**Hate speech Analysis**

**1. Introducere**

Echipa noastră (Carpineanu Alexandru, Cuciureanu Dragos-Adrian, Vasile George-Cristian) a ales tema ”#15 Hate speech, offensive language identification, misogyny / stereotype detection”.

Discursul ce instigă ură - denumit în continuare hate speech - este diferit de libera exprimare, deoarece nu este limitat la exprimarea unor opinii controversate, în schimb implică propagarea discriminării și a violenței.

Hate speech-ul a fost întotdeauna o problemă cu care societatea a trebuit să se confrunte, însă, de la apariția internetului, acesta a fost propulsat în ochii tuturor oamenilor. Orice utilizator putea să întâlnească vorbe ofensive fără sa vrea.

Problema hate speech-ului este prevăzută în legile multor țări, deși interpretarea acesteia poate fi variată și identificarea sa mai dificilă. În special în mediul online, unde volumul la care sunt transmise informațiile, filtrarea canalelor de convorbiri și restricționarea accesului pentru entitățile ce vorbesc cu ură este o problemă ce necesită o soluție eficientă.

Majoritatea platformelor sociale folosesc deja inteligența artificială pentru a detecta hate speech-ul și a-l cenzura automat, însă mereu se pot aduce îmbunătățiri acestor modele pentru a ne asigura că cenzurarea se aplică doar asupra hate speech-ului, orice greșeală a modelului putând să însemne o încălcare a dreptului la liberă exprimare a utilizatorilor.

**2. Setul de date**

Pentru alegerea setului de date am folosit platforma Hugging Face datorită numărului variat de opțiuni pe care îl oferă. Am ales setul de date “tweets\_hate\_speech\_detection”.

Am ales platforma Twitter deoarece postările care puteau fi făcute pe aceasta până recent erau limitate la 280 caractere, astfel tweet-urile fiind un mijloc simplu și bun pentru detectarea hate speech-ului.

În contextul nostru, am definit problema clasificării unui tweet ca fiind unul de hate speech prin identificarea în conținutul unor texte ce au conotații rasiste și/sau sexiste.

Astfel, fiecare tweet din setul de date este etichetat cu 0 sau cu 1, eticheta 1 semnificând prezența rasismului/sexismului. Cele 31.962 de tweet-uri valabile sunt scrise majoritar în limba engleză. Dintre acestea doar 2.242 sunt clasificate cu 1, astfel tweeturile cu 0 fiind predominante. Pentru acest fapt, am selectat doar 10.000 dintre acestea.

Din punct de vedere al impărțirii datelor, am selectat 60% pentru antrenare si 40% pentru testare (ponderile fiind ușor de modificat, dacă se dorește mărirea setului de antrenare și diminuarea celui de testare).

Totodată, setul de date a fost preprocesat pentru a ajunge la o formă normalizată potrivită pentru clasificatori.

Pentru preprocesare am aplicat următoarele operații:

* Transpunerea textelor unicode în caractere cu cod ASCII.
* Transformarea literelor mari în litere mici, deoarece această caracteristică a lor nu este relevantă în procesul nostru.
* Înlăturarea hyperlinkurilor, ele fiind adrese către resurse web și neprezentând nicio semnificație lingvistică.
* Eliminarea numelor de utilizator, identificate prin caracterul “@” urmat de un cuvant format doar din litere, numere și simbolul “\_”.
* Eliminarea hashtag-urilor. În postări de hate speech hashtag-urile pot fi utilizate pentru a organiza întâlniri bazate pe promovarea unor ideologii rasiste sau instigative la ură, poluand analiza conținutului propriu-zis al postarii.
* Înlăturarea punctuatiei, neavând o influența în identificarea hate speech-ului.
* Înlăturarea caracterelor numerice.
* Normalizarea caracterelor repetate. Utilizatorii serviciilor de chat au tendința de a multiplica vocale sau uneori și consoane din diferite cuvinte pentru a întări anumite idei. În orice caz, aceasta forma a mesajelor nu este relevantă în identificarea hate speech-ului, de asemenea pot îngreuna procesul de analiză al modelelor prin nerecunoașterea anumitor cuvinte.
* Lematizarea, având rolul de a reduce varietatea lingvistică redundantă prin proiecția diferitelor forme ale cuvintelor (conjugări, declinări, articulații) în forma lor de baza.
* Eliminarea stopwords. Acestea reprezintă cele mai comune cuvinte dintr-o limbă și în general sunt prepozitii, conjunctii, articole, neavând sens de sine stătător.
* Eliminarea “caracterelor albe”, precum “space”, “tab”, “enter”.

