Universitatea POLITEHNICA din București

Facultatea de Automatică și Calculatoare,
Departamentul de Calculatoare

LUCRARE DE DIPLOMĂ

Sistem informatic pentru vizualizarea electroencefalogramelor cu ajutorul inteligenței artificiale

Autor:
Dochian Alexandru Adrian

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

București, Iulie 2022

Această lucrare nu ar fi fost posibilă fără ajutorul catedrei de fiziologie din cadrul Facultatății 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, Doamnei sl. dr. ing. Irina Andra Tache, Domnului sl. dr. ing. Mihai Trăscău și Domnului sl. dr. ing. Alexandru Șorici pentru timpul, energia și cunoștiințele dedicate acestei lucrări.

Compendiu

În lucrare de față este prezentat un sistem informatic care oferă o nouă perspectivă asupra analizei electroencefalogramelor.

Electroencefalogramele sunt semnale care poartă informații despre evoluția potențialui electric într-un punct ales din cortexul subiectului. Din perspectiva sistemului, ele sunt simple serii de timp. Sistemul procesează electroencefalogramele alese ca intrare și le proiectează ca puncte într-un spațiu bidimensional.

În calea către rezultat, informațiile inițiale trec prin fluxuri configurabile de preprocesare și procesare. Fiecare flux este la rândul său format dintr-un număr variabil de stagii procesatoare.

Fluxul de preprocesare modifică semnalele doar în domeniul timp. Stagiul standard din flux folosit în acest sistem împarte semnalele în ferestre pentru a produce mai multe exemple dintr-o clasă.

În fluxul de procesare sunt folosite stagii care extrag caracteristici din domeniile timp și frecvență.

Ultimul pas al execuției îl constituie execuțiea algoritmului t-SNE pe datele procesate. t-Distributed Stochastic Neighbor Embedding (t-SNE) este un algoritm de învățare nesupervizată care reduce datelor de intrare de dimensiuni mari la două dimensiuni încercând să păstreze similaritățile din dimensiunile superioare.

Este cu putință ca tablourile generate de sistem să releve interacțiuni interesante între intrări și astfel să devină bunuri de valoare pentru analiza electroencefalografică.

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.

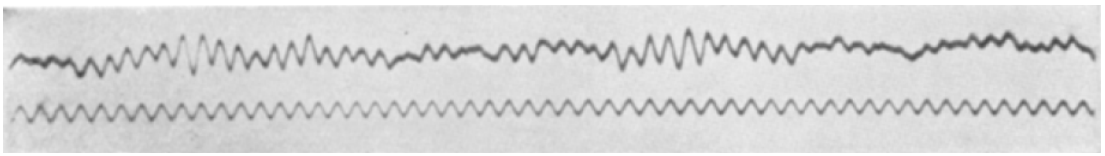

Figura 1.1: Primul EEG uman achiziționat de către Hans Berger în 1924. Semnalele din imagine sunt eșantionate la o frecvență de 10 Hz.

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. Electrozi au fost plasați direct pe cortex în zonele corespunzătoare lobilor frontali și parietali și anestezice au fost administrate direct în creier prin injecție. Patru experimente au fost efectuate pe șobolani diferiți astfel obținându-se câte o electrocorticogramă pentru fiecare experiment.

De-a lungul acestei lucrări experimentele vor fi recunoscute prin fișierele rezultate corespunzătoare: "mouse1.acq", "mouse2.acq", "mouse3.acq" și "mouse4.acq".

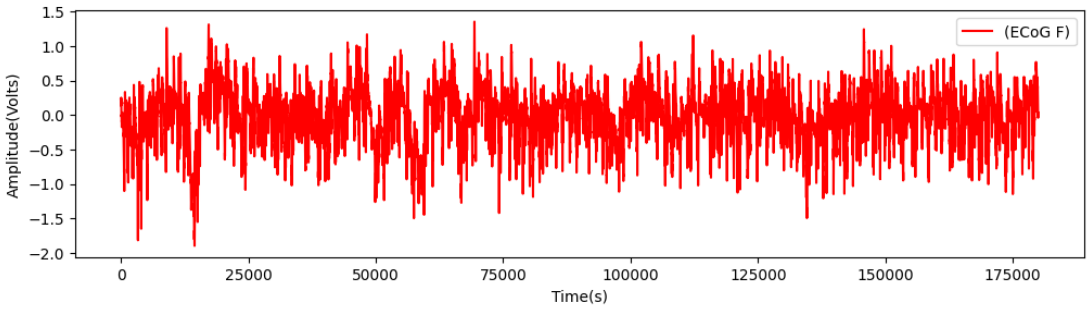

Figura 1.2: Electrocorticograma lobului frontal al șobolanului 2 de-a lungul a aproximativ 180 secunde

