
UNIVERSITATEA DIN BUCUREȘTI

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

SPECIALIZAREA INFORMATICĂ


Lucrare de licență

# CLASIFICARE DE ACCENT UTILIZAND RETELE NEURALE SPIKE

Absolvent
Moarcăs Cosmin-Ionuț

Coordonator științific
Conf. dr. Cristian Rusu

București, iunie 2024

**Rezumat**

În ultimii ani, domeniul inteligenței artificiale a cunoscut o creștere semnificativă. Odată cu integrarea sistemelor de inteligență artificială în diverse domenii, consumul de energie necesar antrenării acestora a crescut considerabil.

Pentru a aborda problema consumului ridicat de energie, companii precum Intel și CyberSwarm din România au dezvoltat cipuri neuromorfice, menite să eficientizeze sistemele convenționale printr-o arhitectură inovatoare, inspirată din principiile de funcționare ale creierului. Aceste cipuri facilitează antrenarea unor rețele neuronale diferite de cele artificiale, numite rețele neuronale spike.

Această lucrare își propune dezvoltarea și evaluarea rețelelor neuronale spike în clasificarea accentelor în limba engleză. Comparând rețelele neuronale adânci cu cele spike, am demonstrat că ambele ating performanțe similare, însă cele din urmă oferă avantajul unui consum redus de energie în timpul antrenării. Modelele de clasificare a accentelor pot îmbunătăți substanțial performanța sistemelor de recunoaștere vocală precum Alexa și Siri. Prin detectarea accentului, se pot dezvolta sisteme specializate pentru fiecare tip de accent, facilitând utilizarea acestor tehnologii de către persoanele cu un accent pronunțat.

**Abstract**

In recent years, the field of artificial intelligence has experienced significant growth. With the integration of AI systems across various domains, the energy consumption required for training these systems has also increased considerably.

To address the issue of high energy consumption, companies such as Intel and CyberSwarm from Romania have developed neuromorphic chips designed to enhance the efficiency of conventional systems through an innovative architecture inspired by the functioning principles of the brain. These chips facilitate the training of neural networks that are different from traditional artificial ones, called spiking neural networks.

This paper aims to develop and evaluate spiking neural networks for accent classification in the English language. By comparing deep neural networks with spiking ones, we demonstrated that both achieve similar performance levels, but the latter offer the advantage of reduced energy consumption during training. Accent classification models can substantially improve the performance of voice recognition systems like Alexa and Siri. By detecting accents, specialized systems can be developed for each accent type, making these technologies more accessible to individuals with pronounced accents.

# Cuprins

# Capitolul 1

# Introducere

Link GitHub: https://github.com/Moarcas/licenta

## 1.1 Contextul și importanța clasificării accentului

Accentul reprezintă modul specific în care o persoană pronunță cuvintele unei limbi. Fiecare vorbitor al unei limbi are un accent distinct, influențat de factori geografici, culturali și sociali. La nivel mondial, există aproximativ 160 de accente diferite, reflectând diversitatea lingvistică și culturală globală.

Studierea accentului este deosebit de importantă datorită legăturilor sale strânse cu multiple probleme sociale, precum acceptarea vorbitorului într-o comunitate și percepția acestuia în cadrul unei anumite clase sociale. De exemplu, accentul poate influența modul în care o persoană este percepută din punct de vedere profesional și social, afectându-i inclusiv oportunitățile de angajare și integrarea socială.

Un caz particular relevant este cel al tinerilor din India, unde engleza a devenit o limbă frecvent utilizată datorită prezenței sale în sistemul educațional. Majoritatea tinerilor vorbesc cel puțin două limbi: una dintre cele 22 de limbi oficiale ale Indiei și limba engleză. Această bilingvism larg răspândit, combinat cu diversitatea accentelor, face ca engleza vorbită în India să fie o resursă valoroasă pentru studierea accentului.

Analizarea accentului în acest context poate contribui semnificativ la dezvoltarea unor sisteme de recunoaștere vocală mai precise și adaptate, care să funcționeze eficient într-un mediu multilingv și multicultural. De asemenea, poate facilita îmbunătățirea asistenților vocali, a traducerilor automate și a altor aplicații ce depind de procesarea limbajului natural.

Prin urmare, clasificarea accentului nu este doar o problemă tehnică, ci și una cu implicații sociale profunde, având potențialul de a îmbunătăți interacțiunile dintre oameni și tehnologie într-un mod mai incluziv.

## 1.2 Obiectivele lucrării

Obiectivul principal al acestei lucrări este de a demonstra eficiența energetică a rețelelor neurale spike în clasificarea a nouă accente distincte. În detaliu, acest obiectiv se va atinge prin:

1. **Preprocesarea datelor audio:** vom prelucra datele audio prin diverse tehnici pentru a pregăti setul de date utilizat în antrenarea rețelelor neurale.

2. **Dezvoltarea unei rețele neurale spike:** vom antrena rețele neurale spike utilizând biblioteca snnTorch din Python.

3. **Dezvoltarea unei rețele neurale artificiale:** vom antrena rețele neurale artificiale tradiționale utilizând biblioteca Torch din Python. Aceste rețele vor servi ca punct de referință pentru comparația performanțelor și consumului energetic cu rețelele neurale spike.

4. **Evaluarea performanței pentru fiecare rețea:** vom analiza performantele fiecarei retea utilizand metrici precum: acuratetea, matricea de confuzie, scorul F1. Aceasta va permite o evaluare detaliată a capacității fiecărei rețele de a clasifica corect accentele.

5. **Analiza performanței energetice:** Vom compara consumul de energie al rețelelor neurale spike cu cel al rețelelor neurale tradiționale. Această comparație va include măsurători detaliate ale consumului energetic în timpul antrenării și inferenței, evidențiind avantajele potențiale ale utilizării rețelelor neurale spike din perspectiva eficienței energetice.

Prin atingerea acestor obiective specifice, lucrarea își propune să demonstreze că rețelele neurale spike nu doar că pot fi utilizate eficient pentru clasificarea accentelor, dar și că acestea oferă avantaje semnificative în ceea ce privește eficiența energetică, aspect esențial în contextul dezvoltării unor sisteme de inteligență artificială sustenabile și eficiente.

## 1.3 Structura lucrării

Structura lucrării este următoarea: capitolul "Preliminarii" furnizează informații esențiale despre rețelele neurale artificiale, spike și datele audio. Capitolul "Fundamentele Teoretice ale Rețelelor Neurale Spike" se concentrează pe teoria din spatele acestor rețele. Capitolul "Setul de Date" introduce și detaliază datele utilizate pentru antrenare și modurile în care acestea au fost procesate. În capitolul "Modele Dezvoltate", sunt prezentate și evaluate atât modelele de rețele neurale artificiale, cât și cele spike dezvoltate, analizându-se performanța și consumul de energie. Capitolul "Concluzii" sintetizează rezultatele și concluziile obținute în cadrul lucrării.

# Capitolul 2

# Preliminarii

## 2.1 Inteligenta artificiala

Inteligența artificială (IA) este un domeniu al informaticii care se ocupă cu crearea de sisteme capabile să execute sarcini care, în mod normal, necesită inteligență umană. Acest domeniu a apărut în 1956 și a evoluat prin multiple perioade de optimism, urmate de perioade de scepticism și lipsă de fonduri, cunoscute sub denumirea de "IA winter".

Interesul pentru inteligența artificială a crescut semnificativ în 2012, când tehnicile de învățare profundă (deep learning) s- au dovedit a fi superioare altor metode de învățare automată. Un alt moment important a fost în 2017, odată cu apariția transformatoarelor (transformers), care au revoluționat procesarea limbajului natural și alte domenii ale IA.

O ramură importantă a inteligenței artificiale este învățarea automată, care se împarte în două subdomenii principale: învățarea supravegheată și învățarea nesupravegheată. Clasificarea accentelor se realizează prin învățarea supravegheată, deoarece datele audio sunt etichetate. Astfel, această metodă va fi utilizată și prezentată în lucrare.

## 2.2 Noțiuni de bază despre rețelele neurale

Rețelele neurale sunt inspirate din modul de funcționare al creierului uman. Acestea constau în modele care conțin neuroni artificiali interconectați prin legături, numite muchii. Neuronul artificial, cunoscut sub numele de perceptron, este o versiune simplificată a neuronului biologic, iar muchiile dintre aceștia imită sinapsele din creier.

Datorită straturilor multiple de perceptroni, rețelele neurale profunde (deep neural networks) și-au demonstrat performanța prin obținerea unor rezultate remarcabile în diverse domenii, cum ar fi viziunea computerizată (computer vision), recunoașterea vorbirii și procesarea limbajului natural.

În contextul clasificării audio, **rețelele neurale convoluționale** (CNN) oferă rezultate deosebit de bune. Acestea utilizează straturi convoluționale care permit rețelei să

învețe relațiile spațiale și temporale ale datelor audio.

## 2.3 Consumul ridicat de energie al rețelelor neurale

Este important să luăm în considerare energia necesară pentru antrenarea modelelor de inteligență artificială și să dezvoltăm tehnologii sustenabile și eficiente din punct de vedere computațional. În 2024, Stanford Institute for Human-Centered Artificial Intelligence (HAI) a publicat raportul Artificial Intelligence Index, estimând costul antrenării modelelor Chat-GPT și Google Gemini Ultra la 78 de milioane, respectiv 191 de milioane de dolari [1]. De asemenea, Sam Altman, directorul executiv al OpenAI, a menționat că progresul în domeniul inteligenței artificiale nu va depinde de dezvoltarea unor modele mai mari, ci de noi metode de antrenare [2].

#### Limitarea arhitecturii Von Neumann

Calculatoarele personale moderne utilizează arhitectura **Von Neumann** (Figura 2.1). În această arhitectură, procesorul este separat de memorie, iar transferul datelor între cele două componente devine un blocaj major atunci când dorim să antrenăm rețele neurale.

Este contraintuitiv faptul că implementăm concepte de inteligență artificială pe arhitecturi care nu seamănă deloc cu modul de funcționare al creierului.

#### Hardware neuromorfic

Creierul nostru consumă între 12 și 20 de wați, în timp ce un calculator personal consumă aproximativ 175 de wați, iar acceleratoarele, precum Nvidia H100, au un consum cuprins între 300 și 700 de wați [3]. Diferența majora de consum ridică întrebarea dacă ne putem inspira din complexitatea arhitecturală a creierului pentru a îmbunătăți eficiența acestor sisteme în domeniul învățării automate.

Companii precum Intel, IBM și Cyber Swarm din România încearcă să răspundă la aceste întrebări prin dezvoltarea de cipuri neuromorfice, cum ar fi Loihi 2, TrueNorth și Swarm Nervous System. Aceste cipuri sunt proiectate fundamental diferit, fiind inspirate de modul de funcționare al creierului. Scopul lor este de a dezvolta componente hardware optimizate pentru modele de inteligență artificială, obținând astfel o eficiență mult superioară, măsurată în câteva ordine de mărime, față de sistemele convenționale.

## 2.4 Introducere în rețelele neurale spike

Pentru a obține modele de inteligență artificială eficiente, este necesară antrenarea unor rețele neurale speciale pe hardware neuromorfic. Aceste rețele sunt cunoscute sub denumirea de rețele neurale spike.

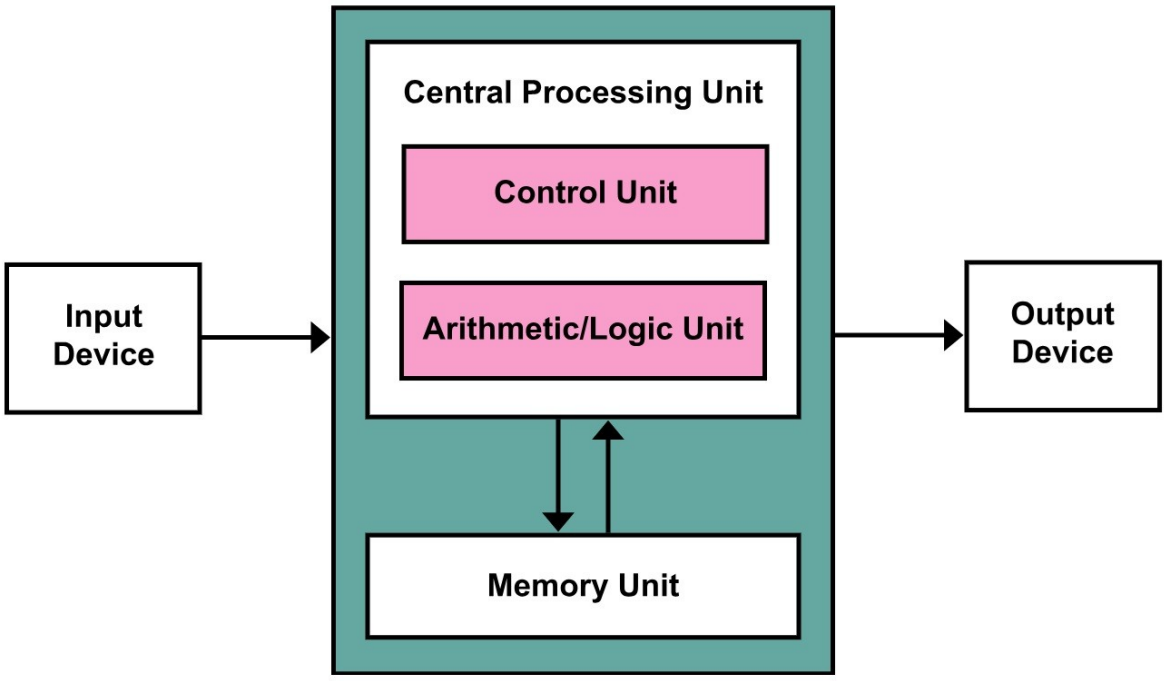


Figura 2.1: Arhitectura Von Neumann - Sursa [4]

Principala diferență dintre rețelele neurale tradiționale și cele spike constă în tipul neuronilor utilizați și în modul de comunicare dintre aceștia. În rețelele neurale tradiționale, informația este codificată prin magnitudinea valorilor din rețea, pe când în rețelele spike, informația este codificată în timp, prin succesiunea impulsurilor neurale. Acești neuroni noi introduc astfel o dimensiune temporală în rețea, oferind un mod de procesare mai apropiat de cel al creierului uman.

