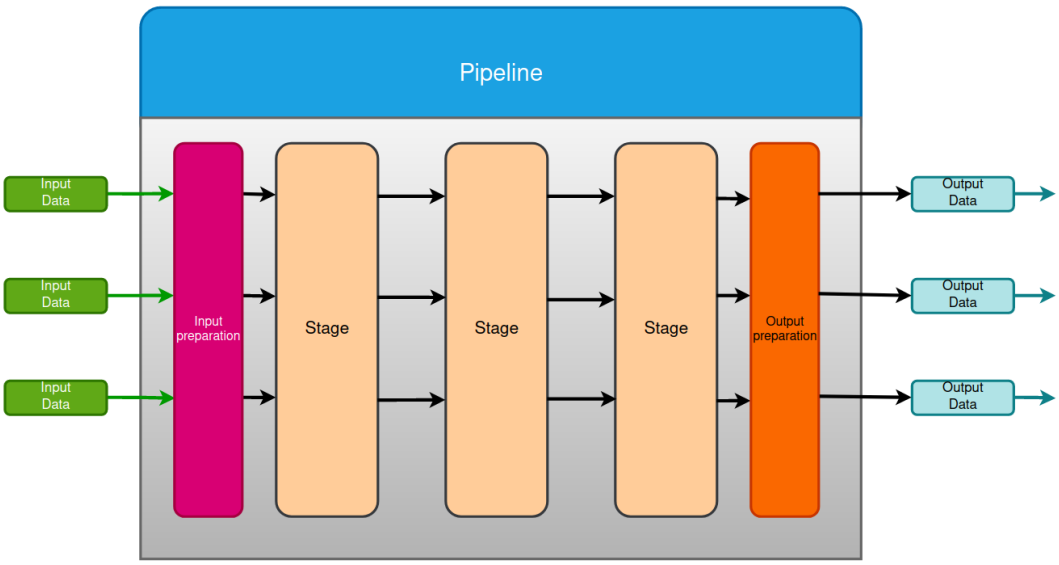

Figura 2.2: Perspectivă globală asupra funcționalității unui flux

În Figura de mai sus observăm comportarea abstractă a unui flux de procesare.

În acest sistem au fost implementate două astfel de fluxuri:

- Flux de preprocesare
- Flux de procesare

Fiecare flux își are propriul protocol de comunicare cu restul sistemului și conține un număr variabil de stagii interne de comportament corespunzător. Merită menționat că un stagiu din fluxul de preprocesare nu poate fi folosit în fluxul de procesare sau invers deoarece implementează interfețe diferite.

Pachetul stages

Aici sunt plasate clasele abstracte care definesc comportamentul ce trebuie respectat de un stagiile efective:

- Stagiu de preprocesare
- Stagiu de procesare

Stagiile instanțiabile ce intră, din perspectiva execuției, în componența fluxurilor sunt definite în pachetul implementations.

Pachetul implementations

Clasele definite în acest pachet trebuie să respecte unul dintre protocoalele impuse de cele 2 stagii abstracte ale sistemului. Sistemul a fost construit într-o manieră deschisă la extensie. În eventualitatea îmbogățirii sistemului cu noi stagii și integrarea lor în execuție ar trebui urmați următorii pași simpli:

1. Crearea clasei care implementează protocolului impus de superclasa aleasă
2. Actualizarea configurației de execuție servită de pachetul configuration

Pachetul internal

Electroencefalogramele pe care sistemul le primește din exterior sunt în formatul bibliotecii bioread. Un obiect de tipul **bioread.biopac.Datafile** conține informații achiziționate pe mai multe canale de tipul **bioread.biopac.Channel**.

Sistemul traduce aceste intrări într-o logică proprie, generică, **KnowledgeAcquisition**.

Problema procesării electroencefalogramelor se reduce în mod implicit la a procesa semnale în domeniile timp și frecvență. Considerând acest lucru am construit clasele **TimeSeries** și **Spectrum**.

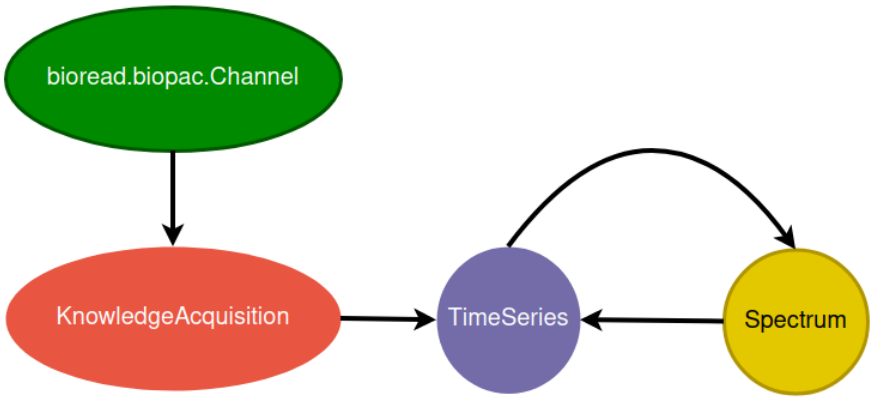

Figura 2.3: Relaționarea obiectelor din pachetul **internal**

TimeSeries și **Spectrum** conțin din punct de vedere matematic aceeași informație esențială.

Sistemul permite, cu ajutorul **transformatei Fourier** obținerea unei instanțe **Spectrum** dintr-o instanță **TimeSeries** și viceversa.

2.1.3 Pachetul exceptions

Excepțiile definite la nivel de sistem au fost plasate în acest pachet pentru modularitate și pentru înlesnirea dezvoltării următoarelor versiuni.

2.1.4 Pachetul configuration

Pachetul respectiv oferă restului aplicației clasa ConfigurationLoader. La nevoie, alte părți ale sistemului cer din acest pachet configurația de execuție a sistemului. Un exemplu de configurație de execuție servită de acest pachet este prezentată în Listarea A.1

Capitolul 3

Execuția sistemului

În acest capitol se va explicita pas cu pas modul în care sistemul procesează electroencefalogramele.