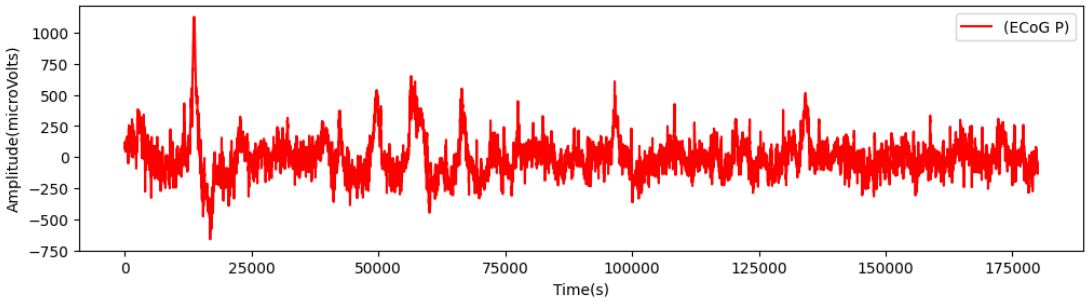

Figura 1.3: Electrocorticograma lobului parietal al șobolanului 2 de-a lungul a aproximativ 180 secunde

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.

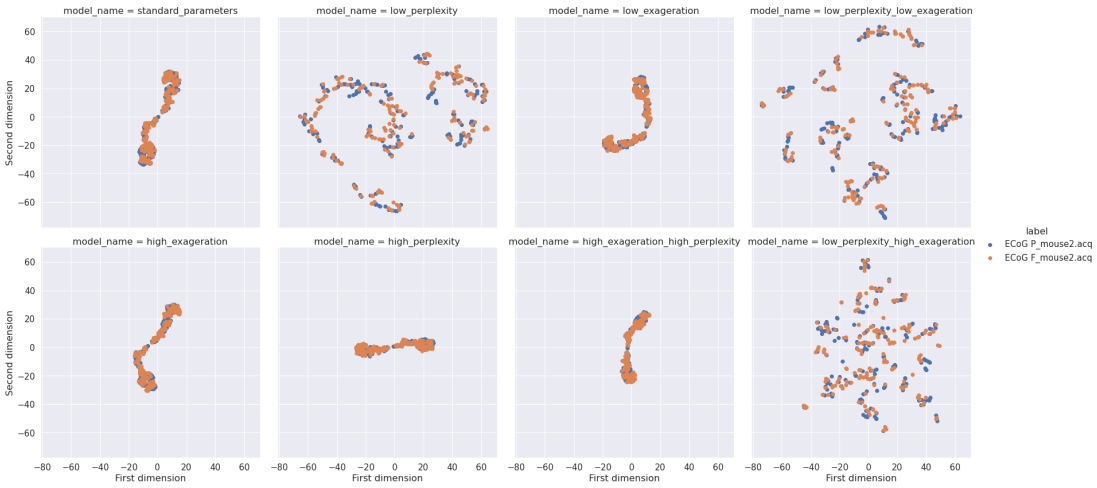

Figura 1.4: Excuția sistemului pe electroencefalogramele prezentate în Secțiunea 1.2.1

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

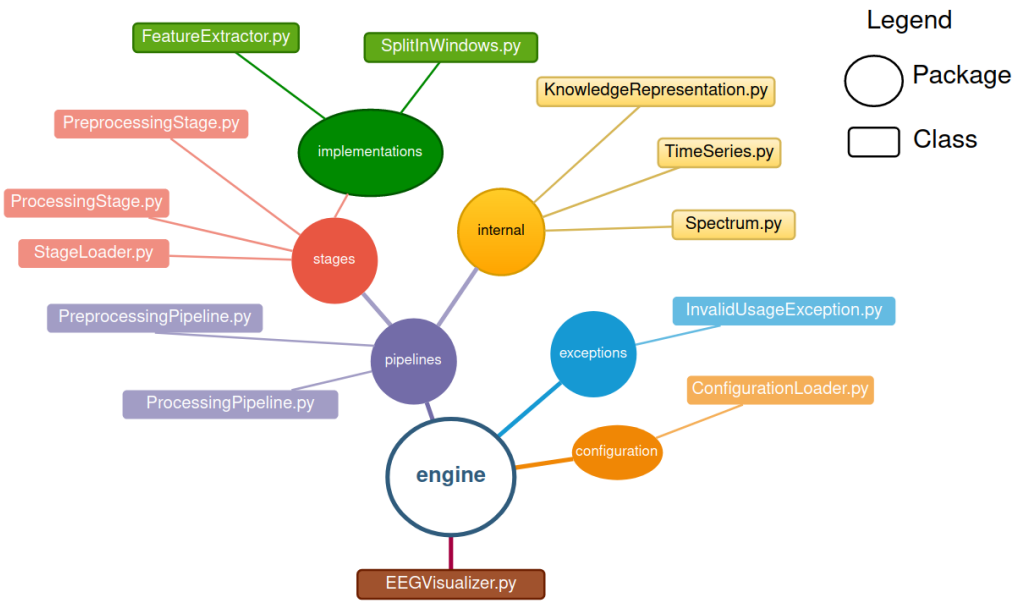

Figura 2.1: Structura codului

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 mouseX.acq 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