Pentru a beneficia de avantajele rețelelor neurale spike, este esențial ca acestea să ruleze pe un hardware neuromorfic. În această lucrare, vom simula antrenarea și inferența acestor rețele pe un calculator personal, utilizând biblioteca snnTorch din Python.

## 2.5 Noțiuni despre datele audio

Sunetul este o undă care se propagă prin medii precum aerul sau apa. Acesta poate fi caracterizat prin două proprietăți principale: intensitate și înălțime. Înălțimea sunetului depinde de frecvența vibrațiilor particulelor din aer. Când particulele vibrează de multe ori pe secundă, se produce un sunet de înălțime ridicată (frecvență înaltă), iar când particulele vibrează de puține ori pe secundă, se produce un sunet de înălțime joasă (frecvență joasă). Urechea umană poate percepe sunete într-un interval de frecvențe cuprins între 20 și 20.000 Hz.

Pentru clasificarea sunetului, este esențial să eliminăm informațiile irelevante, cum ar fi zgomotul de fundal sau emoția. Astfel, trebuie să transformăm sunetul pentru a păstra

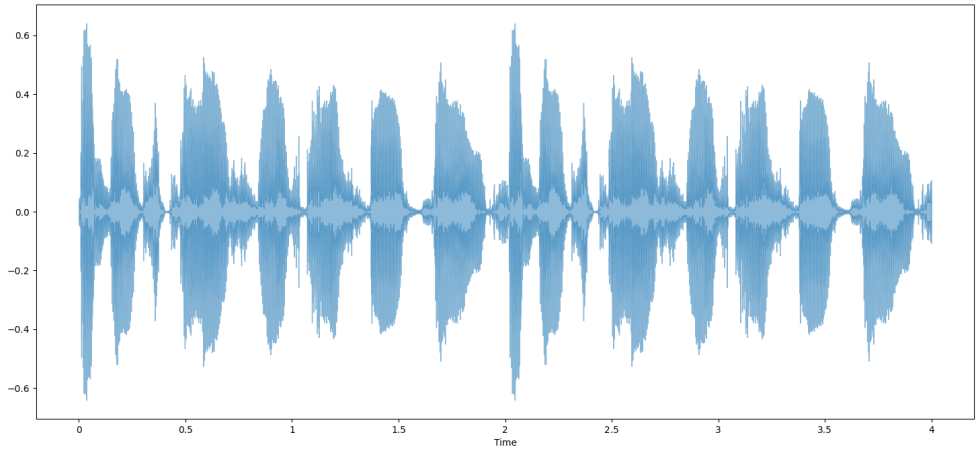


Figura 2.2: Amplitudinea unui semnal audio de 4 secunde

doar informațiile relevante pentru clasificare.

### 2.5.1 Amplitudinea

Amplitudinea reprezintă valoarea variațiilor presiunii aerului generate de undele sonore. O amplitudine mare corespunde unui sunet puternic, în timp ce o amplitudine mică corespunde unui sunet slab.

Pentru a reprezenta sunetul în format digital, microfonul măsoară presiunea aerului la o anumită frecvență de eșantionare. Aceasta se referă la numărul de măsurători efectuate pe secundă pentru a captura variațiile presiunii aerului. În Figura 2.2 este prezentat un semnal audio de 4 secunde, eșantionat cu o frecvență de 22.050 Hz. Acest semnal conține 88.200 de eșantioane, deoarece presiunea aerului a fost măsurată de 22.050 ori pe secundă. Pe axa Ox este reprezentat numărul eșantionului, iar pe axa Oy, valoarea amplitudinii fiecărui eșantion.

### 2.5.2 Spectograme

Spectrogramele reprezintă o modalitate de a vizualiza un sunet în funcție de frecvența sa. Cu ajutorul spectrogramelor, putem determina cum se modifică fiecare frecvență componentă a unui sunet în timp. Calcularea spectrogramelor este inspirată de modul în care funcționează urechea umană, în special melcul osos. În funcție de frecvențele prezente în sunet, melcul osos vibrează în locații diferite și transmite semnale electrice prin nervi, informând creierul despre prezența fiecărei frecvențe.

Spectograma afișează intensitatea diferitelor frecvențe ale unui semnal audio pe parcursul timpului, unde axa orizontală reprezintă timpul, axa verticală reprezintă frecvența, iar intensitatea fiecărei frecvențe este indicată printr-o scară de culori.

### 2.5.3 Spectograma Mel

Urechea umană percepe frecvențele sunetului în mod logaritmic. De exemplu, diferența dintre 50 Hz și 250 Hz ni se pare mai mare decât diferența dintre 1500 Hz și 1700 Hz, chiar dacă numeric aceste diferențe sunt egale. Spectogramele obișnuite nu reflectă această proprietate a auzului uman, astfel a fost dezvoltată scara Mel pentru a corecta această problemă.

Conversia frecvenței în scara Mel se realizează folosind următoarea formulă:

$$\text{mel}(f) = 2595 \cdot \log_{10}\left(1 + \frac{f}{700}\right) \tag{2.1}$$

unde:

- $\text{mel}(f)$ este frecvența pe scara Mel.
- $f$ este frecvența în Hz.

Conversia din scara Mel în frecvență se realizează folosind următoarea formulă:

$$\text{f}(mel) = 700\left(10^{\frac{\text{mel}}{2595}} - 1\right) \tag{2.2}$$

unde:

- $\text{f}(mel)$ este frecvența în Hz.
- $mel$ este frecvența pe scara Mel.

Pentru a calcula spectograma Mel sunt necesari urmatorii pasi: calcularea STFT, convertirea amplitudinii la decibeli si convertirea frecventei la scara Mel.

#### Pasul 1. Calcularea STFT (short-time Fourier transform)

Short-time Fourier transform modifică semnalul audio în domeniul timp-frecvență prin calcularea transformatei Fourier a unor intervale scurte și suprapuse din semnal. Acest proces se bazează pe presupunerea că semnalul nu se modifică semnificativ în intervale scurte de timp.

Dacă intervalul este prea mic, nu există suficiente valori pentru a obține o spectrogramă precisă. Pe de altă parte, dacă intervalul este prea mare, semnalul se modifică prea mult, ceea ce poate duce la pierderea informațiilor detaliate despre variațiile frecvențelor în timp. O diagrama a acestui proces poate fi vizualizata in Figura 2.3

#### Pasul 2. Convertirea amplitudinii la decibeli

Pentru a reprezenta spectrograma într-un mod care reflectă mai bine percepția umană a sunetului, amplitudinile trebuie convertite în decibeli (dB). Decibelii sunt o unitate

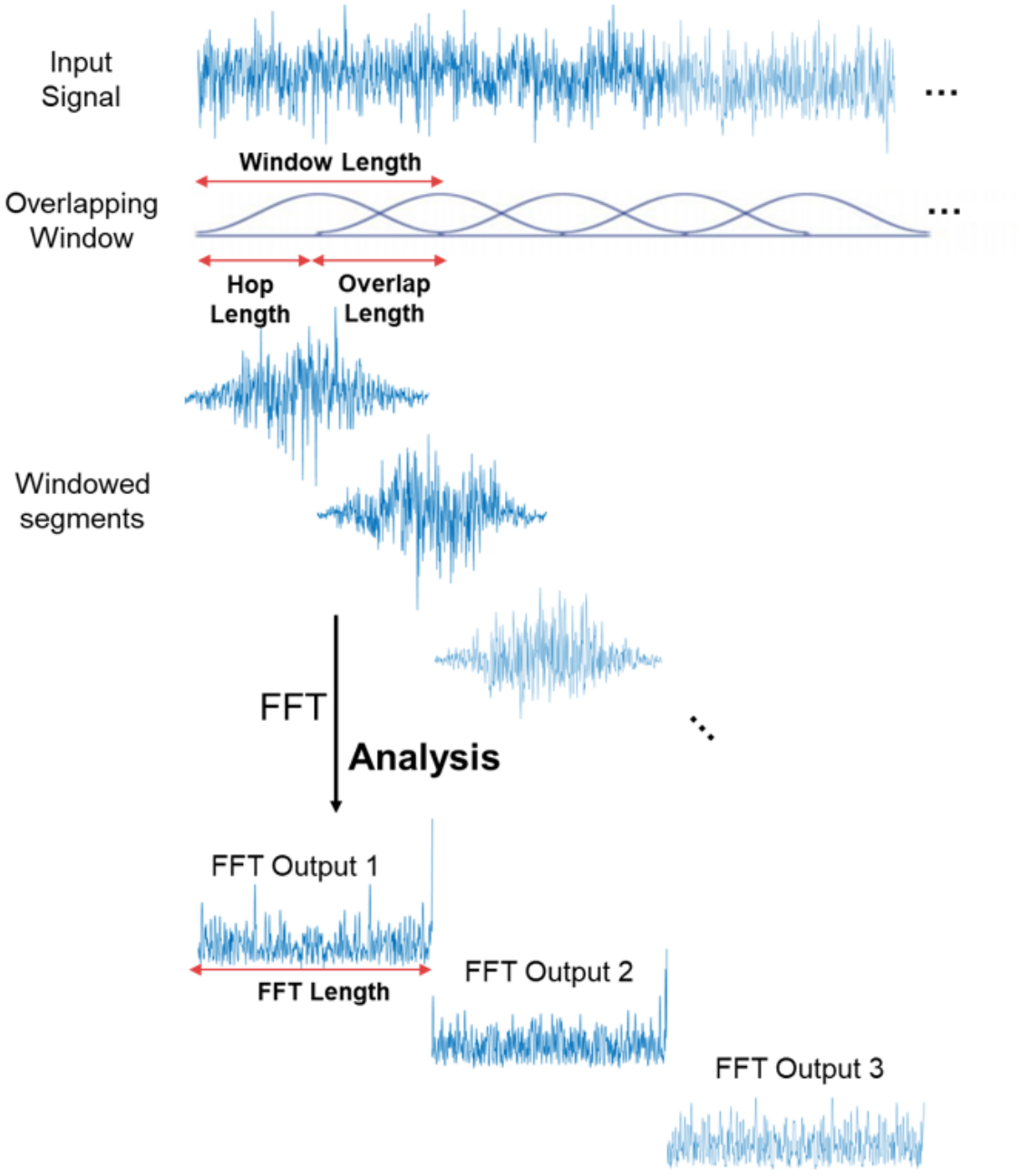


Figura 2.3: STFT - Sursa [5]

logaritmică care măsoară intensitatea sunetului, facilitând compararea diferențelor mari de amplitudine.

Conversia amplitudinii $A$ la decibeli se realizează folosind următoarea formulă:

$$A_{dB} = 20 \cdot \log_{10}(A)$$

unde:

- $A_{dB}$ este amplitudinea în decibeli.
- $A$ este amplitudinea originală.

Această conversie este esențială pentru a obține o reprezentare precisă a spectrogramelor, deoarece urechea umană nu percepe sunetele în mod liniar.

**Pasul 3. Convertirea frecvenței la scara Mel**

Pentru a converti spectrograma la scara Mel sunt necesari următorii pași:

- Se selectează numărul de benzi Mel, notat $N$. Acest număr poate lua diverse valori, precum 20, 40 sau 128, în funcție de specificul problemei.
- Se convertește cea mai mică frecvență ($f_{\text{min}}$) și cea mai mare frecvență ($f_{\text{max}}$) în scala Mel folosind formula (2.1).
- Se generează $N$ valori egal distanțate în intervalul $[\text{mel}(f_{\text{min}}), \text{mel}(f_{\text{max}})]$.
- Se convertește cele $N$ valori înapoi în Hz folosind formula (2.2).
- Valorile sunt aproximative la cea mai apropiată frecvență.
- Pentru fiecare dintre cele $N$ frecvențe se aplică filtrul asociat.

În Figura 2.4, putem observa reprezentarea pașilor 2 și 3 care au ca scop găsirea celor $N$ frecvențe pe care urechea umană le percepe ca fiind egal distanțate. Se alege frecvența cea mai mică $f_1$ și frecvența cea mai mare $f_8$, apoi se calculează punctele $\text{mel}(f_1)$ și $\text{mel}(f_8)$. În intervalul $[\text{mel}(f_1), \text{mel}(f_8)]$ se generează 3 puncte: mel1, mel2 și mel3, astfel încât distanța dintre ele este egală numeric ($D_5 = D_6 = D_7 = D_8$). Convertind aceste trei puncte înapoi în Hz, obținem frecvențe distanțate egal din perspectiva percepției auditive. Deși valorile numerice ale distanțelor $D_1$, $D_2$, $D_3$ și $D_4$ nu sunt identice, ele sunt percepute ca fiind egale datorită modului în care urechea noastră percepe sunetul.