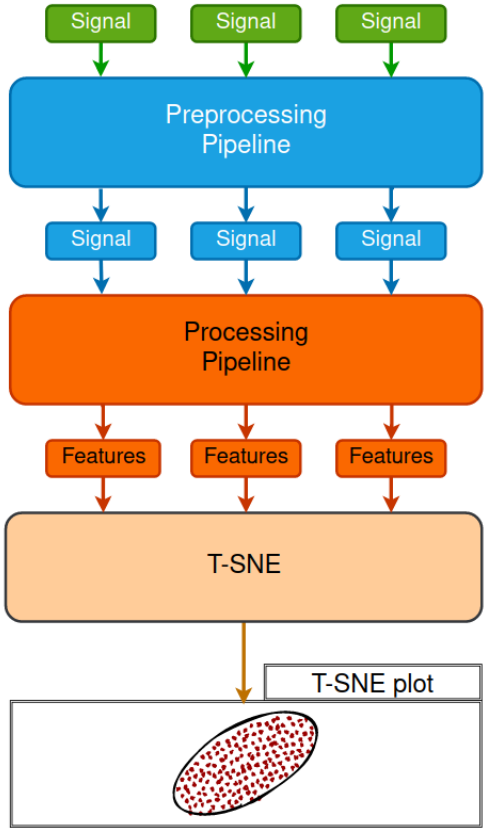

Figura 3.1: Structura execuției

3.1 Inițializarea sistemului

Pentru inițializarea sistemului este nevoie de:

- O configurație de execuție ca cea din Listarea A.1

- Fișiere cu extensia **.acq** create de programul AcqKnowledge ce vor fi încărcate în memorie
- Alegerea unor fișiere și a unor canale de achiziție comune tuturor fișierelor alese

În mod implicit electroencefalogramele prezentate în Secțiunea 1.2.1 sunt încărcate în memorie la instanțierea clasei **EEGVisualizer** și dacă nu primește o nouă configurație de execuție o va folosi pe cea din Listarea A.1.

Configurația de execuție este reprezentată în formatul json(JavaScript Object Notation) și conține 3 chei.

preprocessing_pipeline_stages și processing_pipeline_stages

Primele două chei, **preprocessing_pipeline_stages** și **processing_pipeline_stages**, corespund celor două fluxuri de execuție din Figura 3 și conțin colecții cu 0 sau mai multe obiecte de tipul:

```
{
  "stage_name": "SplitInWindows",
  "constructor_kwargs": {
    "window_size": 1000
  }
}
```

Listarea 3.1: Exemplu unui stagiu definit la nivelul configurației de execuție

Existența unui astfel de obiect în configurația unui flux implică, în timpul procesării, instanțierea clasei corespunzătoare cu parametrii specificați.

t_sne_models

Ultima cheie din configurația de execuție, **t_sne_models**, conține o colecție cu cel puțin un obiect de următoarea structură:

```
{
  "model_name": "standard_parameters",
  "parameters": {
    "n_jobs": 6,
    "learning_rate": "auto",
    "init": "random",
    "n_components": 2,
    "early_exaggeration": 12,
    "perplexity": 30
  }
}
```

Listarea 3.2: Exemplu unui model T-SNE definit la nivelul configurației de execuție

Numărul de modele regăsite în configurație implică numărul de tablouri rezultate. Am gândit o astfel de funcționalitate deoarece hiperparametrii unui model T-SNE influențează mult rezultatul.

3.2 Fluxul de preprocesare

Fluxul de preprocesare și stagiile sale interne respectă același protocol do comunicare, primesc și produc o colecție de obiecte de tipul **TimeSeries**. În acest punct al execuției se fac procesările

dorite în domeniul timp. Sistemul poate fi ușor extins în viitor cu alte stagi, cu condiția să respecte protocolul impus.

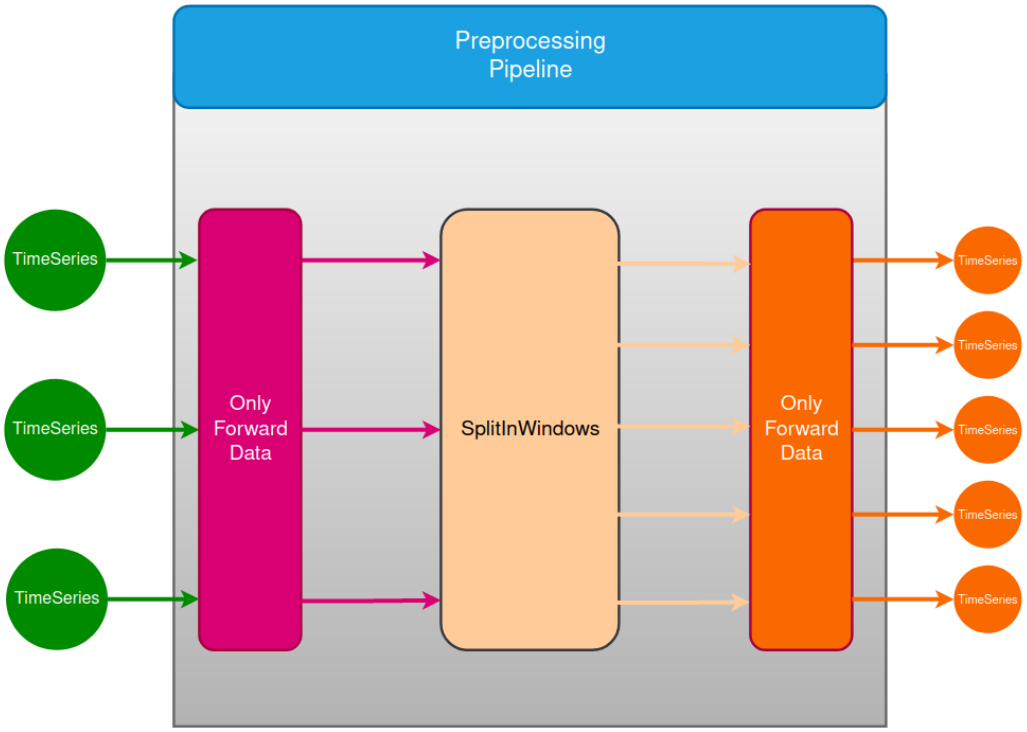

Figura 3.2: Modul de funcționare al fluxului de preprocesare

3.2.1 SplitInWindows

Dacă sistemul ar executa cu un flux vid de preprocesare atunci pentru fiecare semnal de intrare sistemul va produce un punct în tabloul rezultat. Observăm că în această situație procesarea a doar două electroencefalograme nu ar putea produce un rezultat relevant. Este evident că un grafic cu două puncte relevă foarte puține informații despre semnalele procesate.

