Laboratorul 3
Modele moderne pentru Recunoașterea Automată a Vorbirii (ASR)

Obiective:

- Înțelegerea conceptului de modelare end-to-end în ASR și a avantajelor sale față de sistemele hibride (DNN-HMM).
- Utilizarea unor modele ASR pre-antrenate (ex: Wav2Vec 2.0, Whisper) pentru transcrierea fișierelor audio.
- Calcularea și interpretarea metricii Word Error Rate (WER) pentru evaluarea cantitativă a performanței ASR.
- Compararea performanței a două modele ASR pe același set de date și analiza tipurilor de erori (substituții/ștergeri/inserții).
- Familiarizarea cu conceptele de identificare și verificare a vorbitorului și cu metrica Equal Error Rate (EER).
- Conectarea modelelor secvențiale (RNN/LSTM) la sarcini de clasificare în vorbire (bonus, opțional).

Resurse necesare:

- Un mediu de programare Python (ex: Jupyter Notebook, Google Colab) cu acces la internet.
- Bibliotecile Python: transformers, datasets, torch, torchaudio, librosa, soundfile.
- Biblioteca jiwer pentru calculul WER.
- Un set de date audio cu transcrieri de referință (ex: Mini LibriSpeech, Common Voice, Speech Commands) sau un set propriu de 10-15 fișiere .wav + transcrieri corecte.

Sarcini:

1. Configurarea Mediului de Lucru:
    - Instalați bibliotecile necesare (în Colab se recomandă instalarea în prima celulă).

Exemplu comandă:

pip install transformers datasets torch torchaudio soundfile librosa jiwer

2. Transcriere Audio cu un Model ASR Pre-antrenat (Wav2Vec 2.0):
    - Selectați un fișier audio local (ex: test.wav) și asigurați-vă că este mono. Re-eșantionați la 16 kHz dacă este necesar.
    - Încărcați un model ASR din Hugging Face și generați transcrierea.

Exemplu (Wav2Vec2):

```
from transformers import pipeline
import librosa

transcriber = pipeline("automatic-speech-recognition", model="facebook/wav2vec2-base-960h")
speech, sr = librosa.load("test.wav", sr=16000)
```

out = transcriber(speech)
print(out["text"])

3. Compararea a două modele ASR:
    - Alegeți un al doilea model ASR (ex: openai/whisper-base).
    - Rulați ambele modele pe același set de 10-15 fișiere audio și salvați transcrierile în două liste (ipoteze_model1 și ipoteze_model2).
    - Păstrați transcrierile de referință (ground truth) într-o listă separată (referinte).
4. Evaluarea WER cu *jiwer*:
    - Instalați și importați biblioteca *jiwer*.
    - Calculați WER pentru fiecare model, atât per fișier, cât și pe toate exemplele.
5. Analiza calitativă a erorilor:
    - Identificați 2-3 exemple unde ambele modele greșesc și descrieți tipul erorilor (substituții/ștergeri/inserții).
    - Identificați 2-3 exemple unde un model este corect iar celălalt greșește; explicați posibil motivul (zgomot, nume proprii, accent, viteză de vorbire etc.).
    - Scrieți un scurt raport (1-2 paragrafe) cu concluzii: care model e mai robust pe datele alese și ce tipuri de erori domină.
6. Recunoașterea vorbitorului (opțional):
    - Selectați 2-3 fișiere audio de la același vorbitor și 2-3 de la un alt vorbitor (pot fi înregistrări proprii).
    - Utilizați un model pre-antrenat de embedding-uri de vorbitor (ex: SpeechBrain) pentru a extrage vectori și calculați similaritatea (cosine).
    - Discutați cum ați seta un prag de decizie pentru verificare (1:1) și cum ați calcula EER pe un set mai mare.
7. LSTM pe o sarcină de clasificare (opțional):
    - Folosiți MFCC-urile din Laboratorul 2 ca intrare și construiți un clasificator simplu pentru cuvinte cheie (ex: subset *Speech Commands*).
    - Implementați un model RNN și un model LSTM (PyTorch/TensorFlow) și comparați acuratețea finală și curbele de învățare.
    - Interpretați diferențele în contextul problemei vanishing gradient și al dependențelor pe termen lung.

Întrebări de Reflecție:

- De ce modelele end-to-end simplifică pipeline-ul clasic (model acustic + dicționar + model lingvistic), dar pot necesita mai multe date și resurse?
- Ce tip de erori (S/D/I) apar cel mai frecvent în experimentele dumneavoastră și de ce credeți că se întâmplă acest lucru?
- În ce situații un sistem de verificare a vorbitorului poate eșua chiar dacă WER-ul unui ASR este bun?
- De ce LSTM/GRU sunt, în general, mai potrivite decât un RNN simplu pentru secvențe lungi?

Test Formativ 3

Secțiunea I – Definiții și Concepte

1. Care este principalul avantaj al unui model ASR end-to-end față de un sistem hibrid DNN-HMM în procesul de dezvoltare?
2. Ce rol joacă simbolul „blank” (ϵ) în funcția de pierdere Connectionist Temporal Classification (CTC)?

Secțiunea II – Modele Statistice și Aliniere

3. Numiți cele trei tipuri de probabilități care definesc un model HMM, notate de obicei ca $\lambda = (A, B, \pi)$.
4. Asociați fiecare problemă HMM cu algoritmul corespunzător:

Problemă	Algoritm
1. Evaluare (Calculul verosimilității)	a) Baum-Welch
2. Decodare (Găsirea secvenței de stări)	b) Forward
3. Învățare (Antrenarea parametrilor)	c) Viterbi

Secțiunea III – Arhitecturi Neuronale pentru Secvențe

5. Care este principala limitare a unei Rețele Neuronale Dense (DNN) în procesarea vorbirii, pe care o rezolvă o Rețea Neuronală Recurentă (RNN)?
6. Numiți cele trei porți (gates) fundamentale ale unei celule LSTM și descrieți pe scurt rolul fiecăreia.

Secțiunea IV – Evaluare și Recunoașterea Vorbitorului

7. Diferențiați între verificarea vorbitorului și identificarea vorbitorului, menționând tipul de potrivire (1:1 vs. 1:N) pentru fiecare. Ce reprezintă metrica Equal Error Rate (EER)?
8. Problemă de calcul WER: Referință: „mergem la film mâine seară” (6 cuvinte). Ipoteză: „mergem film mâine”. Calculați numărul de substituții (S), ștergeri (D), inserții (I) și valoarea WER.

Secțiunea V – Reflecție Scurtă

9. Explicați pe scurt diferența fundamentală dintre calculul efectuat de algoritmul Viterbi și cel Forward (în contextul HMM).