În Figura 2.5 este prezentată o diagramă a filtrelor triunghiulare. Aceste filtre au rolul de a redistribui energia frecvențelor către cele trei benzi Mel. Valoarea pentru a treia bandă Mel se determină prin însumarea amplitudinilor frecvențelor ponderate de filtrul frecvenței f5:

$$A_{\text{Mel3}} = \sum_{i=1}^{8} A(f_i) \cdot \text{filtrul}_{\text{Mel3},i}$$

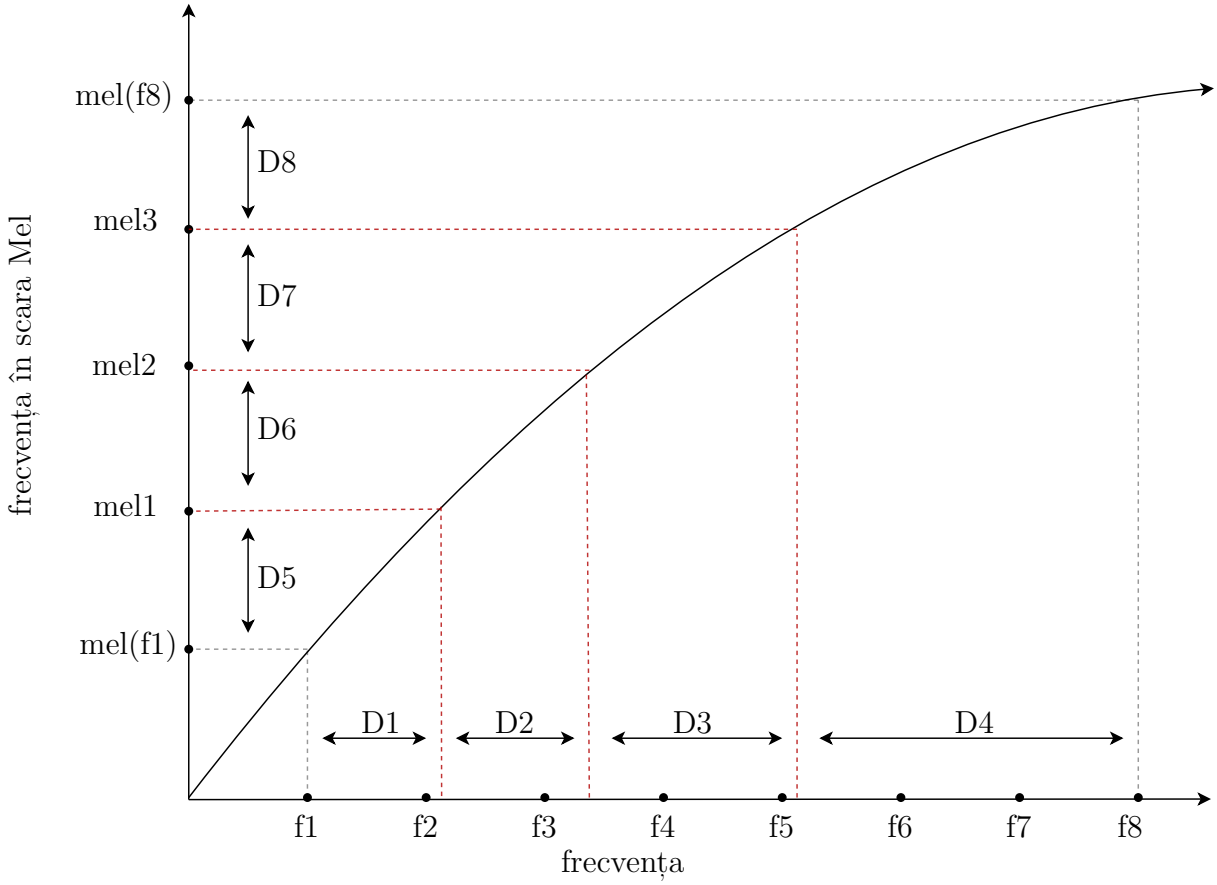


Figura 2.4: Converitrea frecvenței la scara Mel pentru N = 3

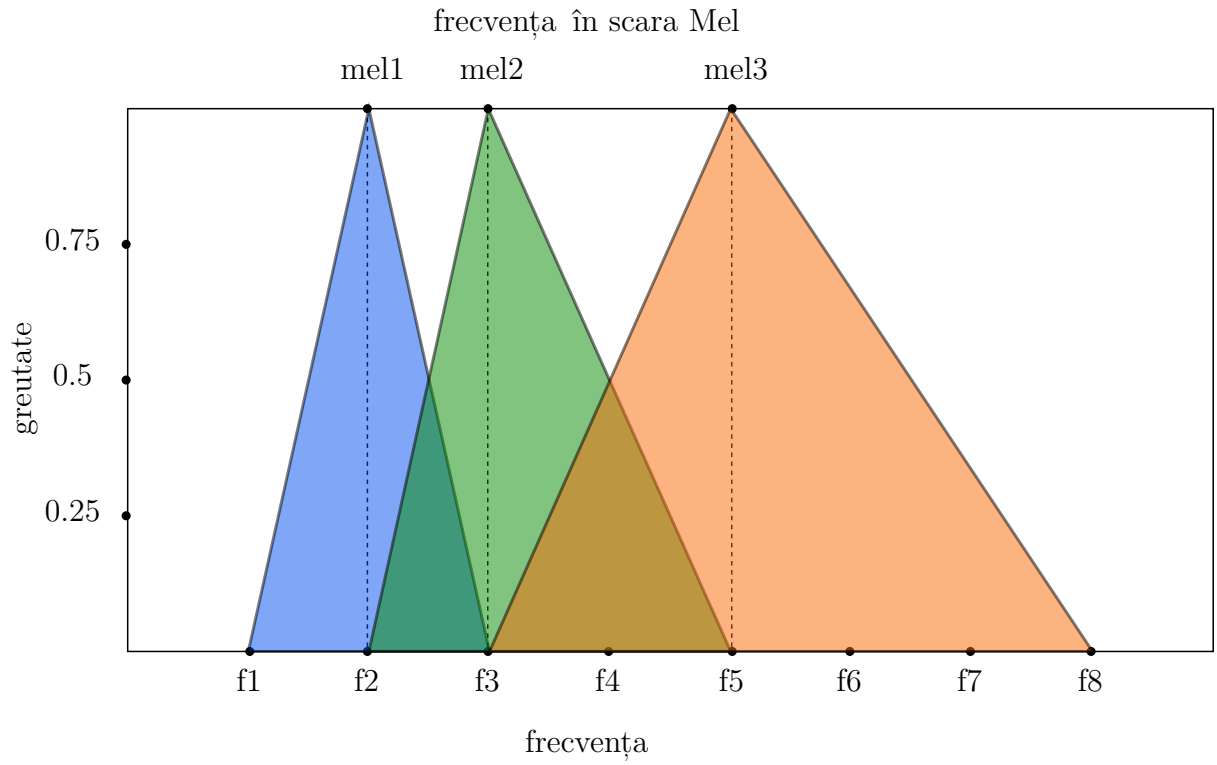


Figura 2.5: Filtre Mel

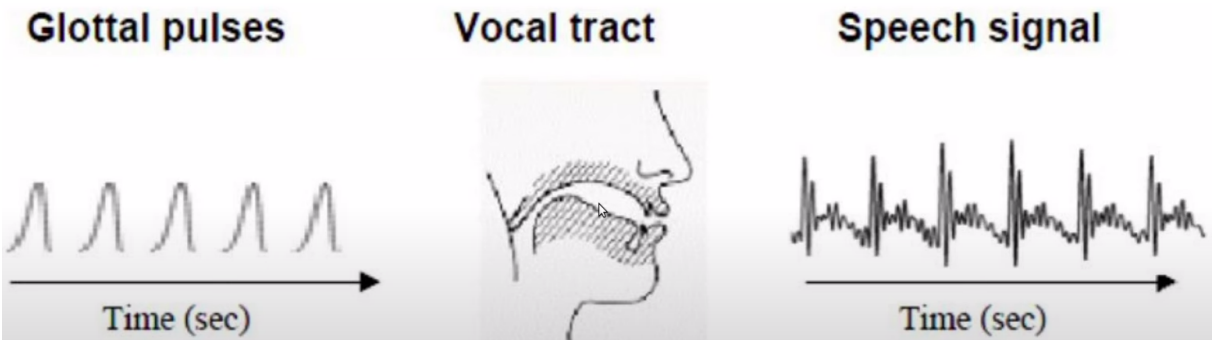


Figura 2.6: Procesul prin care producem sunete - Sursa [6]

### 2.5.4 Coeficienți cepstrali

Mel-Frequency Cepstral Coefficients (MFCC) au fost dezvoltați de o echipă de cercetători din cadrul MIT în anul 1960, inițial pentru studierea semnalelor seismice.

Coeficientii cepstrali sunt utilizați în prezent într-o varietate de aplicații, inclusiv prelucrarea semnalelor audio și învățarea automată.

Pentru a înțelege rolul acestora, este esențial să înțelegem modul în care funcționează vocea umană.

Când comunicăm, sunetul este rezultatul interacțiunii complexe dintre corzile vocale și cavitatea bucală. Corzile vocale generează un sunet inițial cu frecvență înaltă, care este apoi modelat de cavitatea bucală prin care trece. Această cavitate acționează ca un filtru, modificând sunetul produs de corzile vocale. Ideea principală este că sunetele sunt produse prin modelarea cavității bucale, ceea ce determină modificarea frecvențelor din componenta sunetului inițial.

Din această observație, reiese că sunetul produs de o persoană poate fi analizat și descompus în două componente distincte: sunetul inițial generat de corzile vocale și filtrul format de cavitatea bucală. Pentru sistemele de recunoaștere a vorbirii, informațiile relevante se găsesc în mod special în filtrul creat de cavitatea bucală. Sunetul inițial generat de corzile vocale, în sine, nu oferă informații esențiale pentru recunoașterea vorbirii, deoarece nu conține detalii specifice legate de conținutul mesajului transmis. Procesul prin care sunetul este produs poate fi vizualizat in Figura 2.6.

Coeficientii cepstrali sunt utili deoarece furnizează informații esențiale despre vorbitor și conținutul mesajului transmis, preponderent regăsite în filtrul format de cavitatea bucală.

Pentru a converti spectrograma în coeficienți cepstrali, sunt necesari următorii pași:

- Transformăm spectrograma inițială într-o spectrogramă Mel.
- Aplicăm spectrogramei Mel algoritmul Discrete Cosine Transform (DCT).

Algoritmul DCT se folosește pentru a transforma benzile Mel în coeficienți cepstrali. Primii 12-13 coeficienți conțin cea mai relevantă informație despre semnalul audio. De

asemenea, algoritmul ajută la reducerea dimensiunii datelor și la decorelarea benzilor Mel. Corelația dintre benzile Mel apare datorită filtrelor triunghiulare suprapuse.

## 2.6 Metrici de evaluare

Pentru a măsura performanța rețelelor neurale dezvoltate în această lucrare, vom utiliza următoarele metrici: acuratețea, matricea de confuzie și consumul energetic.

### 2.6.1 Acuratețea

Acuratețea este una dintre principalele metrici utilizate pentru evaluarea performanței unui model de clasificare. Acuratețea este definită ca procentul de predicții corecte din totalul predicțiilor făcute de model. Formula pentru calcularea acurateței este:

$$\text{Acuratețe} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

### 2.6.2 Precizia și recall

Precizia indică procentul de predicții pozitive care sunt corecte, în timp ce recall-ul arată procentul de exemple pozitive care au fost prezise corect. Aceste măsuri pot fi calculate folosind formulele:

$$\text{Precizie} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

În această lucrare, vom aplica metoda Macro-Averaging, care calculează individual precizia și recall-ul pentru fiecare clasă, iar apoi realizează media acestora.

### 2.6.3 Matricea de confuzie

Matricea de confuzie este o metrică esențială pentru evaluarea performanței unui model de clasificare multi-clasă. Aceasta prezintă, într-un format tabular, numărul de valori True Positive (TP), True Negative (TN), False Positive (FP) și False Negative (FN), oferind astfel o vedere de ansamblu asupra predicțiilor modelului.

### 2.6.4 Consumul energetic

Pentru a măsura consumul de energie al fiecărei rețele, vom utiliza biblioteca snnTorch. Aceasta ne permite să estimăm consumul energetic al rețelelor neurale artificiale pe arhitectura de tip Von Neumann, precum și al rețelelor neurale de tip spike pe arhitectura neuromorfică. Costul energetic al unei inferențe prin rețea va fi măsurat în Jouli.

# Capitolul 3

# Fundamentele teoretice ale rețelelor neurale spike

## 3.1 Cele trei proprietăți ale codului neural

Există multiple teorii privind modul în care creierul codifică informația, fiecare se bazeaza pe trei proprietăți fundamentale: comunicarea prin spike-uri, sparitatea datelor și suprimarea statică. Aceste proprietăți sunt esențiale pentru înțelegerea principiilor de funcționare ale creierului și pentru aplicarea lor în îmbunătățirea eficienței rețelelor neurale artificiale.

### 3.1.1 Spike-uri

Neuronii biologici comunică prin intermediul impulsurilor electrice numite spike-uri, care au o amplitudine de aproximativ 100 mV. Absența unui spike poate fi reprezentată prin 0, iar prezența unui spike prin 1, acesta fiind unul dintre motivele pentru care rețelele neurale bazate pe spike-uri sunt mai eficiente. În Figura 3.1 putem vizualiza un neuron și modelarea output-ului acestuia sub forma unui tensor.