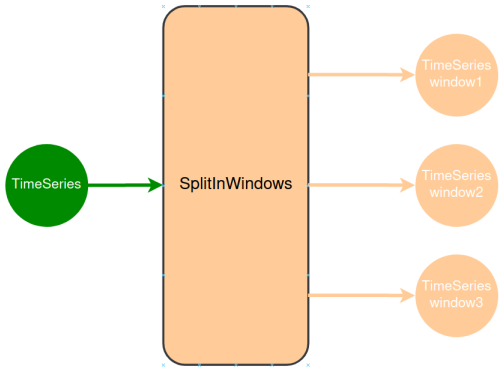

Figura 3.3: Modul de funcționare al stagiuliu SplitInWindows

Stagiul de preprocesare SplitInWindows rezolvă problema procesării unui număr mic de exemple prin multiplicarea un semnal în ferestre de dimensiune fixă păstrând etichetate ca semnalul

inițial. Folosirea unui astfel de stagiu permite și procesarea unui singur semnal, astfel relevând informații despre evoluția sa în timp.

3.3 Fluxul de procesare

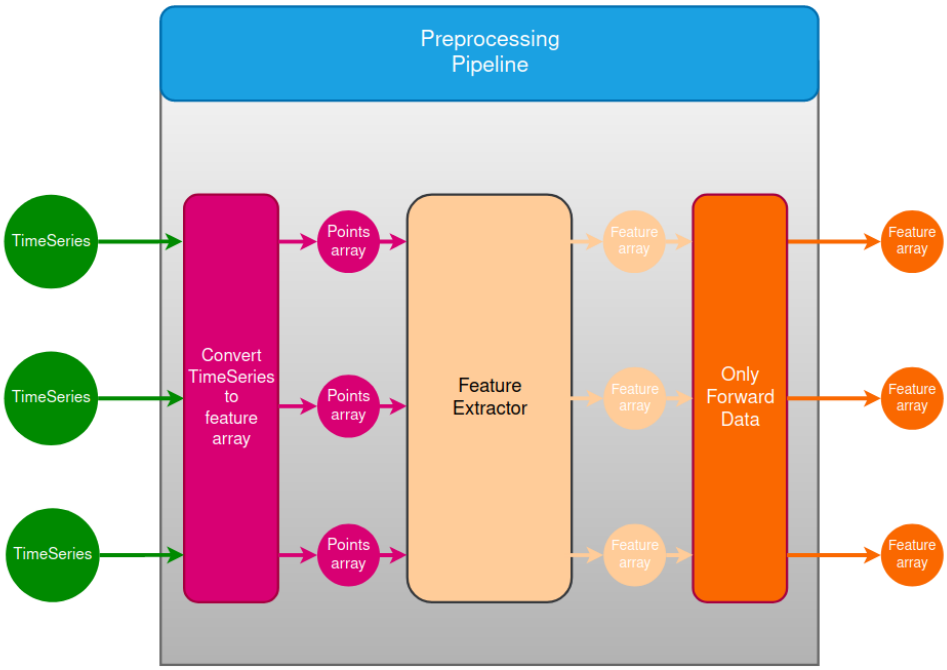

Figura 3.4: Modul de funcționare al fluxului de procesare

Fluxul de procesare conține stagii care acționează asupra caracteristicilor semnalelor. Din perspectivă globală, acest flux traduce intrări de tipul **TimeSeries** în vectori de caracteristici. Stagiile interne primesc vectori de caracteristici și produc tot vectori de caracteristici. Observăm că fluxul, din perspectiva integrării cu restul sistemului, nu are același protocol cu stagiile sale interne. Până ca datele să fie trimise către primul stagiu se construiește vectorul de caracteristici din semnalul inițial. Acest vector de caracteristici inițial conține punctele semnalului.

Acest flux, la fel ca și cel de procesare poate fi ușor extins cu noi stagii care respectă protocolul impus.

3.3.1 FeatureExtractor

Acest stagiu primește ca intrare un vector de valori de dimensiune N și produce un vector de dimensiune M, unde M este numărul de caracteristici alese prin prisma configurației de execuție:

Luând ca exemplu configurația standard de execuție prezentată în Listarea A.1 numărul M de caracteristici este egal cu lungimea vectorului **features**.

```
1 {
2   "stage_name": "FeatureExtractor",
3   "constructor_kwargs": {
4     "features": [
5       "alpha_spectrum_coeffiecients_sum",
6       "beta_spectrum_coeffiecients_sum",
```

```
7         "low_gamma_spectrum_coeffiecients_sum",
8         "high_gamma_spectrum_coeffiecients_sum",
9         "delta_spectrum_coeffiecients_sum",
10        "theta_spectrum_coeffiecients_sum",
11        "mean",
12        "standard_deviation"
13      ]
14    }
15  }
```

Listarea 3.3: Execution

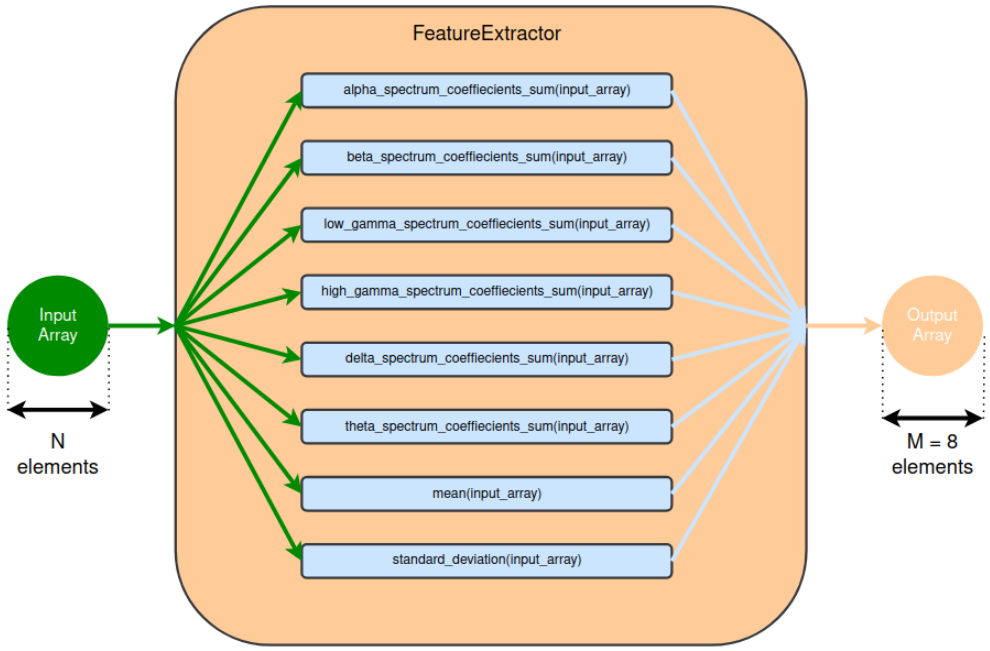

Figura 3.5: Modul de funcționare al stagiului FeatureExtractor

În cazul acestui stagiu observăm un nivel în plus de configurabilitate. Numărul de caracteristici și tipul lor este dat de modul de execuție ales. Pentru variante ulterioare ale sistemului, introducerea unei noi caracteristici în acestă clasă se poate face în 2 pași simpli:

