Tema 2 – Învățare Automată

v1.0

1. Introducere/Descriere Generală

Scopul este acela de a implementa un flux complet de analiză și procesare pentru rezolvarea unor sarcini de clasificare de imagini și analiza de sentiment pe text în limba română utilizând rețele neurale pentru consolidarea înțelegerii antrenării, testării, analizei și îmbunătățirii modelelor.

2. Cerințe

2.1 Partea 1 – Clasificare de imagini [5p]

În această prima parte veți rezolva două sarcini de clasificare de imagini, folosind două seturi de date furnizate, și veți analiza dificultățile specifice antrenării modelelor pentru date vizuale, explorând strategii de îmbunătățire a performanței în regimuri diferite de date.

Pentru problemele de clasificare veți folosi următoarele seturi de date:

- Imagebits – imaginile au rezoluția de 96×96 pixeli, color (RGB). Setul conține 10 clase: airplane, bird, car, cat, deer, dog, horse, monkey, ship, truck. Sunt disponibile 800 de imagini de antrenare și 500 de imagini de test per clasă (8.000 antrenare, 5.000 test în total).
- Land patches – un set de date bazat pe imagini satelitare, proiectat pentru clasificarea utilizării și acoperirii terenului (land use / land cover). Imaginile au rezoluția de 64×64 pixeli și sunt disponibile în varianta RGB (3 canale). Setul conține 10.000 de imagini etichetate și distribuite în 10 clase: AnnualCrop, Forest, HerbaceousVegetation, Highway, Industrial, Pasture, PermanentCrop, Residential, River, SeaLake.

Explorarea datelor [1p]

Pentru fiecare set de date realizați o scurtă analiză care să includă (dar nu este limitată la): echilibrul claselor, variabilitatea intra- și inter-clasă (e.g., analiză calitativă prin vizualizarea unor exemple din fiecare clasă), particularități relevante pentru antrenare.

Antrenarea de rețele neurale [4p]

Pentru sarcinile propuse implementați și antrenați **cel puțin două tipuri de arhitecturi**:

- arhitectura de tip **MLP** (Multi-Layer Perceptron) [2p]
- arhitectura de tip **CNN** (Convolutional Neural Networks) [2p]

Explorați modalități de îmbunătățire a performanței, având în vedere că aceasta este influențată nu doar de arhitectură, ci și de alegerea hiperparametrilor (e.g., optimizator, learning rate, batch size, regularizare), folosirea de straturi de normalizare (ex. BatchNorm), de straturi de tip pooling, tehnicile de augmentare folosite etc.

Pentru augmentări, puteți urmări indicații și exemple din documentația bibliotecii Albumentations pentru task-ul de clasificare de imagini, fiind o bibliotecă ce înglobează multe tipuri de augmentări (atât standard, cât și specializate) pentru lucrul cu date vizuale.

ATENȚIE: În raportul final trebuie să dați o justificare fiecărei alegeri arhitecturale, de augmentare sau de optimizare făcute. Altfel spus, trebuie să spuneți ce problemă avea antrenarea voastră, care v-a făcut să schimbați arhitectura, augmentările folosite, sau configurația optimizatorului.

Pentru augmentări, arătați pe același grafic rezultatele curbelor de loss sau performanță, rezultate în urma folosirii vs. nefolosirii augmentărilor în cauză.

În vederea îmbunătățirii performanței pentru setul de date cu suport redus (Land patches) pentru clasele de interes, puteți investiga dacă pornirea de la o inițializare diferită de una aleatorie este de ajutor. În acest sens, urmăriți tutoriale care vă indică procedura generală de fine-tuning (e.g. acest tutorial din PyTorch), și explorați fine-tuning-ul rețelei antrenate de voi de la setul de date Imagebits la cel cu Land patches. De ținut cont că pentru fine-tuning-ul unui model de la un set de date la altul, rezoluția imaginilor de intrare trebuie să fie identică.

ATENȚIE: **În rezolvarea sarcinilor propuse nu este permisă folosirea de backbone-uri pre-antrenate pe seturi de date externe!**

2.2 Partea 2 - Analiza de Sentiment pe Text in Română [5p]

În a doua parte a temei veți rezolva o sarcină de analiză de sentiment pe text în limba română, folosind rețele neurale recurente pentru a clasifica recenzii în 2 categorii de sentiment: pozitiv sau negativ.

Pentru problema de clasificare veți folosi setul de date *ro_sent*, disponibil pe HuggingFace. Acest set de date conține recenzii de produse și filme scrise în limba română, fiecare etichetată cu sentiment (pozitiv sau negativ). Setul de date include 17.941 de exemple pentru antrenare și 11.005 exemple pentru testare. Link către setul de date: https://huggingface.co/datasets/dumitrescustefan/ro_sent
Link descărcare:

- https://raw.githubusercontent.com/dumitrescustefan/Romanian-Transformers/examples/examples/sentiment_analysis/ro/train.csv

- https://raw.githubusercontent.com/dumitrescustefan/Romanian-Transformers/examples/examples/sentiment_analysis/ro/test.csv

Explorarea Datelor [1p]

1. Analiza echilibrului de clase

Realizați un grafic al frecvenței de apariție a fiecărei etichete (pozitive / negative) în setul de date de antrenare / test, folosind bar plot / count plot.