Într-o rețea neurală artificială, pentru a transmite activarea unui neuron către alt

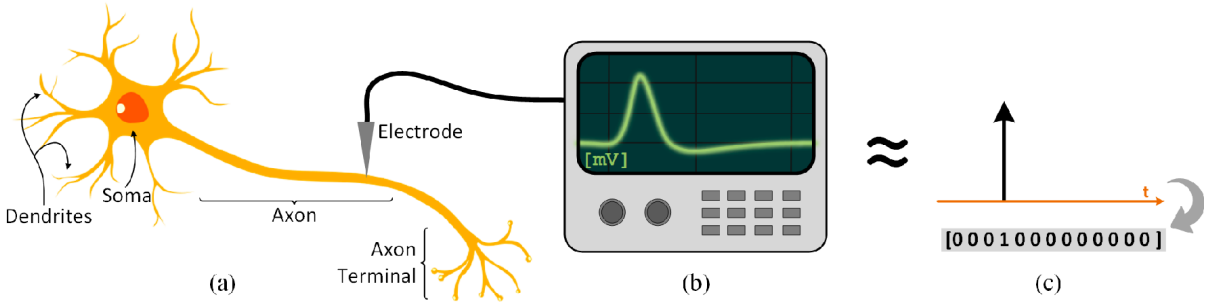


Figura 3.1: Reprezentarea output-ului unui neuron prin intermediul unui tensor - Sursa [7]

neuron, este necesară înmulțirea a două numere de înaltă precizie: activarea și greutatea. Această operație nu este eficientă și produce latență. Într-o rețea cu spike-uri, pentru a transmite activarea unui neuron către alt neuron, este necesară doar înmulțirea greutății cu un spike (valoarea 1). Astfel, înmulțirea numerelor de înaltă precizie este înlocuită cu citirea greutății din memorie, ceea ce reduce semnificativ complexitatea și latența operațiilor.

Deși activările din rețelele neurale spike sunt valori discrete de 0 și 1, există diferențe semnificative între acestea și rețelele neurale binarizate. În rețelele neurale spike, momentul apariției spike-urilor este esențial.

### 3.1.2 Sparsitatea

Sparsitatea se referă la proprietatea neuronilor de a rămâne în repaus majoritatea timpului, activările fiind de obicei 0. Datorită acestei proprietăți, putem salva eficient activările neuronilor. În loc să stocăm întregul vector de valori 0 și 1, putem salva doar pozițiile în care au apărut spike-urile (unde valoarea este 1). De exemplu, vectorul [0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1] poate fi înlocuit cu o structură de date care indică pozițiile în care apare valoarea 1: pozițiile 2, 7 și 16.

### 3.1.3 Suprimarea statică

Suprimarea statică reprezintă capacitatea senzorilor de a transmite informații doar atunci când apare ceva nou. În cazul unei camere video obișnuite, videoclipul este alcătuit dintr-o succesiune de imagini capturate la intervale scurte de timp. În schimb, un senzor bazat pe evenimente, cum ar fi senzorul de viziune dinamică (DVS) sau retina de siliciu, transmite informații doar atunci când există o schimbare. Fiecare pixel funcționează independent, capturând și comunicând doar noutățile. În Figura 3.2 se poate observa modul de funcționare al unui senzor bazat pe evenimente. Singurele informații transmise sunt zonele roșii și verzi, deoarece în aceste regiuni se detectează mișcare. Diferenta dintre o camera obisnuita si un senzor bazat pe evenimente este ilustrat in Figura 3.3.

Prin utilizarea senzorului bazat pe evenimente, consumul energetic este redus semnificativ datorită volumului scăzut de date transmise. Acest tip de senzor se concentrează exclusiv pe schimbările relevante din mediu, eliminând necesitatea de a procesa și transmite informații redundante sau statice.

## 3.2 Modele de neuroni

Rețelele neurale artificiale și cele bazate pe spike-uri pot modela aceleași arhitecturi de rețele. Diferența dintre ele constă în tipul de neuroni utilizați.

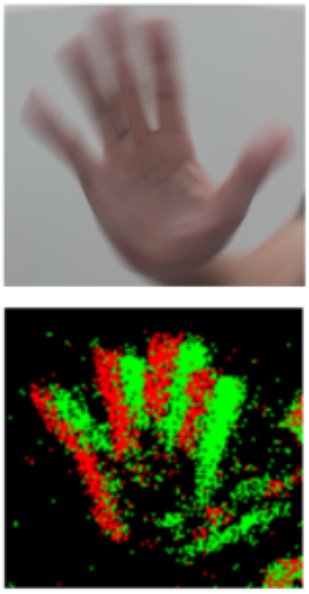

Figura 3.2: Senzor bazat pe evenimente (DVS) - Sursa [8]

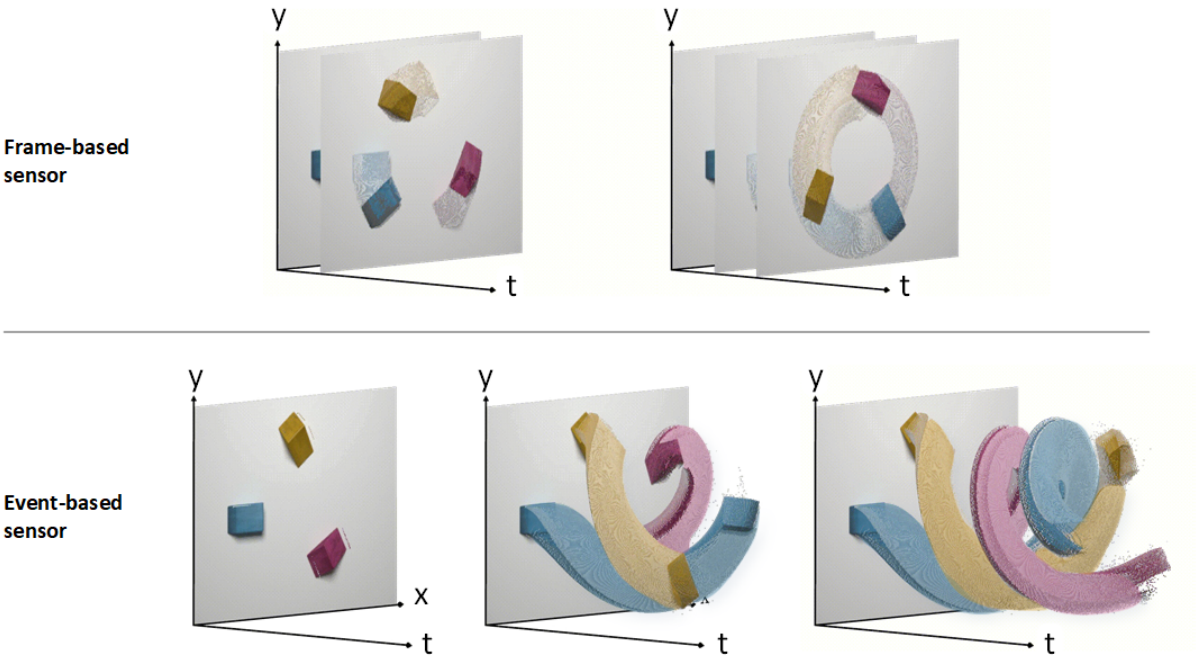


Figura 3.3: Comparație intre o camera convențională și un senzor bazat pe evenimente - Sursa [9]

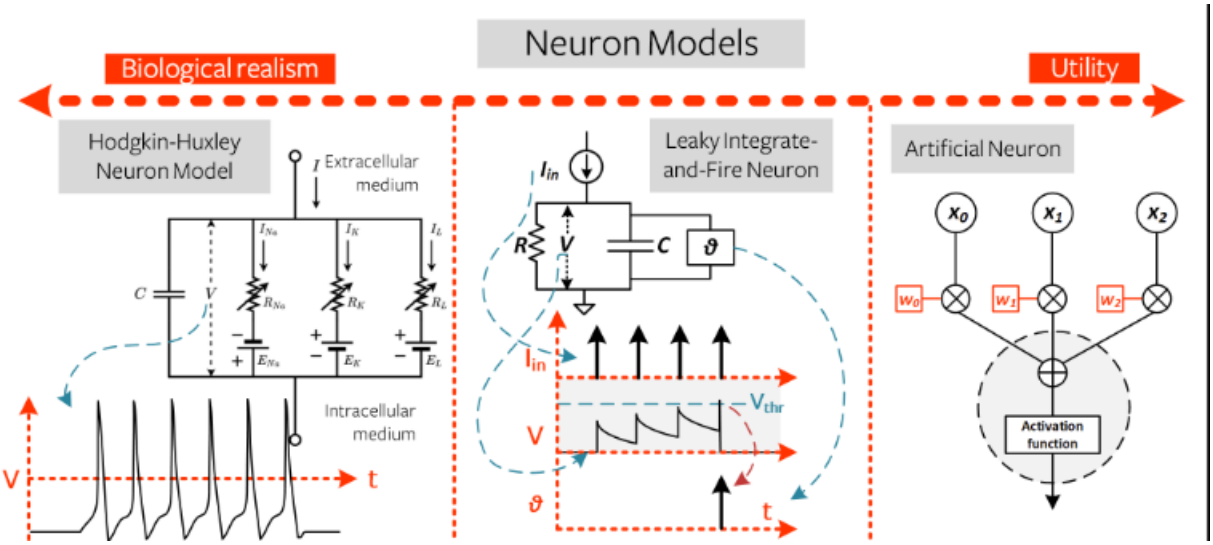


Figura 3.4: Modele de neuroni - Sursa [10]

Există numeroase tipuri de neuroni, care variază de la modele complexe, precum modelul Hodgkin-Huxley, până la perceptronul utilizat în rețelele neurale artificiale. Modelele complexe sunt mai plauzibile din punct de vedere biologic, dar au performanțe scăzute. În schimb, perceptronul este un model simplist care a obținut rezultate remarcabile.

Figura 3.4 prezintă spectrul modelelor existente de neuroni. În partea stângă se află modelul cel mai plauzibil din punct de vedere biologic, iar în partea dreaptă se află perceptronul utilizat în fiecare rețea neurală artificială. Modelul "Leaky Integrate-and-Fire" se situează la granița dintre cele două.

### 3.2.1 Perceptronul

Perceptronul reprezintă modelul de neuron utilizat în rețelele neurale artificiale. Inputurile perceptronului sunt înmulțite cu greutățile asociate, iar suma acestor înmulțiri este introdusă în funcția de activare precum sigmoid sau ReLU. Acest model a obținut rezultate impresionante în numeroase domenii ale învățării automate.

### 3.2.2 Neuronul biologic

Neuronul este celula responsabilă de recepționarea și transmiterea informației prin impulsuri electrice. În structura sa complexă, neuronul include corpulul celular, dendritele și axonul. Structura acestuia se poate observa in figura 3.5.

#### Dentritele

Dendritele au rolul de a capta semnalele primite de la alți neuroni și de a le transmite către corpul celular. Ele joacă un rol crucial în generarea potențialului de acțiune, care este esențial pentru transmiterea informației în sistemul nervos

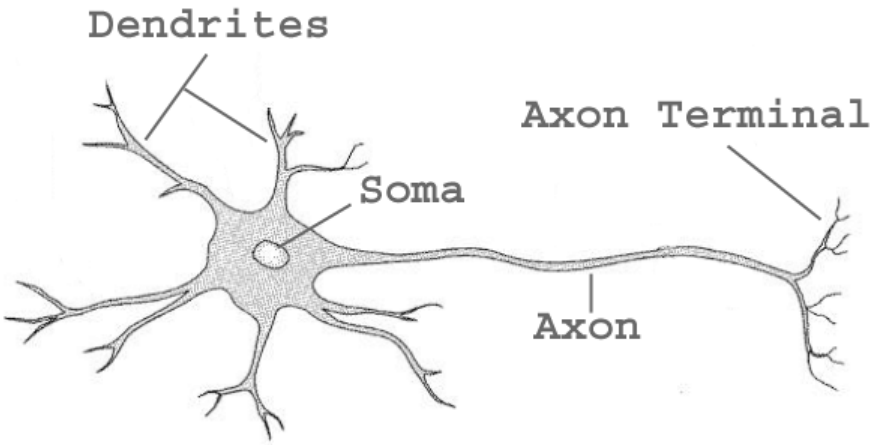


Figura 3.5: Structura unui neuron - Sursa: [11]

**Corpul celular**

Corpul celular, care conține nucleul celulei, împreună cu dendritele formează regiunea receptoare a neuronului. Corpul celular integrează toate informațiile primite de la dendrite și, dacă aceste semnale sunt suficient de puternice, generează un impuls electric numit potențial de acțiune, care este transmis prin axon.

**Axon**

Axonul face parte din regiunea conducătoare a neuronului și este responsabil de transmiterea potențialului de acțiune de la corpul celular către alți neuroni. La capătul axonului se află terminațiile axonale, care formează sinapse prin conectarea cu dendritele altor neuroni, facilitând astfel comunicarea neurală.

### 3.2.3 Neuronul Leaky Integrate-and-Fire

Modelul a fost dezvoltat de Louis Lapicque în anul 1907. Lapicque a stimulat nervul piciorului unei broaște cu impulsuri electrice pentru a observa cât stimul este necesar pentru a provoca mișcarea piciorului.

Neuronul Leaky Integrate-and-Fire este inspirat de neuronul biologic și funcționează trimițând un semnal mai departe doar dacă input-ul este suficient de puternic. Deoarece este puțin probabil ca neuronul să primească mai multe spike-uri simultan la un moment dat, este esențială o proprietate de persistență temporală a potențialului membranei pentru a genera spike-uri. Termenul ”integrate” se referă la capacitatea acestui model de a acumula și reține valoarea potențialului membranei în timp. ”Leaky” se referă la difuzia

ionilor prin membrană, ceea ce înseamnă că la fiecare pas de timp, potențialul membranei scade cu o rată $\beta$.