1. Se creează o nouă metodă în clasa FeatureExtractor. Noua metodă trebuie să respecte același protocol cu celelalte metode de extragere de caracteristici definite deja.
2. Numele metodei trebuie adăugat în vectorul **features** din configurația corespunzătoare clasei FeatureExtractor.

Exemplu de caracteristică extrasă de sistem: alpha_spectrum_coeffiecients_sum

Frecvențele de tăiere folosite de sistem pentru obținerea undelor cerebrale sunt următoarele:

- delta : (1, 4) Hz
- theta : (4, 8) Hz
- alpha : (8, 12) Hz

- beta : (13, 30) Hz
- low_gamma : (30, 70) Hz
- high_gamma : (70, 150) Hz

Caracteristica **alpha_spectrum_coeffiecients_sum** calculează suma coeficienților Fourier corespunzători benzii de frecvență alpha.

Alegem spre exemplificare lobului frontal al șoarecelui 3:

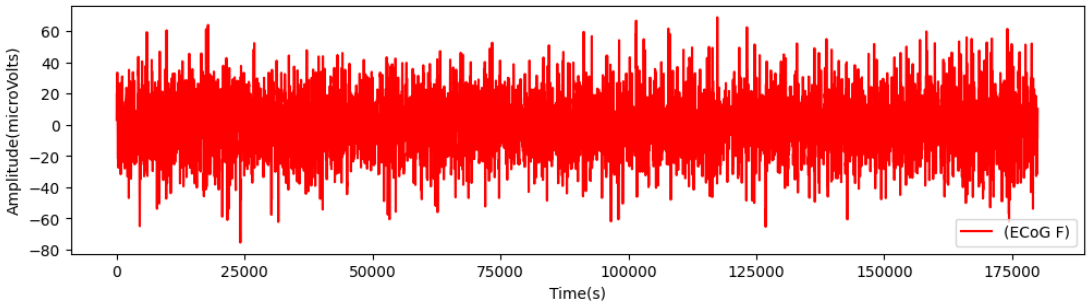

Figura 3.6: Lobului frontal al șoarecelui 3

Cu următorii coeficienți Fourier:

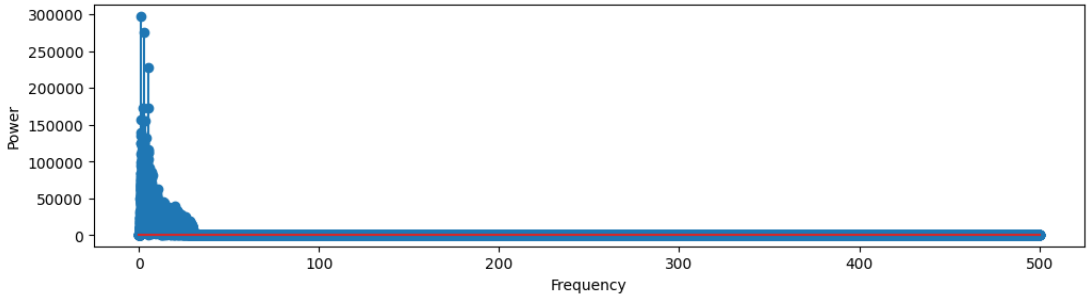

Figura 3.7: Spectrul complet al lobului frontal al șoarecelui 3

Se realizează o filtrare bandă între 8 Hz și 12 Hz și se calculează suma coeficienților Fourier (Suma este egală cu 66959256):

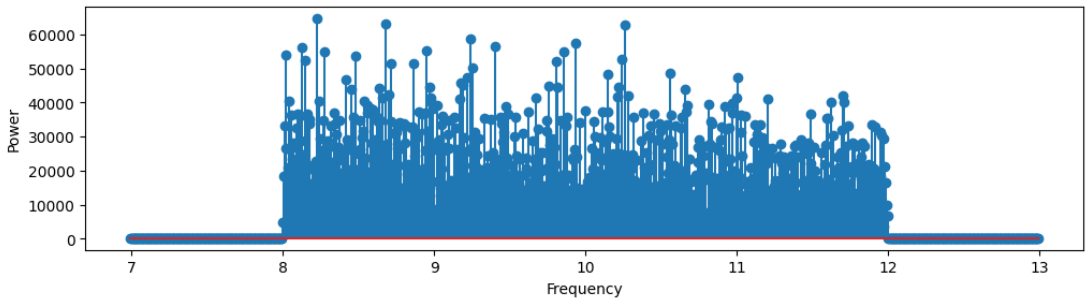

Figura 3.8: Banda de frecvență alpha a lobului frontal al șoarecelui 3

3.4 Execuția algoritmului t-SNE

În urma fluxului de procesare fiecare semnal este reprezentat într-un spațiul de dimensiune M (considerând configurația standard de execuție, M = 8). Desigur că nu ne-am putea opri aici,

după fluxul de procesare, cu analiza datelor deoarece este prea greu, dacă nu chiar imposibil, pentru oameni să poată imagina un spațiu M-dimensional sau să raționeze gradul de similaritate dintre datele ce aparțin unui astfel de spațiu.

Avem nevoie de un mecanism care să ne poată proiecta aceste date neinteligibile într-o formă specifică modului nostru de înțelegere a lucrurilor. Aceasta este situația în care pasul final al analizei sistemului, execuția algoritmului t-SNE, intră în scenă.