2. Statistici despre text

Afișați distribuția lungimii textelor (număr de cuvinte sau caractere) pentru fiecare clasă de sentiment. Identificați și vizualizați cele mai frecvente cuvinte pentru fiecare clasă.

Notă: Diagramele din această secțiune sunt cele minimal cerute: Nu sunt singurele pe care le puteți face.

Tokenizare și Embedding Layer [1p]

Implementați procesul de preprocesare a textului și embedding:

- Curățarea datelor: Primul pas constă în preprocesarea textului brut - eliminarea caracterelor speciale, normalizarea textului, eliminarea stopwords (opțional), etc.
- Tokenizare: Transformați textele în secvențe de token-uri (indici numerici). Specificați vocabularul utilizat și cum sunt tratate cuvintele necunoscute. Hint: spacy (Tokenization Using Spacy - GeeksforGeeks)
- Embedding Layer: Importati un model de embedding care transformă indicii în vectori denși. Hint: fasttext (How to Use Pretrained FastText Word Vectors for English fxis.ai)
- Padding: Normalizați lungimea secvențelor la o lungime fixă.

Utilizarea modelelor de Rețele Neurale Recurente [3p]

Sunt propuse spre evaluare următoarele arhitecturi de rețele neurale recurente:

- Arhitectură de tip RNN simplu [1.5p]
  - Experimentați cu numărul de straturi, dimensiunea acestora și dimensiunea hidden state-ului.

- Arhitectură de tip LSTM [1.5p]
  - Puteți genera propria voastră arhitectură explorând:
    - Folosind straturi LSTM unidirecționale sau bidirecționale
    - Numărul de straturi LSTM și dimensiunea hidden state-ului
    - Combinarea cu straturi liniare pentru partea finală a rețelei

Explorați modalități de îmbunătățire a performanței, având în vedere că aceasta este influențată nu doar de arhitectură, ci și de alegerea hiperparametrilor (e.g., optimizator, learning rate, batch size, regularizare), folosirea de straturi de normalizare (ex. BatchNorm), de straturi de tip pooling, tehnicile de augmentare folosite etc.

Pentru îmbunătățirea performanței și a capacității de generalizare a modelului, explorați tehnici de augmentare specifice datelor text. Câteva tehnici recomandate:

- Random Swap/Delete/Insert: Operații aleatorii asupra cuvintelor din propoziție
- Back-Translation: Traducerea textului în engleză și înapoi în română pentru a genera variații
- Contextual Word Embeddings: Folosirea unui model BERT românesc pentru înlocuirea contextuală a cuvintelor

Pentru augmentări, puteți urmări indicații și exemple din: https://neptune.ai/blog/data-augmentation-nlp

ATENȚIE: În raportul final trebuie să dați o justificare fiecărei alegeri arhitecturale, de augmentare sau de optimizare făcute. Altfel spus, trebuie să spuneți ce problemă avea antrenarea voastră, care v-a făcut să schimbați arhitectura, augmentările folosite, sau configurația optimizatorului.

Pentru augmentări, arătați pe același grafic rezultatele curbelor de loss sau performanță, rezultate în urma folosirii vs. nefolosirii augmentărilor în cauză.

2.3 Evaluarea modelelor

Raportul final trebuie să includă următoarele:

- Pentru fiecare model antrenat trebuie raportat setup-ul de antrenare:
  - descrierea arhitecturii utilizate;
  - detalierea configurației, incluzând cel puțin: optimizatorul folosit, learning rate (și eventual scheduler), dimensiunea batch-urilor, numărul de epoci de antrenare, metodele de regularizare, precum și orice alți hiperparametri relevanți utilizați.
- Prezentați curbele de loss pentru antrenare și validare pe același grafic, iar separat, pe un alt grafic, curbele pentru cel puțin o metrică relevantă pentru task, tot pentru antrenare și validare.
- Realizați un tabel în care liniile reprezintă configurațiile arhitecturale și de optimizare, iar coloanele includ metricele de performanță (e.g., acuratețe, F1).
- Creați matricea de confuzie corespunzătoare clasificării realizate, cel puțin pentru cea mai bună configurație asociată unui tip de arhitectură.

3. Predarea Temei

Tema va fi încărcată pe Moodle însoțită de un raport sub formă de fișier PDF, care include:

- Vizualizări și diagrame relevante.
- Raportarea completă a evaluării rețelelor antrenate.

Toate rezultatele, cantitative și calitative, trebuie însoțite de interpretări și analiză în text (e.g., care este influența arhitecturii, cât de puternic este impactul hiperparametrilor asupra performanței, care sunt clasele cu cele mai bune predicții). Textul trebuie să includă toate elementele necesare reproducerii setup-ului experimental (e.g., tipuri de augmentări, proceduri de antrenare, valori ale hiperparametrilor).

Rezultatele temei vor fi prezentate în cadrul laboratoarelor, **exclusiv** pe baza rapoartelor încărcate.