Neuronul Leaky Integrate-and-Fire se distinge prin potențialul membranei, rata de difuzie $\beta$ și un prag specific. Similar perceptronului, acest model insumează produsul dintre input și greutăți. Suma obținută se adaugă la potențialul membranei, iar dacă aceasta depășește pragul neuronului, se emite un spike. La fiecare pas de timp, potențialul membranei scade cu o rată $\beta$, reflectând difuzia ionilor prin membrană.

Pentru a înțelege mai bine modul de funcționare al acestui model, vom prezenta pașii executați într-o simulare de $N$ iterații:

1. **Inițializare**: Setăm rata de difuzie $\beta$, pragul $P$ și potențialul membranei la 0.
2. **Calculul input-ului**: Pentru un input $X$, calculăm produsul scalar dintre $X$ și greutățile asociate.
3. **Actualizarea potențialului**: Adăugăm rezultatul de la pasul 2 la potențialul membranei.
4. **Scăderea potențialului**: Potențialul membranei scade prin înmulțirea cu rata $\beta$.
5. **Emiterea spike-ului**: Dacă potențialul membranei depășește $P$, neuronul emite un spike, iar potențialul este resetat.
6. **Repetare**: Repetăm pașii 2, 3, 4 și 5 de $N$ ori.

Acești pași descriu modul în care neuronul Leaky Integrate-and-Fire procesează informația, acumulează potențialul și emite spike-uri ca răspuns la stimuli.

### Formula pentru potențialul membranei

$$U[t] = \beta U[t-1] + W \cdot X[t] - S_{\text{out}}[t-1]\theta \tag{3.1}$$

unde:

- $U[t]$ reprezintă potențialul membranei la momentul de timp $t$.
- $\beta$ este rata de difuzie a potențialului.
- $W \cdot X[t]$ este rezultatul produsului scalar dintre greutățile ($W$) și input-urile ($X[t]$) neuronului.
- $\theta$ este threshold-ul neuronului.
- $S_{\text{out}}[t] \in \{0, 1\}$ reprezintă output-ul neuronului la momentul $t$.
- $S_{\text{out}}[t-1]\theta$ reprezintă condiția în care neuronul a emis un spike și potențialul membranei trebuie resetat.

**Determinarea output-ului neuronului**

$$S_{\text{out}}[t] = \text{H}(U[t] - \theta) = \begin{cases} 1 & \text{dacă } U[t] \geq \theta \\ 0 & \text{altfel} \end{cases} \tag{3.2}$$

**Modalități de resetare a potențialului membranei**

Există două metode care se pot utiliza:

- **Soft reset**
  - **Formula:** $\beta S_{\text{out}}[t-1]\theta$.
  - Cand pragul este depășit, potențialul membranei nu este resetat la 0.
  - Oferă performanțe mai bune.
- **Hard reset sau resetarea la zero**
  - **Formula:** $S_{\text{out}}[t-1]\theta$.
  - Cand pragul este depășit, potențialul membranei este setat la 0.

## 3.3 Reprezentarea Datelor

Atât în creierul uman, cât și în rețelele neurale de tip spike, comunicarea se realizează prin intermediul impulsurilor electrice, denumite spike-uri. Informațiile din mediul înconjurător, cum ar fi temperatura sau presiunea aerului, sunt captate de senzori și reprezentate prin numere reale. Prin urmare, pentru a transmite și interpreta datele într-o rețea neurală spike, sunt necesare diverse metode de conversie.

### 3.3.1 Encodarea input-ului

Neuronul Leaky Integrate-and-Fire poate accepta valori continue, permițând astfel rețelelor neuronale spike să primească atât spike-uri, cât și numere reale. Există trei metode principale de a converti datele continue în spike-uri: rate coding, latency coding și delta modulation.

**Rate coding**

Această metodă de codare asociază valorilor mari un număr ridicat de spike-uri, iar valorilor mici un număr redus de spike-uri. De exemplu, într-o imagine, un pixel luminos va genera mai multe spike-uri decât un pixel întunecat într-un interval de timp. Atunci când nu se produc spike-uri, rețeaua nu învață, situație cunoscută sub numele de problema

neuronului mort. Rate coding elimină această problemă deoarece promovează un număr mare de spike-uri. Relația dintre valoarea input-ului și numărul de spike-uri conferă acestei metode o toleranță ridicată la eroare. Cu toate acestea, rate coding are un consum energetic ridicat din cauza numărului mare de spike-uri generate.

#### Latency coding

În această metodă, nu numărul de spike-uri produse este esențial, ci momentul în care apar aceste spike-uri. Un pixel luminos dintr-o imagine va produce primul spike, în timp ce un pixel întunecat îl va produce ultimul sau deloc. Spre deosebire de rate coding, latency coding atribuie o importanță mai mare fiecărui spike, ceea ce face metoda mai susceptibilă la posibile erori. Cu toate acestea, avantajul major al acestei metode constă în consumul redus de energie. De asemenea, un număr mai mic de spike-uri implică mai puține accesări ale memorie, ceea ce contribuie la eficiența generală.

#### Delta modulation

Metoda delta modulation este ideală pentru date cu dimensiune temporală, cum ar fi un videoclip sau o înregistrare audio. În cazul unui videoclip, neuronul asociat unui pixel va emite un spike doar atunci când diferența dintre două frame-uri succesive depășește un anumit prag. În figura 3.3, putem observa cum senzorul bazat pe eveniment transmite doar schimbările din cadru, ignorând fundalul.

### 3.3.2 Decodearea output-ului

Reprezinta modul in care interpretam output-ul retelei neurale spike. Exita 3 metode principale: rate coding, latency coding si population coding.

#### Rate coding

Considerăm cazul clasificării în N clase. Într-o rețea neurală artificială, vom avea N neuroni în stratul de ieșire, iar clasa prezisă este asociată neuronului cu activarea cea mai mare. În cazul unei rețele neuronale spike care utilizează rate coding pentru decodarea output-ului, clasa prezisă este determinată de neuronul care produce cele mai multe spike-uri într-un anumit interval de timp.

#### Latency coding

Clasa prezisă este asociată cu primul neuron care produce un impuls electric. Această metodă este inspirată de modul în care creierul nostru răspunde la stimuli din mediul înconjurător.

#### Population coding

Aceasta metodă funcționează similar cu rate coding, dar implică un număr mai mare de neuroni asociați fiecărei clase.

## 3.4 Antrenarea rețelelor neurale spike

Antrenarea rețelelor neuronale spike poate fi realizată prin trei metode principale: shadow training, reguli de învățare locală și backpropagation bazat pe spike-uri.

**Shadow Training** implică antrenarea unei rețele neurale artificiale convenționale, care este ulterior convertită într-o rețea neurală spike.

**Local learning rules** se bazează pe aplicarea algoritmului de backpropagation în zone locale din rețea, permițând ajustarea localizată a conexiunilor neurale.

**Algoritmul de backpropagation bazat pe spike-uri** permite antrenarea rețelelor neurale de tip spike de la zero. În lucrare, vom aplica această metodă pentru antrenarea rețelelor neurale.

### 3.4.1 Backpropagation

Algoritmul backpropagation facilitează antrenarea rețelelor neurale prin minimizarea unei funcții obiectiv. Acesta ne permite să analizăm contribuția fiecărui parametru la minimizarea funcției folosind regula lanțului. Odată ce gradientul este calculat, acesta ne ajută să actualizăm parametrii în mod optim pentru a minimiza funcția obiectiv.

Actualizarea greutăților $W_{\text{in}}$ se realizează în modul următor:

$$W_{\text{in}} \leftarrow W_{\text{in}} - \eta \frac{\partial L}{\partial W_{\text{in}}} \tag{3.3}$$

Pentru a actualiza $W_{\text{in}}$ este necesară derivata funcției obiectiv în raport cu greutățile neuronului. Această derivată se calculează utilizând regula lanțului astfel:

$$\frac{\partial L}{\partial W_{\text{in}}} = \frac{\partial L}{\partial S_{\text{out}}} \frac{\partial S_{\text{out}}}{\partial U} \frac{\partial U}{\partial W_{\text{in}}} \tag{3.4}$$

Componenta din mijloc $\frac{\partial S_{\text{out}}}{\partial U}$ reprezintă derivata funcției Heaviside (3.2). Această funcție este utilizată deoarece modelează bine modul în care este calculat output-ul neuronului. Cu toate acestea, derivata ei introduce câteva probleme.

#### Funcția treaptă (Heaviside) nediferențiabilă

Derivata funcției treaptă reprezintă funcția Dirac-Delta:

$$\frac{\partial S}{\partial U} = \delta(U[t] - \theta) \in \{0, \infty\} \tag{3.5}$$

Problema neuronilor inactivi (dead neuron problem) apare atunci când derivata funcției este preponderent zero, ceea ce împiedică actualizarea greutăților neuronilor. În plus, funcția Dirac nu este derivabilă în punctul 0, unde valoarea funcției tinde spre infinit.

### Surrogate gradients

Pentru a rezolva această problemă, se pot utiliza "surrogate gradients" (gradienti de substituție). Această metodă constă în utilizarea funcției Heaviside în pasul de forward, dar înlocuirea acesteia cu o funcție continuă similară în pasul de backward. În librăria snnTorch, funcția Heaviside este înlocuită cu funcția arctan în timpul pasului de backward. Această abordare permite actualizarea eficientă a greutăților și evitarea problemelor asociate cu derivata nulă a funcției Heaviside.

# Capitolul 4

# Setul de date

## 4.1 Descriere

Pentru antrenarea rețelelor neuronale, am utilizat baza de date AccentDB [12]. Aceasta conține aproximativ 20 de ore de date audio, înregistrate la o frecvență de 22050 Hz. Înregistrările audio sunt etichetate cu 9 accente distincte:

- patru accente non-native: Bangla, Malayalam, Odiya și Telugu.
- un accent indian metropolitan.
- patru accente native: American, Australian, British și Welsh.

Cele patru accente non-native au fost înregistrate de persoane cu accent pronunțat, în timp ce restul accentelor au fost generate de sistemul text-to-speech Amazon Polly.

Distribuția datelor în funcție de accent poate fi vizualizată în Figura 4.2, iar detalii suplimentare despre dimensiunea setului se regăsesc în Figura 4.1.

## 4.2 Preprocesarea datelor

În urma analizei datelor audio, am observat că multe înregistrări conțineau pauze la început și la final. Deoarece aceste pauze nu oferă informații relevante pentru clasificarea accentelor, am creat și utilizat scriptul Python `remove_silence.py`, care elimină segmentele de semnal audio în care amplitudinea nu atinge un anumit prag. Diferențele dintre distribuția lungimii înregistrărilor audio cu și fără pauze pot fi observate în Figurile 4.3 și 4.4.

Al doilea pas în preprocesarea datelor a constat în uniformizarea duratei înregistrărilor audio. În cazul înregistrărilor mai scurte, acestea au fost repetate pentru a atinge dimensiunea dorită, iar în cazul celor mai lungi, au fost tăiate. Dimensiunea aleasă trebuie să fie o balanța între a nu pierde informații prin tăiere și a nu introduce redundanță

| | Accent | Number of Samples | Duration | Number of Speakers |
|---|---|---|---|---|
| **AccentDB** | Bangla | 1528 | 2 h 13 min | 2 |
| | Malayalam | 2393 | 3 h 32 min | 3 |
| | Odiya | 748 | 1 h 11 min | 1 |
| | Telugu | 1515 | 2 h 10 min | 2 |
| | **Total** | **6184** | **9 h 6 min** | **8** |
| **Amazon Polly** | American | 5760 | 5 h 44 min | 8 |
| | Australian | 1440 | 1 h 21 min | 2 |
| | British | 1440 | 1 h 26 min | 2 |
| | Indian | 1440 | 1 h 29 min | 2 |
| | Welsh | 720 | 0 h 43 min | 1 |
| | **Total** | **10 800** | **10 h 43 min** | **15** |
| **Total** | | **16 984** | **19 h 49 min** | **23** |

Figura 4.1: Datele din baza de date accentDB - Sursa [12]