3.4.1 Despre t-SNE

T-SNE (t-distributed Stochastic Neighbour Embedding) este un algoritm care reușește să grupeze date dintr-un spațiu de dimensiune mare, **M**, într-un spațiu de dimensiune mică, **d**, păstrând în același timp relațiile de similaritate existente în spațiul inițial: $M >> d \; și \; d = 2$

În continuare voi specifica modul în care t-SNE reușește să proceseze datele generate de fluxul de procesare. Ipotezând că fluxul de procesare produce următoarea mulțime de elemente:

$$X = \{x_1, x_2, ..., x_N\}, \; x_i \in \mathcal{R}^M \; \forall \; i \in [1, N]$$

t-SNE va produce următorul rezultat:

$$Y = \{y_1, y_2, ..., y_N\}, \; y_i \in \mathcal{R}^d \; \forall \; i \in [1, N]$$

3.4.2 t-SNE pe un alt set de date

În această secțiune o să exemplific cum algoritmul t-SNE reușește să producă acea informație bidimensională pe un set de date popular, MNIST.

MNIST este un set de date ce conține exemple ce cifre scrise de mână:

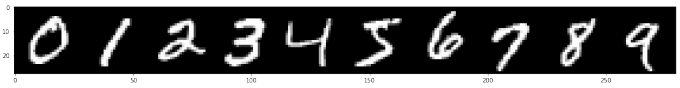

Figura 3.9: Vizualizarea setului de date MNIST

Fiecare exemplu este o matrice de dimensiune (28, 28) cu valorile elementelor în intervalul [0 : 255], nuanțe de gri.

Dacă remodelăm forma tuturor exemplelor, din matrici de (28, 28) în vectori coloană de forma (784, 1), obținem forma datelor pe care t-SNE o așteaptă.

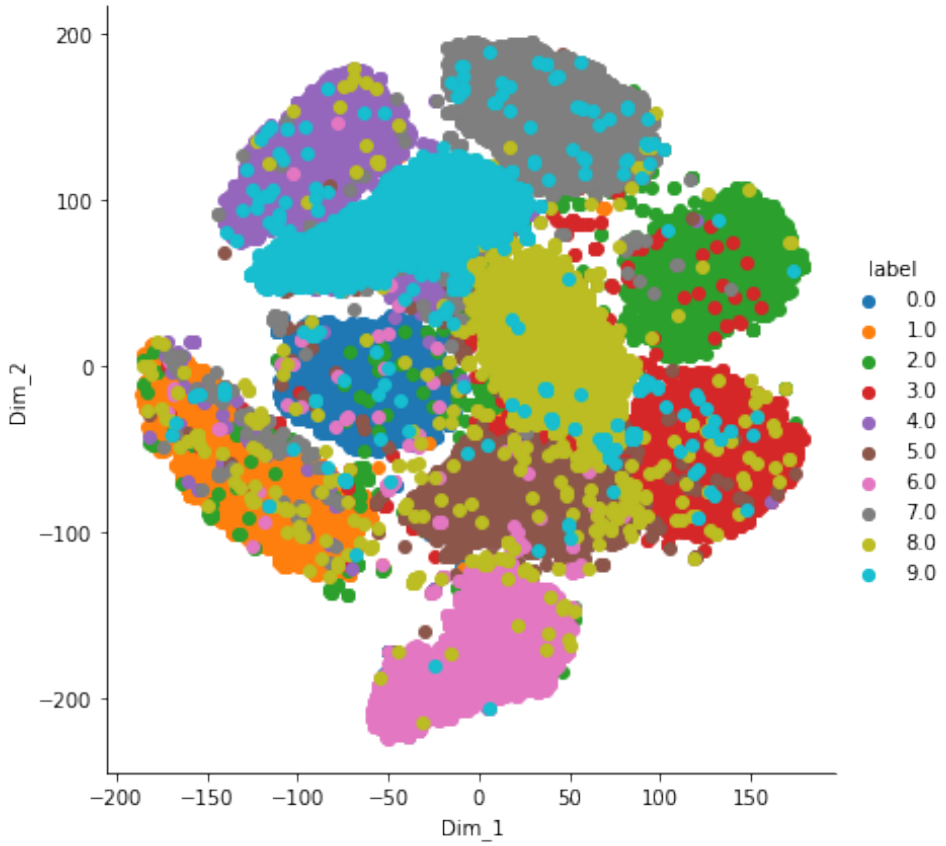

Figura 3.10: Execuția algoritmului T-SNE pe setul de date MNIST

Observăm cum algoritmul reușește să găsească similarități între datele inițiale și să le grupeze în spațiul bidimensional rezultat.