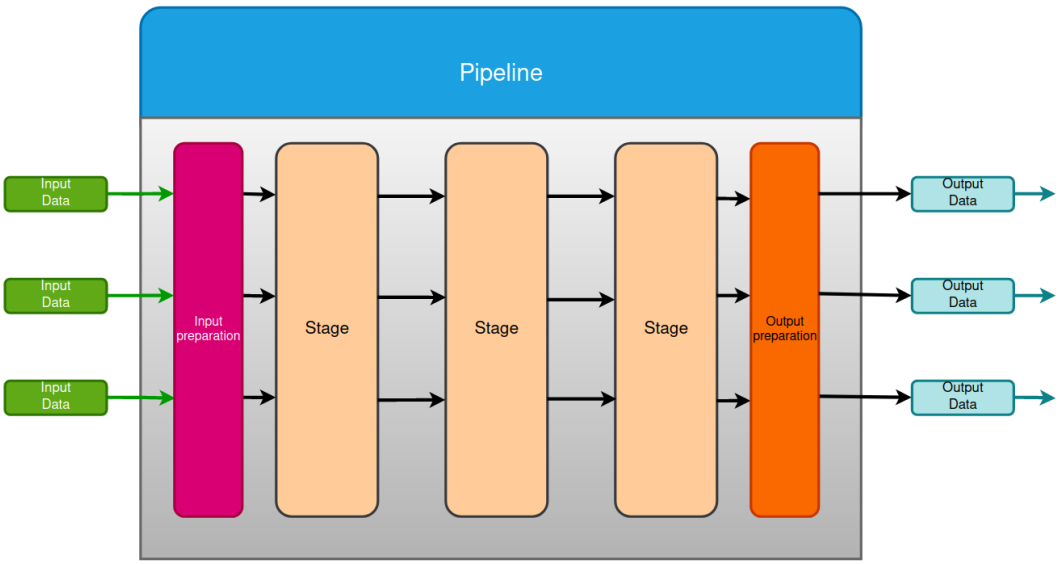

Figura 2.2: Perspectivă globală asupra funcționalității unui flux

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**.

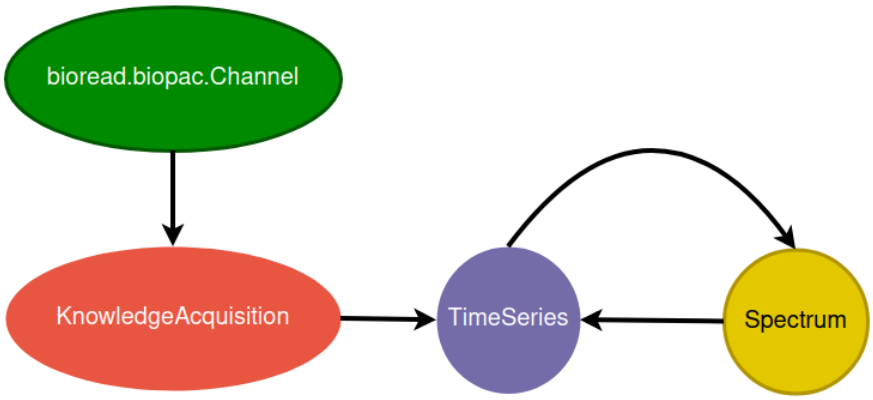

Figura 2.3: Relaționarea obiectelor din pachetul **internal**

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 prezenta și explicita aplicația din diverse unghiuri și nivele de adâncime.

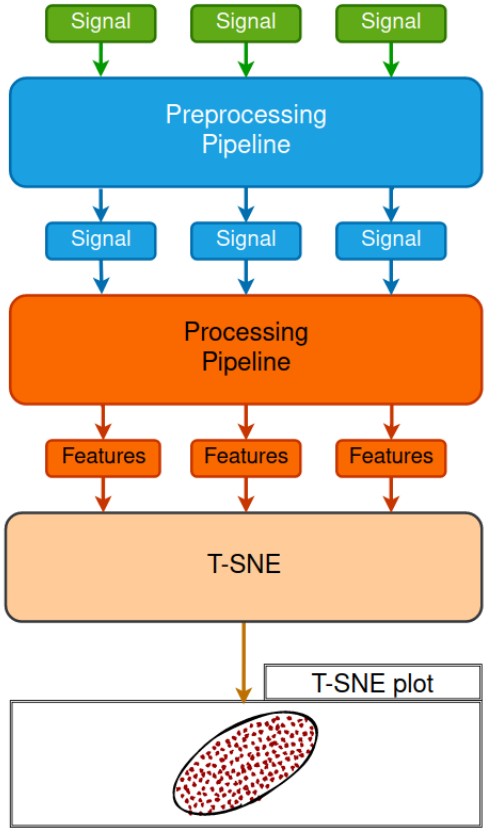

Figura 3.1: Structura execuției

3.1 Inițializarea sistemului

3.2 Fluxul de preprocesare

3.3 Fluxul de procesare

3.4 Execuția algoritmului T-SNE

Capitolul 4

Rezultate și Concluzii

TODO:

Capitolul 5

Lucrări similare

TODO:

Capitolul 6

Viitor

TODO:

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.