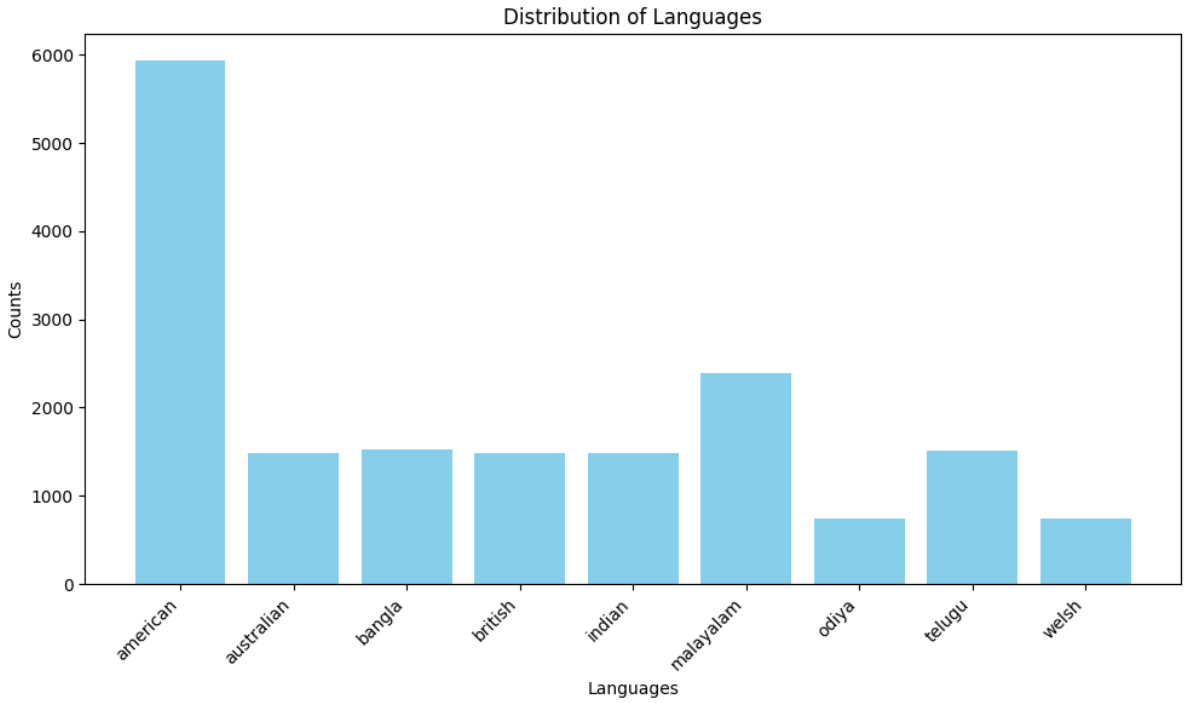


Figura 4.2: Distribuția accentelor

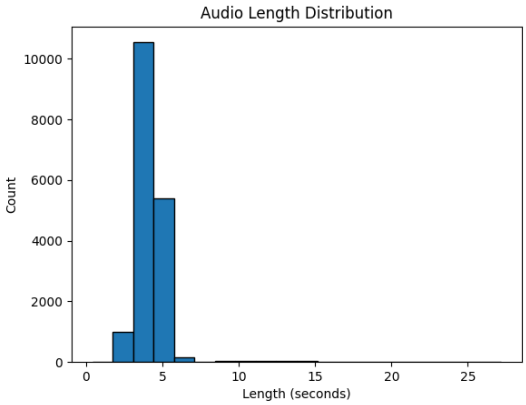


Figura 4.3: Distribuția lungimilor datelor audio cu pauze

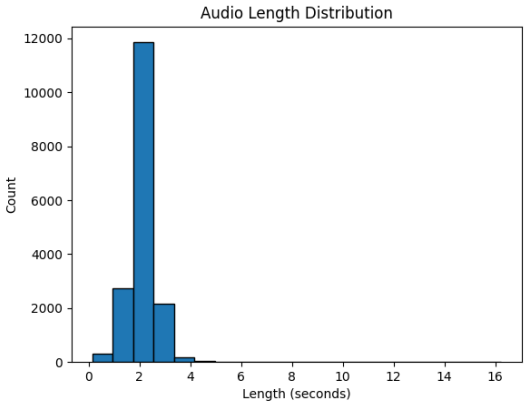


Figura 4.4: Distribuția lungimilor datelor audio fără pauze

prin repetare. Ținând cont de distribuția lungimilor, dimensiunea aleasă a fost de patru secunde. Deoarece datele audio au fost înregistrate la 22050 Hz, fiecare fișier audio a avut după preprocesare 88200 de elemente.

Pentru a observa diferențele dintre accente, am redus dimensiunea datelor utilizând analiza componentelor principale (PCA). Am reprezentat grafic cele trei componente principale rezultate din fiecare înregistrare, facilitând vizualizarea separării dintre accente, așa cum se poate observa în Figura 4.5.

## 4.3 Extragerea trăsăturilor

Pentru extragerea trăsăturilor din datele audio, am utilizat biblioteca Librosa din Python. Cele trei trăsături extrase sunt: amplitudinea, spectrograma Mel și coeficienții cepstrali.

Scriptul folosit pentru generarea acestor trăsături este `convert_to_features.py`. Acesta acceptă un parametru care specifică tipul trăsăturilor dorite: amplitude, mel sau mfcc.

### Amplitudinea

Una dintre problemele asociate utilizării amplitudinii ca trăsătură este numărul mare de dimensiuni în comparație cu numărul de înregistrări, un fenomen cunoscut sub numele de blestemul dimensionalității. Pentru a evita această problemă, am efectuat o reducere a dimensiunii prin subeșantionarea înregistrărilor audio. Am redus rata de eșantionare de la 22050 Hz la 16000 Hz, reducând astfel dimensiunea input-ului de la 88200 elemente la 64000 de elemente. Această reducere nu doar că ajută la diminuarea complexității datelor, dar și îmbunătățește eficiența procesării și antrenării modelelor.

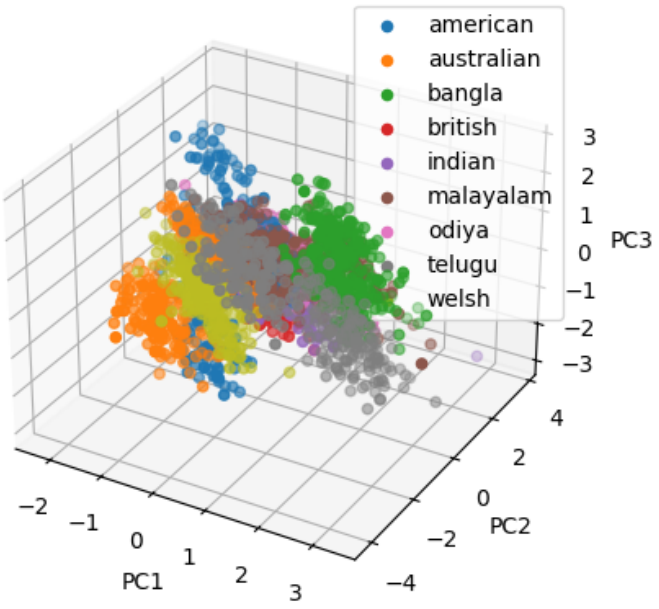


Figura 4.5: Reprezentarea PCA a datelor pentru diferite accente

### Spectrograma Mel

Dimensiunea ferestrei și dimensiunea pasului pentru algoritmul STFT au fost setate la 512 și, respectiv, 256, iar numărul benzilor Mel a fost ales 20. Numărul ferestrelor este dat de dimensiunea semnalului împărțită la dimensiunea pasului, adică $88200/256 \approx 345$. Fiecare fereastră are 20 de benzi Mel, rezultând o matrice de dimensiune (345, 20).

### Coeficienții cepstrali

Am extras 13 coeficienți cepstrali utilizând aceiași parametri menționați anterior pentru spectrograma Mel. În urma transformării, fiecare înregistrare a generat o matrice de dimensiune (345, 13).

## 4.4 Transformarea trăsăturilor în spike-uri

Prin utilizarea programului `convert_to_spikes.py`, trăsăturile extrase sunt convertite în date spike. Programul acceptă doi parametrii: tipul trăsăturilor (amplitudine, spectrogramă Mel sau coeficienți cepstrali) și tipul de codare utilizat (rate coding, latency coding sau delta modulation).

## 4.5 Pregătirea datelor pentru antrenarea rețelelor

Pentru a reprezenta datele împreună cu etichetele asociate, am creat o clasă denumită `AudioDataset`. Aceasta facilitează stocarea și manipularea datelor într-un mod organizat.

#### Setul de antrenare, validare si testare

Am împărțit setul de date în trei subseturi distincte:

- **Setul de antrenare (70%)**: Utilizat pentru antrenarea modelelor.
- **Setul de validare (15%)**: Folosit pentru evaluarea performanței modelelor în timpul antrenării și pentru ajustarea hiperparametrilor.
- **Setul de testare (15%)**: Utilizat pentru evaluarea performanței finale a modelelor după finalizarea antrenării.

#### Normalizarea datelor

Normalizarea datelor a fost realizată prin două metode:

- **Min-max**: Datele au fost scalate în intervalul [0, 1].
- **Standard score**: Datele au fost transformate astfel încât media să fie 0 și deviația standard să fie 1.

Normalizarea a fost efectuată separat pentru fiecare subset pentru a evita introducerea informațiilor din seturile de test și validare în setul de antrenare. Această practică este esențială pentru a asigura că modelele nu beneficiază de cunoașterea prealabilă a datelor de testare sau validare, ceea ce ar duce la rezultate nerealiste. Deși aceste metode de normalizare nu au oferit rezultate superioare semnificative în toate cazurile, ele au contribuit la stabilizarea procesului de antrenare.

#### Crearea batch-urilor

În modelele implementate, am utilizat batch-uri cu dimensiunea de 64 de exemple. Aceste batch-uri au fost create folosind clasa `DataLoader` din PyTorch. Utilizarea batch-urilor este esențială pentru a gestiona eficient memoria și pentru a optimiza procesul de antrenare.

# Capitolul 5

# Modele dezvoltate

## 5.1 Rețea neurală convoluțională antrenată cu amplitudinile semnalelor

### 5.1.1 Introducere

Semnalul audio conține variații ale amplitudinii în timp, iar procesul de convoluție este esențial pentru captarea schimbărilor locale din sunet. Aceasta abilitate de a detecta și analiza caracteristicile locale ale semnalului audio ajută semnificativ în clasificarea accentelor.

### 5.1.2 Arhitectura rețelei

Arhitectura rețelei neurale convoluționale utilizate pentru antrenarea cu amplitudinile semnalelor constă din două straturi convoluționale, două straturi de max-pooling și două straturi complet conectate.

Primul strat convoluțional utilizează 32 de filtre cu o dimensiune de 100 × 100. Al doilea strat convoluțional folosește 16 filtre cu o dimensiune de 100 x 100.

După fiecare strat convoluțional, este aplicat un strat de max-pooling cu o fereastră de 2 × 2 și un stride de 2. Acest lucru reduce dimensiunea spațială a caracteristicilor extrase, păstrând în același timp informațiile esențiale.

În continuare, rețeaua include două straturi complet conectate. Primul strat complet conectat are 512 de neuroni și utilizează funcția de activare ReLU. Al doilea strat complet conectat conține 9 neuroni și servește drept strat de ieșire. Pentru a reduce overfitting-ul, tehnica dropout este aplicată după fiecare strat complet conectat, dezactivând aleatoriu 50% dintre neuroni în timpul antrenării.

Funcția de cost utilizată pentru clasificare este cross-entropy loss. Optimizatorul ales pentru antrenare este Adam, cu o rată de învățare de 0.001.

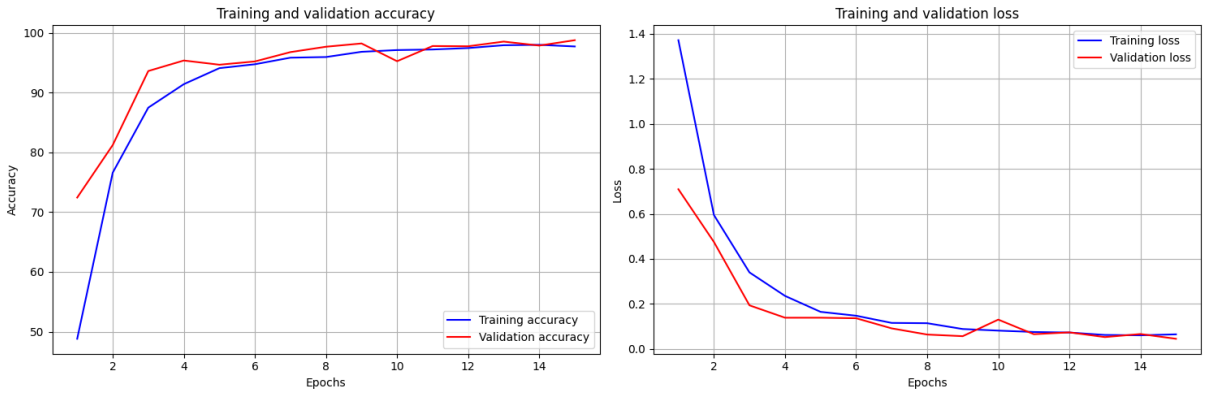


Figura 5.1: Acuratețea si loss-ul in timpul antrenării rețelei neurale convoluționale cu amplitudinile semnalului audio

### 5.1.3 Evaluarea rețelei

#### Performanță

După 15 epoci de antrenare, rețeaua neurală a atins o acuratețe de 98.4% pe setul de test. Figura 5.1 prezintă evoluția acurateței și a loss-ului în timpul antrenării.

Pentru o înțelegere mai profundă a performanței modelului, este util să examinăm precizia și recall-ul acestuia. Tabelul 5.1 oferă rezultatele pentru aceste metrici.

| Metrica | Valoare |
|---|---|
| Macro Precision | 0.9918 |
| Macro Recall | 0.9858 |

Tabela 5.1: Precizia si recall-ul rețelei neurale convoluționale antrenată cu amplitudinile semnalului audio

În plus, matricea de confuzie, prezentată în Figura 5.2, furnizează o imagine detaliată a modului în care modelul clasifică fiecare clasă în parte.

#### Consum energetic

Programul `estimate_energy.py CNN_amplitude` oferă datele obținute despre model, prezentate în Tabelul 5.6.

Numărul total de date de antrenare este 12121. Fiecare batch conține 64 de exemple, astfel că într-o epocă se vor procesa aproximativ $\frac{12121}{64} \approx 189$ de batch-uri. Modelul consumă 25 Jouli per inferență pentru un batch, deci consumul de energie al rețelei într-o singură epocă este de $189 \times 25\,\text{J} = 4725\,\text{J}$. Modelul este antrenat pe parcursul a 15 epoci, așadar consumul total de energie estimat pentru antrenarea modelului este de $15 \times 4725\,\text{J} = 70875\,\text{Jouli}$.