Capitolul 4

Rezultate și Concluzii

TODO

4.1 Lobii frontali și parietali ai primului șoarece(mouse1.acq)

TODO

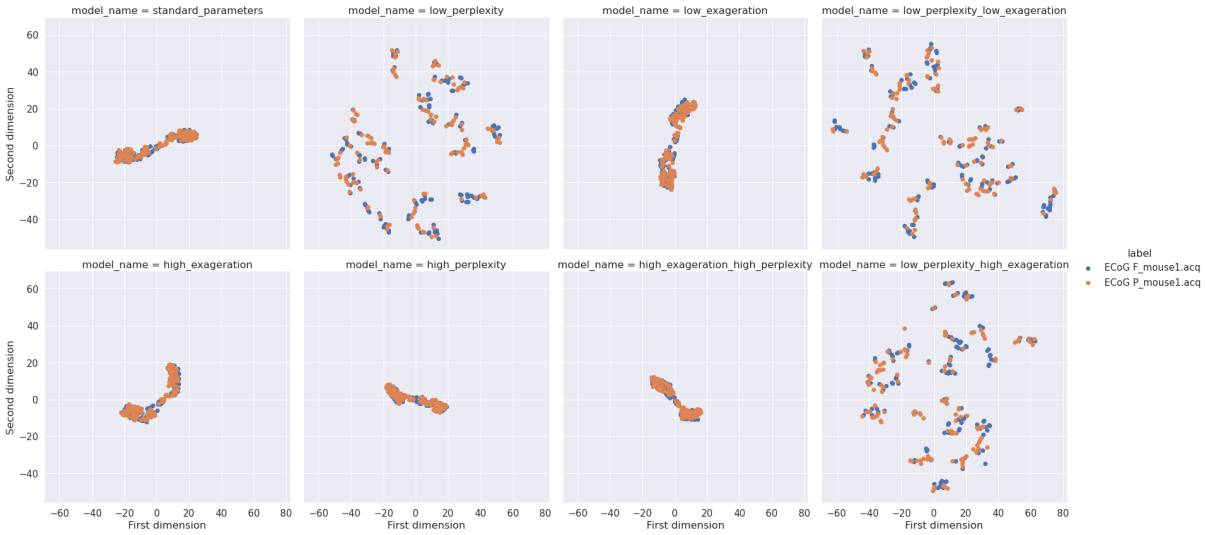

Figura 4.1: Tabloul lobiilor frontali și parietali ai primului șoarece

4.2 Lobii frontali și parietali ai celui de-al doilea șoarece(mouse2.acq)

TODO

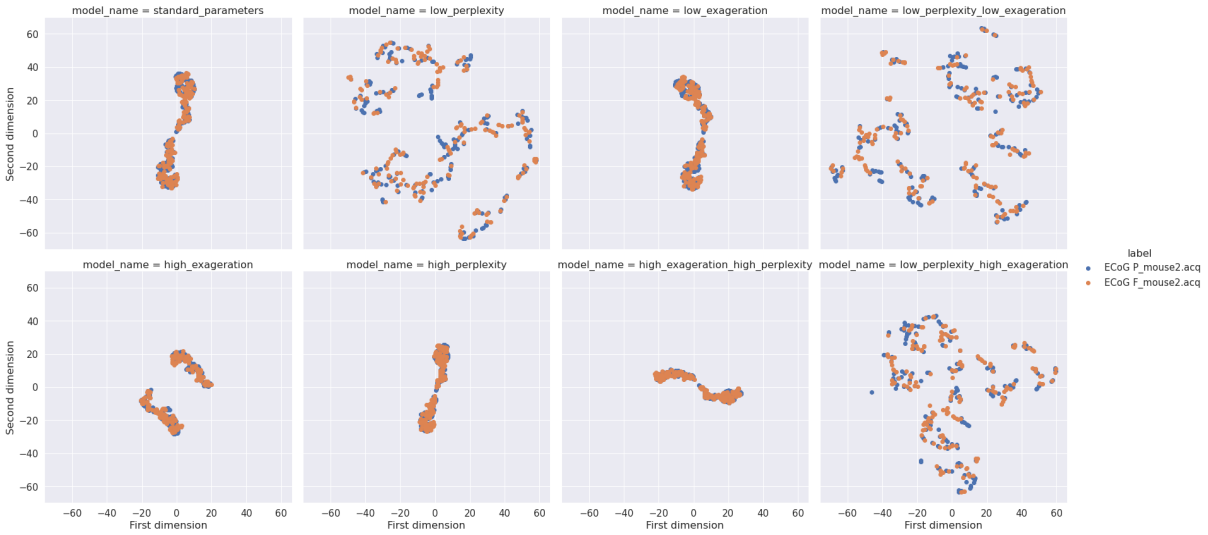

Figura 4.2: Tabloul lobiilor frontali și parietali ai celui de-al doilea șoarece

4.3 Lobii frontali și parietali ai celui de-al treilea șoarece(mouse3.acq)

TODO

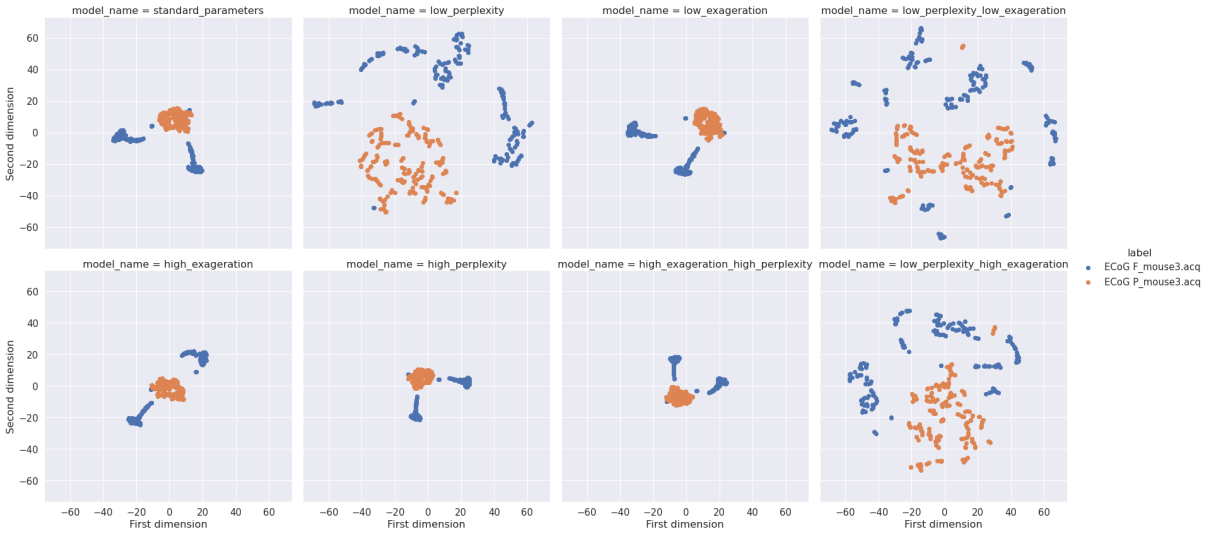

Figura 4.3: Tabloul lobiilor frontali și parietali ai celui de-al treilea șoarece