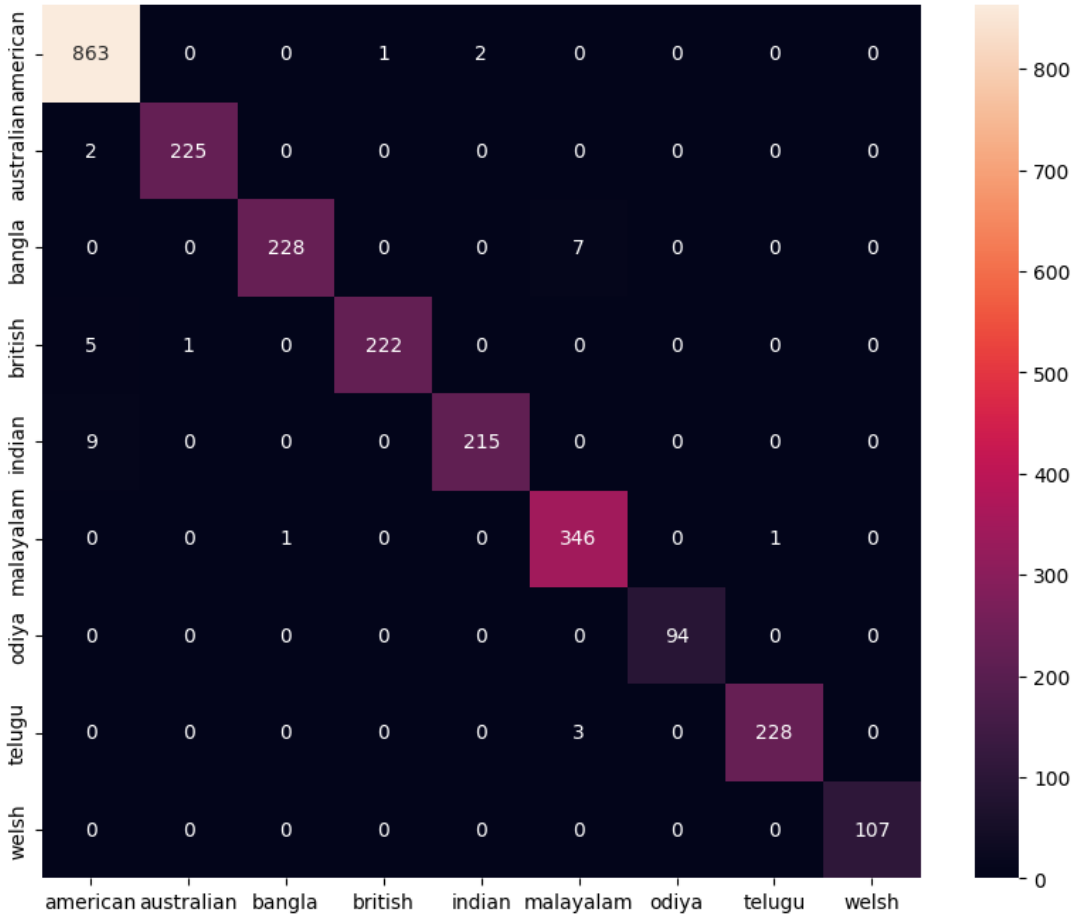


Figura 5.2: Matricea de confuzie a rețelei neurale convoluționale antrenată cu amplitudinile semnalului audio

| Strat | # Sinapse | # Neuroni | Rata spike-uri [Hz] | # Evenimente | # Spike-uri | Energie consumată [J] |
|---|---|---|---|---|---|---|
| **Conv1d** | | | | | | |
| Strat 1 | 3,232 | 0 | 0.0075 | 1,727,717,376 | 12,969,969 | 14.9 |
| Strat 2 | 1,638,912 | 0 | 0.0176 | 1,178,337,280 | 20,694,094 | 10.1 |
| **Linear** | | | | | | |
| Strat 1 | 3,150,274,560 | 0 | 0.0925 | 1,269,768 | 117,405 | 0.0109 |
| Strat 2 | 2,363,904 | 0 | 0.7407 | 4,608 | 3,406 | 3.96e-05 |
| **Total** | | | | | | |
| CNN amplitudine | 3,154,280,608 | 0 | 0.0116 | 2,907,329,032 | 33,784,874 | 25 |

Tabela 5.2: Consumul energetic al rețelei neurale convoluționale antrenată cu amplitudinile semnalului audio

## 5.2 Rețea neurală convoluțională antrenată cu coeficienți cepstrali

### 5.2.1 Introducere

Rețeaua neurală convoluțională utilizează ca input o matrice de dimensiune (13, 345), care reprezintă evoluția celor 13 coeficienți cepstrali pe parcursul a 345 de pași temporali. Convoluțiile aplicate asupra acestei matrice captează caracteristicile locale ale sunetelor, facilitând astfel recunoașterea accentelor.

### 5.2.2 Arhitectura rețelei

Rețeaua neurală convoluțională antrenată cu coeficienți cepstrali este compusă din două straturi convoluționale, două straturi de max-pooling și două straturi complet conectate. Funcția de activare utilizată în toate straturile este ReLU.

Primul strat convoluțional conține 32 de filtre de dimensiune 3 x 3, iar al doilea strat convoluțional conține 64 de filtre de dimensiune 3 x 3. Fiecare strat convoluțional este urmat de un strat de max-pooling cu o dimensiune de 2 x 2 și un stride de 2.

În plus, rețeaua include două straturi complet conectate: primul strat conține 512 neuroni, iar al doilea strat conține 9 neuroni, corespunzând claselor de ieșire.

Pentru a preveni supraînvățarea, am aplicat tehnica de regularizare dropout, eliminând aleatoriu 50% din neuroni în timpul antrenării.

Funcția de cost utilizată este cross-entropy loss. Optimizatorul ales pentru antrenare este Adam, cu o rată de învățare de 0.0005.

### 5.2.3 Evaluarea rețelei

#### Performanță

Rețeaua neurală a atins o acuratețe de 98.3% pe setul de test după 15 epoci de antrenare. Evoluția acurateței și a loss-ului în timpul antrenării este prezentată în figura 5.3.

În tabelul 5.3 sunt prezentate cele patru metrici specifice evaluării clasificatorilor multiclasă: Macro Precision și Macro Recall, iar matricea de confuzie este prezentată în figura 5.4.

#### Consum energetic

Programul `estimate_energy.py CNN_mfcc` oferă datele obținute despre model. Acestea se regasesc in tabelul 5.4.

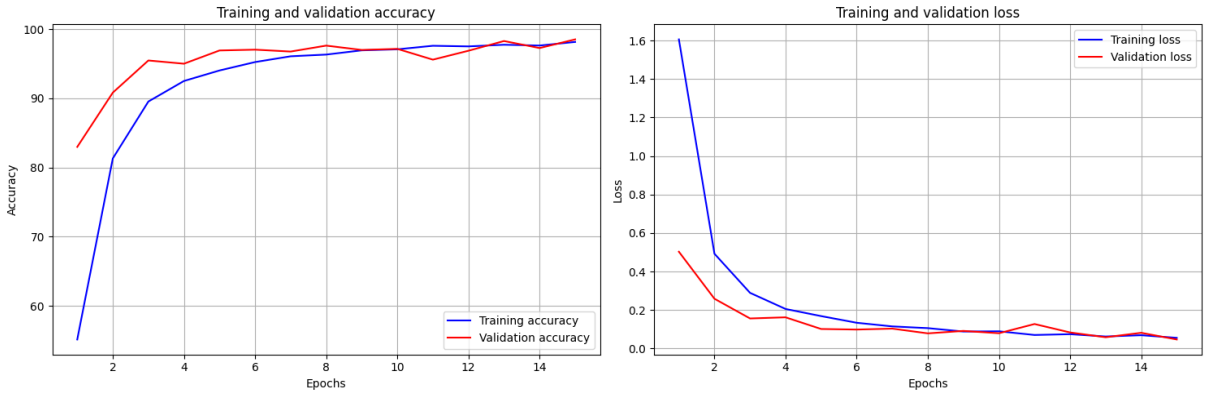


Figura 5.3: Acuratețea si loss-ul in timpul antrenării rețelei neurale convoluționale cu coeficienți cepstrali

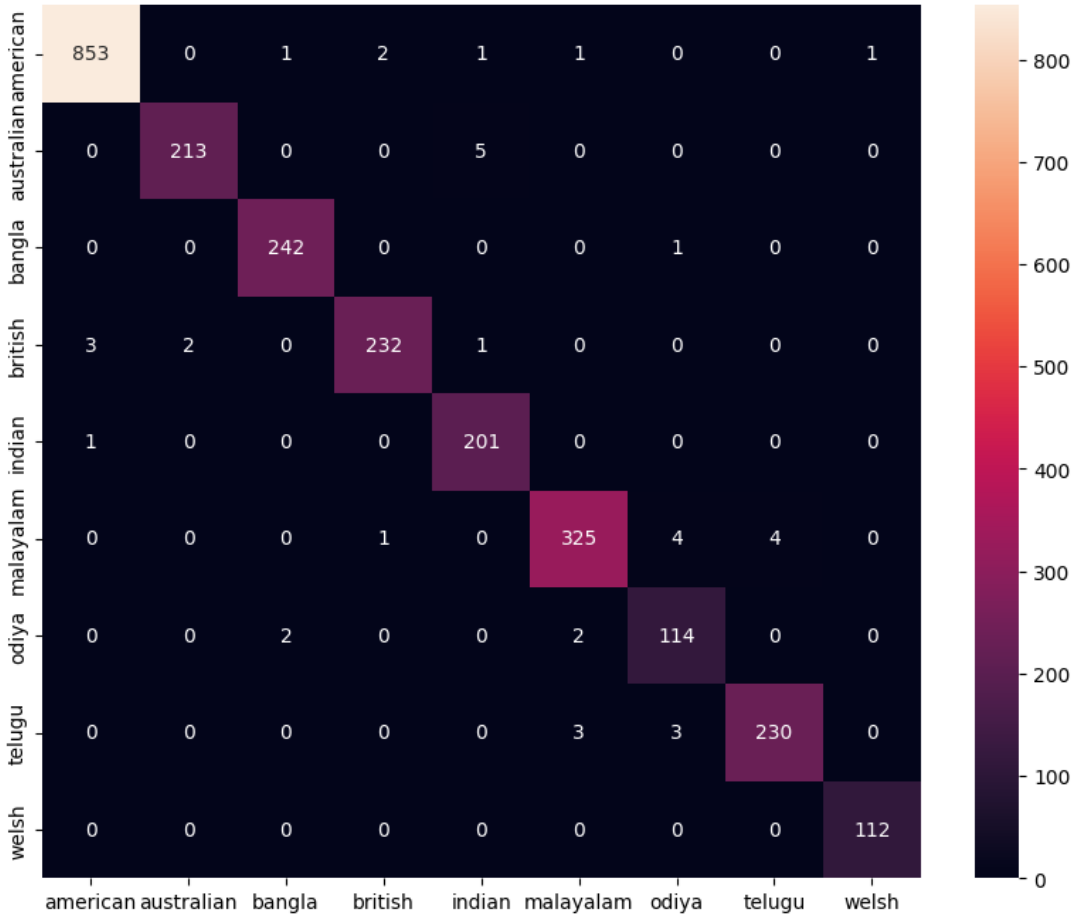


Figura 5.4: Matricea de confuzie a rețelei neurale convoluționale antrenată cu coeficienți cepstrali

| Metrica | Valoare |
|---|---|
| Macro Precision | 0.9797 |
| Macro Recall | 0.9832 |

Tabela 5.3: Precizia si recall-ul rețelei neurale convoluționale antrenată cu coeficienți cepstrali

| Strat | # Sinapse | # Neuroni | Rata spike-uri [Hz] | # Evenimente | # Spike-uri | Energie consumată [J] |
|---|---|---|---|---|---|---|
| **Conv2d** | | | | | | |
| Strat 1 | 320 | 0 | 0.1383 | 316,811,264 | 43,830,484 | 0.377 |
| Strat 2 | 591,872 | 0 | 0.759 | 4,851,105,792 | 3,681,927,168 | 31.7 |
| **Linear** | | | | | | |
| Strat 1 | 14,800,257,024 | 0 | 0.4047 | 2,752,520 | 1,113,937 | 0.00958 |
| Strat 2 | 2,363,904 | 0 | 1.3023 | 4,608 | 6,001 | 5.16e-05 |
| **Total** | | | | | | |
| CNN amplitudine | 14,803,213,120 | 0 | 0.7208 | 5,170,674,184 | 3,726,877,590 | 32.1 |

Tabela 5.4: Consumul energetic al rețelei neurale convoluționale antrenată cu coeficienți cepstrali

Numărul total de date de antrenare este 12121. Fiecare batch conține 64 de exemple, astfel că într-o epocă se vor procesa aproximativ $\frac{12121}{64} \approx 189$ de batch-uri. Modelul consumă 32.1 Jouli per inferență pentru un batch, deci consumul de energie al rețelei într-o singură epocă este de $189 \times 32.1\,\text{J} = 6066.9\,\text{J}$. Modelul este antrenat pe parcursul a 15 epoci, așadar consumul total de energie estimat pentru antrenarea modelului este de $15 \times 6066.9\,\text{J} = 91003.5$ Jouli.

## 5.3 Rețea neurală spike antrenată cu coeficienți cepstrali

### 5.3.1 Introducere

Dimensiunea temporală a datelor audio face rețelele neurale spike deosebit de potrivite pentru clasificarea accentelor. În acest context, rețeaua primește ca input o matrice de dimensiune (13, 345), unde cei 13 coeficienți cepstrali sunt procesați în 345 de pași temporali.

### 5.3.2 Arhitectura rețelei

Rețeaua neurală spike este formată din trei straturi distincte. Primul strat, stratul de input, cuprinde 13 neuroni corespunzători celor 13 coeficienți cepstrali. Al doilea strat este un strat complet conectat, constând din 128 de neuroni de tip Leaky Integrate-And-Fire (LIF). Al treilea strat, de asemenea complet conectat, conține 9 neuroni de tip LIF și reprezintă stratul de ieșire. Acești 9 neuroni din stratul de ieșire sunt responsabili pentru

clasificarea finală a accentelor.