4.4 Lobii frontali și parietali ai celui de-al patrulea șoarece(mouse4.acq)

TODO

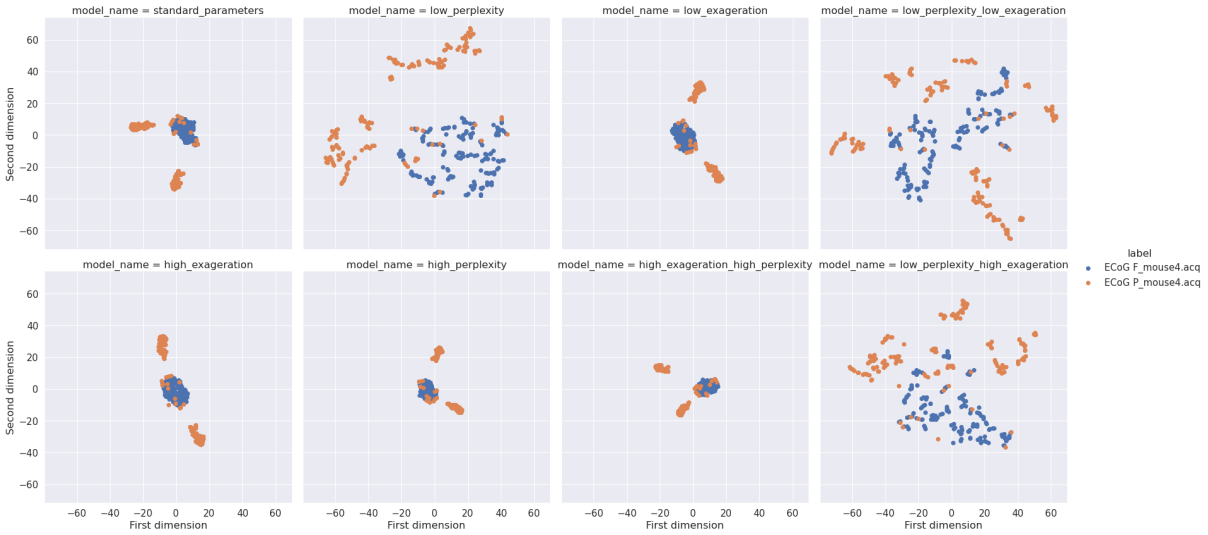

Figura 4.4: Tabloul lobiilor frontali și parietali ai celui de-al patrulea șoarece