Funcția de activare a perceptronului este înlocuită de funcționalitatea neuronului Leaky Integrate-And-Fire. La începutul antrenării, fiecare neuron LIF pornește cu o rată de decadere și un prag stabilite aleatoriu. Pe parcursul antrenării, aceste valori sunt optimizate individual pentru fiecare neuron. Astfel, la finalul antrenării, fiecare neuron va avea o rată de decadere și un prag specific.

Pentru a clasifica o înregistrare audio, rețeaua primește ca input secvențe de 13 coeficienți cepstrali de 345 de ori. Output-ul rețelei este o matrice de dimensiune (9, 345), unde fiecare element indică dacă un neuron din stratul de ieșire a emis sau nu un spike la fiecare dintre cei 345 de pași temporali.

Decodarea output-ului se realizează prin metoda rate coding. Clasa prezisă este determinată de neuronul care a produs cele mai multe spike-uri în cei 345 de pași. Funcția de cost utilizată pentru antrenare este cross-entropy loss, iar optimizatorul folosit este Adam, cu o rată de învățare setată la 0.001.

### 5.3.3 Evaluarea rețelei

#### Performanță

După 15 epoci de antrenare, rețeaua neurală spike a atins o acuratețe de 96.5% pe setul de test. În Figura 5.5 este prezentată evoluția acurateței și a loss-ului pe parcursul antrenării.

Metricile Macro Precision, Macro Recall, Micro Precision și Micro Recall sunt detaliate în Tabelul 5.5, iar matricea de confuzie este ilustrată în Figura 5.6.

| **Metrica** | **Valoare** |
|---|---|
| Macro Precision | 0.9478 |
| Macro Recall | 0.9628 |

Tabela 5.5: Precizia si recall-ul rețelei spike antrenată cu coeficienți cepstrali

#### Consum energetic

Programul `estimate_energy.py SNN_mfcc` oferă datele obținute despre model, prezentate în Tabelul 5.6.

Numărul total de date de antrenare este 12121. Fiecare batch conține 64 de exemple, astfel că într-o epocă se vor procesa aproximativ $\frac{12121}{64} \approx 189$ de batch-uri. Modelul consumă 0.0158 Jouli per inferență pentru un batch, deci consumul de energie al rețelei într-o singură epocă este de $189 \times 0.0158\,\text{J} = 2.9862\,\text{J}$. Modelul este antrenat pe parcursul a 15 epoci, așadar consumul total de energie estimat pentru antrenarea modelului este de $15 \times 2.9862\,\text{J} = 44.8\,\text{Jouli}$.

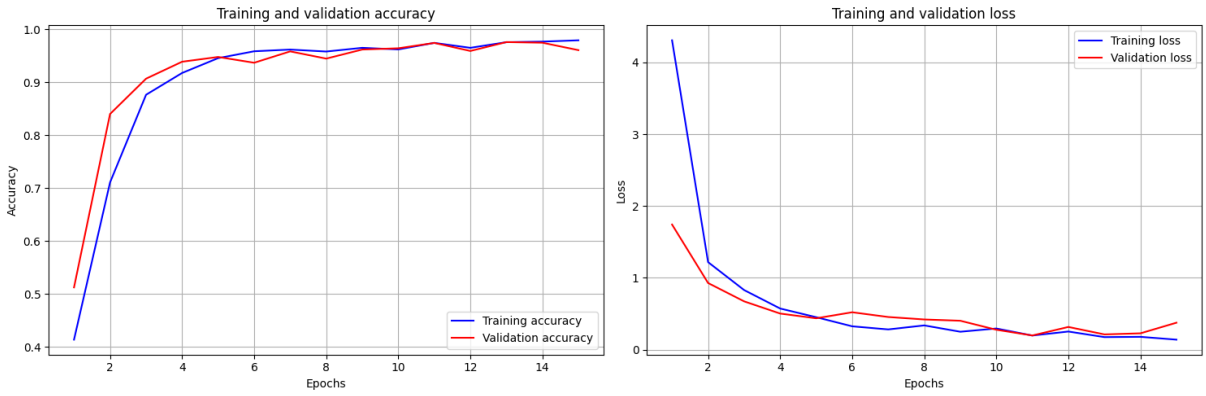


Figura 5.5: Acuratețea si loss-ul in timpul antrenării rețelei spike

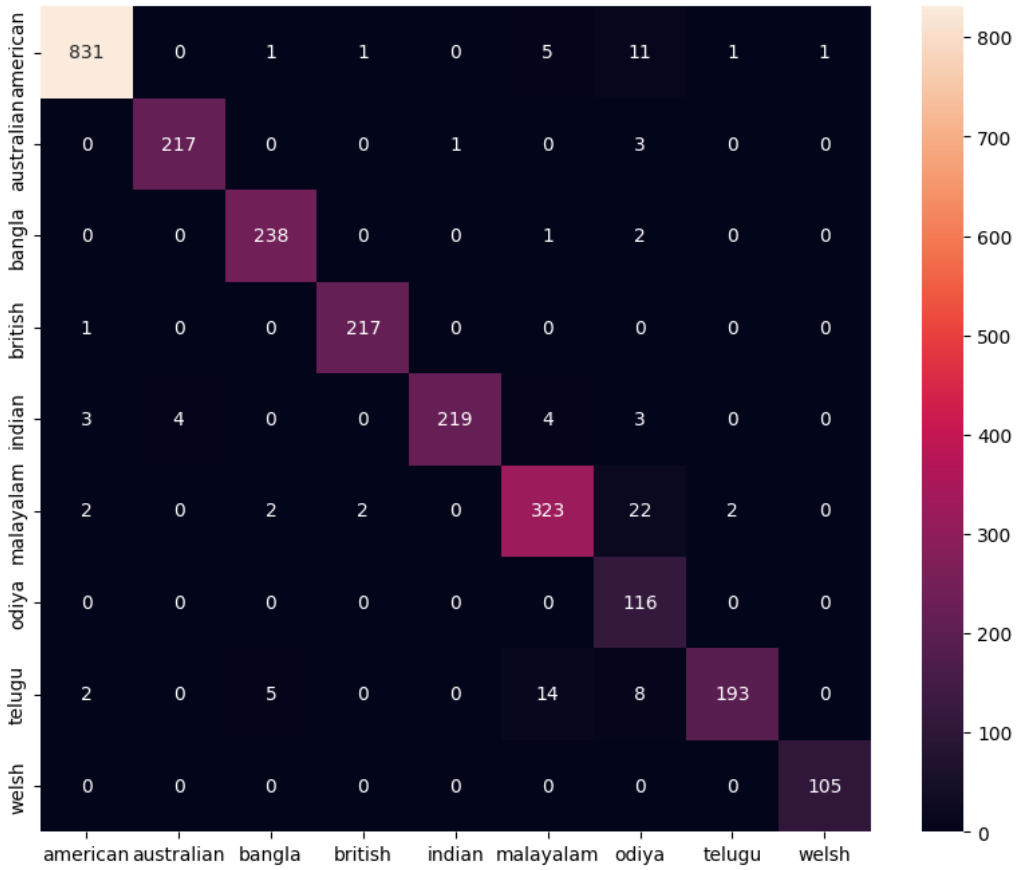


Figura 5.6: Matricea de confuzie a rețelei spike antrenată cu coeficienți cepstrali

| Strat | # Sinapse | # Neuroni | Rata spike-uri [Hz] | # Evenimente | # Spike-uri | Energie consumată [J] |
|---|---|---|---|---|---|---|
| **Linear** | | | | | | |
| Strat 1 | 114,688 | 0 | 0.9995 | 2,829,345 | 2,827,813 | 0.00283 |
| Strat 2 | 7,4304 | 0 | 0.4605 | 25,439,265 | 11,715,399 | 0.0117 |
| **Leaky** | | | | | | |
| Strat 1 | 0 | 8,192 | 0.4605 | 2,826,240 | 1,301,366 | 0.0013 |
| Strat 2 | 0 | 576 | 0.128 | 198,720 | 25,430 | 2.54e-05 |
| **Total** | | | | | | |
| SNN | 188,992 | 8,768 | 0.5071 | 31,293,570 | 15,870,008 | 0.0159 |

Tabela 5.6: Consumul energetic al rețelei spike antrenată coeficienți cepstrali

# Capitolul 6

# Concluzii

În această lucrare, am demonstrat capabilitatea rețelelor neurale spike de a oferi rezultate comparabile cu modelele convenționale de învățare automată, având avantajul unui consum redus de energie în timpul antrenării. Am evaluat eficiența rețelelor neurale în clasificarea accentelor folosind trei arhitecturi distincte: o rețea neurală convoluțională antrenată cu amplitudinile semnalelor, o rețea neurală convoluțională antrenată cu coeficienții cepstrali și o rețea neurală spike antrenată cu coeficienții cepstrali. Fiecare rețea a fost evaluată atât din punct de vedere al performanței, cât și al consumului energetic.

În ceea ce privește performanța, cele trei arhitecturi au demonstrat rezultate remarcabile, cu diferențe relativ mici între ele. Cele mai bune rezultate au fost obținute de rețeaua convoluțională antrenată cu amplitudinile semnalelor, oferind cea mai mare acuratețe, de 98.4%. Următoarea a fost rețeaua convoluțională antrenată cu coeficienții cepstrali, cu o acuratețe de 98.3%. Deși rețeaua neurală spike a obținut cea mai mică acuratețe, de 96.5%, aceasta a fost foarte apropiată de primele două.

Din punct de vedere al eficienței energetice, evaluările au arătat următoarele rezultate: 70,875 Jouli pentru antrenarea rețelei convoluționale cu amplitudinile semnalului audio, 910,043.5 Jouli pentru antrenarea rețelei convoluționale cu coeficienți cepstrali și doar 44.8 Jouli pentru antrenarea rețelei neurale spike. Aceste cifre evidențiază avantajul semnificativ al rețelelor neurale spike în ceea ce privește consumul de energie. Mai precis, rețeaua neurală spike este de aproximativ 1581 de ori mai eficientă energetic decât rețeaua convoluțională antrenată cu amplitudinile semnalului audio și de aproximativ 20313 de ori mai eficientă decât rețeaua convoluțională antrenată cu coeficienți cepstrali.

În concluzie, am demonstrat că rețelele neurale spike reprezintă o alternativă viabilă și sustenabilă la modelele convenționale de învățare automată, oferind un echilibru excelent între performanță și eficiență energetică. Aceste rețele deschid noi oportunități pentru cercetare și aplicabilitate practică într-o gamă largă de domenii tehnologice, promițând soluții inovatoare și mai eficiente energetic.

# Bibliografie


[1] Stanford Institute for Human-Centered Artificial Intelligence (HAI), *Artificial Intelligence Index Report*, rap. teh., Accesat în mai 2024, 2024, URL: https://aiindex.stanford.edu/report/.

[2] *OpenAI's CEO Says the Age of Giant AI Models Is Already Over*, Articol pe Wired, 2023, URL: https://www.wired.com/story/openai-ceo-sam-altman-the-age-of-giant-ai-models-is-already-over/.

[3] Asianometry, *Why Brain-Like Computers Are Hard*, Accesat în mai 2024, 2023, URL: https://www.youtube.com/watch?v=FegeRT5N3A4.

[4] Eric Woodell, *Von Neumann Architecture Diagram*, https://www.linkedin.com/pulse/hidden-trap-gpu-based-ai-von-neumann-bottleneck-dr-eric-woodell-n9m9e, Accesat în mai 2024.

[5] MathWorks, *dsp.STFT*, Accesat în mai 2024, URL: https://www.mathworks.com/help/dsp/ref/dsp.stft.html.

[6] Valerio Velardo - The Sound of AI, *Speech process*, accesat pe 23 mai 2024, URL: https://www.youtube.com/watch?v=4_SH2nfbQZ8&t=2231s&ab_channel=ValerioVelardo-TheSoundofAI.

[7] J. K. E. Shraghian și al., *Training Spiking Neural Networks Using Lessons from Deep Learning*, https://arxiv.org/pdf/2109.12894, Pagina 7, accesat pe 20 mai 2024.

[8] J. K. E. Shraghian și al., *Training Spiking Neural Networks Using Lessons from Deep Learning*, https://arxiv.org/pdf/2109.12894, Pagina 7, accesat pe 21 mai 2024.

[9] J. K. E. Shraghian și al., *Training Spiking Neural Networks Using Lessons from Deep Learning*, https://arxiv.org/pdf/2109.12894, Pagina 8, accesat pe 24 mai 2024.

[10] Jason K. Eshraghian, *TUTORIAL 2 - THE LEAKY INTEGRATE-AND-FIRE NEURON*, accesat pe 25 mai 2024, URL: https://snntorch.readthedocs.io/en/latest/tutorials/tutorial_2.html.

[11] *An Open Domain-Extensible Environment for Simulation-Based Scientific Investigation (ODESSI) - Scientific Figure on ResearchGate*, accesat pe 23 mai 2024, URL: https://www.researchgate.net/figure/The-main-structur-of-a-neuron-consists-of-soma-dendrites-and-axon_fig3_220856618.

[12] Afroz Ahamad, Ankit Anand și Pranesh Bhargava, „AccentDB: A Database of Non-Native English Accents to Assist Neural Speech Recognition", în *Proceedings of The 12th Language Resources and Evaluation Conference*, Marseille, France: European Language Resources Association, Apr. 2020, pp. 5353–5360, URL: https://www.aclweb.org/anthology/2020.lrec-1.659.