4.5 Lobul frontal pentru toți șoarecii

TODO

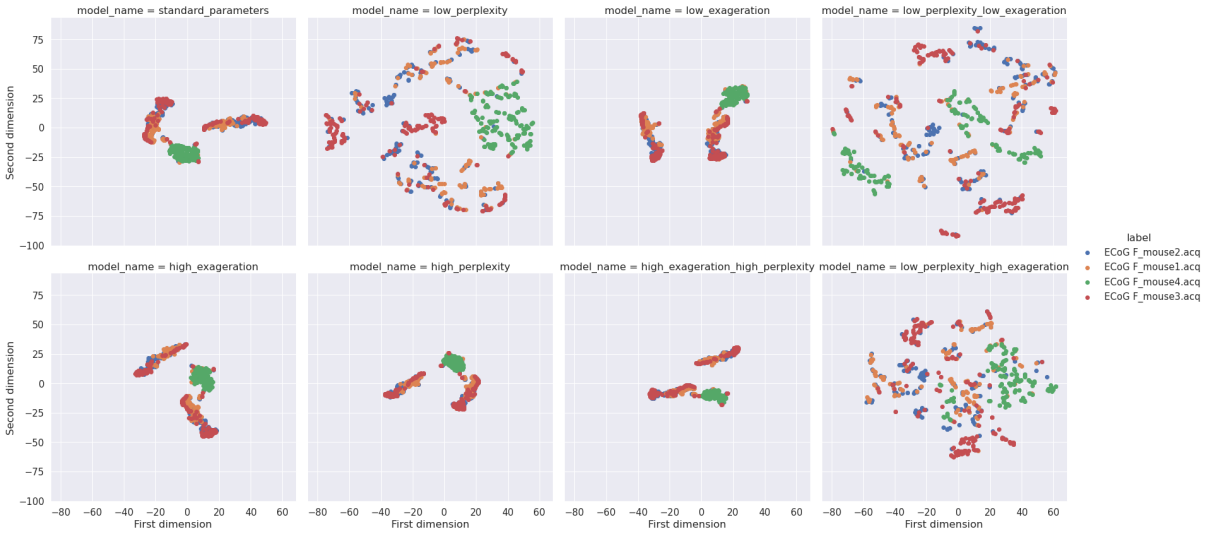

Figura 4.5: Tabloul lobiilor frontali ai tuturor șoarecilor

4.6 Lobul parietal pentru toți șoarecii

TODO

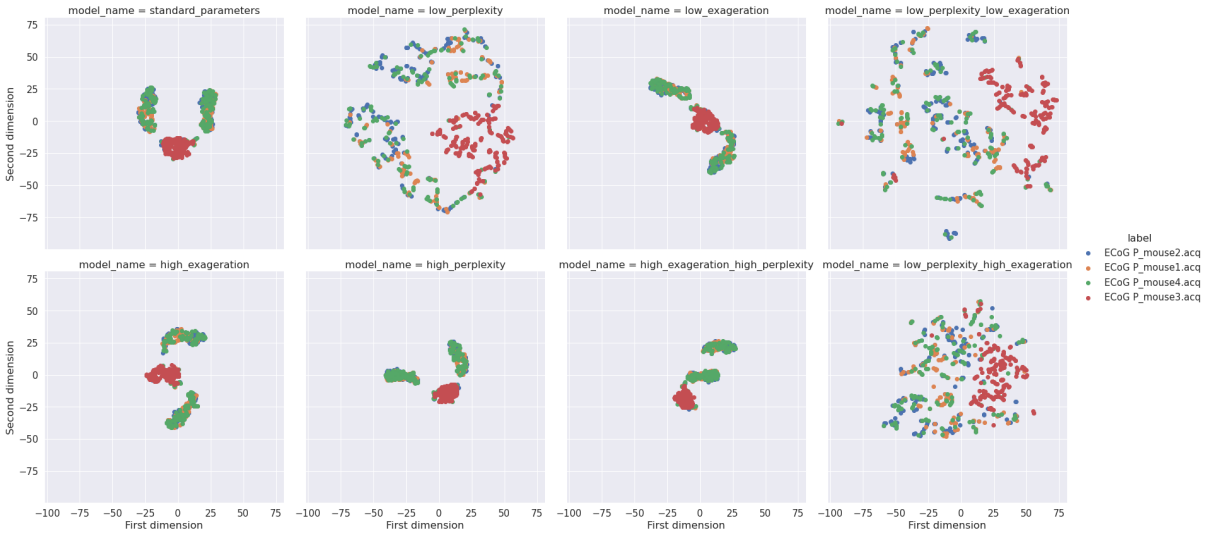

Figura 4.6: Tabloul lobiilor parietali ai tuturor șoarecilor

4.7 Lobii frontali și parietali pentru toți șoarecii

TODO

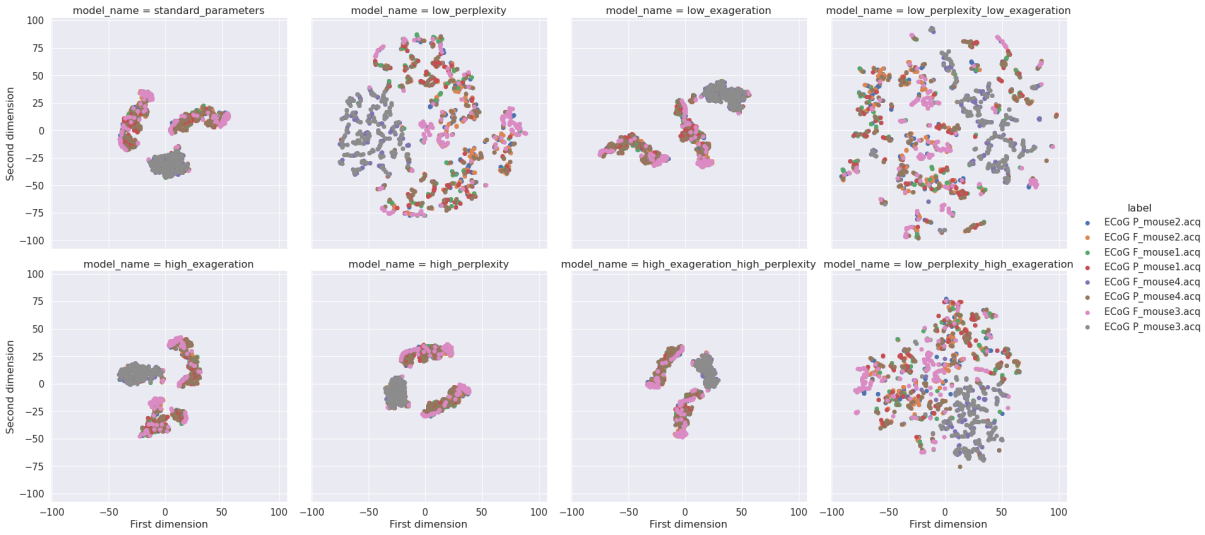

Figura 4.7: Tabloul lobiilor frontali și parietali ai tuturor șoarecilor

Capitolul 5

Viitor

5.1 Comunicare peste un protocol de comunicație

În forma curentă sistemul se comportă ca o bibliotecă de python. Din acest punct de vedere interacțiunea cu el este greoaie. Un utilizator ar trebui să descarce fișierele de cod pe stația sa și să folosească sistemul conform protocolului descris. În această situație, propun perspectiva comunicării peste un protocol din internet ca HTTP (Hypertext Transfer Protocol), un protocol arhicunoscut și ușor de implementat de către aplicațiile client.

De asemenea, persistența datelor este un element de mare importanță pentru aplicațiile care încarcă și procesează date, cum este și sistemul de față. În acest moment, sistemul nu persistă datele, ci le păstrează doar în memorie. Din această cauză, datele trebuie încărcate în sistem la fiecare pornire. O tehnologie de baze de date poate și ar trebui uzitată pentru persistența datelor încărcate în sistem.

5.2 Interfață grafică

În Secțiunea 5.1 am propus o nouă formă a sistemului, însă chiar și în această situație un utilizator atehnic nu ar putea să proceseze electroencefalogramele. El are nevoie de o interfață grafică care să translateze acțiunile sale (clicuri, apăsări de taste) în apeluri HTTP pe care sistemul să le înțeleaga. Îmi imaginez că această interfață ar fi potrivit să funcționeze în interiorul unei aplicații browser pentru înlesnirea comunicării cu sistemul, direct din internet, de pe orice stație pe care funcționează o astfel de aplicație browser.

Appendix A

Configurații de execuție

A.1 Configurația standard de execuție

```
{
  "preprocessing_pipeline_stages": [
    {
      "stage_name": "SplitInWindows",
      "constructor_kwargs": {
        "window_size": 1000
      }
    }
  ],
  "processing_pipeline_stages": [
    {
      "stage_name": "FeatureExtractor",
      "constructor_kwargs": {
        "features": [
          "alpha_spectrum_coeffiecients_sum",
          "beta_spectrum_coeffiecients_sum",
          "low_gamma_spectrum_coeffiecients_sum",
          "high_gamma_spectrum_coeffiecients_sum",
          "delta_spectrum_coeffiecients_sum",
          "theta_spectrum_coeffiecients_sum",
          "mean",
          "standard_deviation"
        ]
      }
    }
  ],
  "t_sne_models": [
    {
      "model_name": "standard_parameters",
      "parameters": {
        "n_jobs": 6,
        "learning_rate": "auto",
        "init": "random",
        "n_components": 2,
        "early_exaggeration": 12,
```

```
        "perplexity": 30
      }
    },
    {
      "model_name": "low_perplexity",
      "parameters": {
        "n_jobs": 6,
        "learning_rate": "auto",
        "init": "random",
        "n_components": 2,
        "early_exaggeration": 12,
        "perplexity": 5
      }
    },
    {
      "model_name": "low_exageration",
      "parameters": {
        "n_jobs": 6,
        "learning_rate": "auto",
        "init": "random",
        "n_components": 2,
        "early_exaggeration": 1,
        "perplexity": 30
      }
    },
    {
      "model_name": "low_perplexity_low_exageration",
      "parameters": {
        "n_jobs": 6,
        "learning_rate": "auto",
        "init": "random",
        "n_components": 2,
        "early_exaggeration": 1,
        "perplexity": 5
      }
    },
    {
      "model_name": "high_exageration",
      "parameters": {
        "n_jobs": 6,
        "learning_rate": "auto",
        "init": "random",
        "n_components": 2,
        "early_exaggeration": 300,
        "perplexity": 30
      }
    },
    {
      "model_name": "high_perplexity",
      "parameters": {
        "n_jobs": 6,
        "learning_rate": "auto",
        "init": "random",
```

```
89          "n_components": 2,
90          "early_exaggeration": 12,
91          "perplexity": 40
92        }
93      },
94      {
95        "model_name": "high_exageration_high_perplexity",
96        "parameters": {
97          "n_jobs": 6,
98          "learning_rate": "auto",
99          "init": "random",
100         "n_components": 2,
101         "early_exaggeration": 300,
102         "perplexity": 40
103       }
104     },
105     {
106       "model_name": "low_perplexity_high_exageration",
107       "parameters": {
108         "n_jobs": 6,
109         "learning_rate": "auto",
110         "init": "random",
111         "n_components": 2,
112         "early_exaggeration": 300,
113         "perplexity": 5
114       }
115     }
116   ]
117 }
```

Listarea A.1: Configurația standard de execuție

Bibliografie

[1] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(11), 2008.