**3. Extragerea feature-urilor**

Am ales două metode de a extrage feature-urile, **TF-IDF** (Term Frequency - Inverse Document Frequency) și **CBOW** (Continuous Bag of Words). Am antrenat și testat fiecare model pe ambele tipuri de feature.

1. **CBoW**

Continuous Bag of Words este o metoda de extragere a feature-urilor ce utilizeaza o retea neuronala pre-antrenata cu 2 layere, care reconstruieste contextul cuvintelor primite ca input dupa vectorii creati in urma antrenarii.

Aceasta ajusteaza n-vectorii, astfel incat cuvintele gasite in contexte similare apar apropiate unele de altele in spatiul vectorial creat la antrenare.

1. **Tf-Idf**

Term frequency - Inverse document frequency este un mod de extragere a feature-urilor ce evalueaza importanta fiecarui cuvant din corpus, in relatie cu numarul sau de aparitii in fiecare document.

Spre exemplu, cuvinte precum ‘the’ sau ‘is’ au o importanta mult mai mica fata de ‘blacklives’ intr-un corpus de hate-speech.

Tf-Idf nu ia in considerare contextul in care regasim cuvintele in propozitie.

**4. Modele de clasificatori**

Am ales o gamă variată de modele, pentru a observa diferențele dintre ele. Mai mult, am folosit și cinci modele pre antrenate preluate tot de pe platforma HuggingFace pentru a avea un spațiu de comparatie mai mare.

1. **Multinomial Naive Bayes**

Un algoritm probabilist care utilizează teorema lui Bayes ce presupune independența caracteristicilor pentru a decide clasa din care face parte input-ul.

Parametrii:

* **alfa**: Un parametru de netezire. Ajută la evitarea overfitting ului.

Rezultate:

|  |  |  |
| --- | --- | --- |
| ***MultinomialNB*** | **accuracy** | **macro f1** |
| **Tf-Idf** | 0.84 | 0.57 |
| **CBoW** | 0.84 | 0.62 |

1. **Ridge Classifier**

Clasificator binar care folosește regresia Ridge. Acesta penalizează complexitatea pentru a evita overfitting ul.

Parametrii:

* **alfa**: Utilizat pentru a controla puterea regularizarii. Reduce variația aproximărilor.
* **solver**: Algoritmul folosit pentru a rezolva problema de optimizare asociată modelului.

Rezultate:

|  |  |  |
| --- | --- | --- |
| ***Ridge Classifier*** | **accuracy** | **macro f1** |
| **Tf-Idf** | 0.93 | 0.86 |
| **CBoW** | 0.87 | 0.71 |

1. **SVC (kernel ‘rbf’)**

Un algoritm de clasificare binară ce funcționează prin transformarea inputului într-un spațiu dimensional mai mare folosind funcții neliniare numite funcții radiale de baza (RBF) kernel. În acest spațiu mai mare, algoritmul încearca sa găsească un hyperplan optim care separa cele doua clase de date maximizand separarea dintre clase (distanta minima dintre un element al primei clase și un element al celei de-a doua). Câțiva parametrii cheie sunt:

* **C**: Parametrul de regularizare, ce determină balanța dintre a obține o margine mai mare și permiterea unor clasificări eronate. O valoare mai mică a lui C permite o margine mai mare, iar o valoare mai mică a sa implică o clasificare mai strictă.
* **gamma** (coeficientul kernel): Defineste influența unui singur exemplu de antrenament asupra liniei de decizie. Un gamma mai mic înseamnă o influență mai largă și poate rezulta într-o linie de decizie mai uniforma, in timp ce un gamma mare inseamna o influenta mai restransa si poate duce la o linie de decizie mai complexă și mai ondulată.
* **kernel**: Alegerea kernelului poate avea un impact semnificativ asupra performanței algoritmului. RBF SVC utilizează implicit kernelul RBF, dar pot fi utilizate și alte kerneluri, cum ar fi cel liniar, polinomial sau sigmoidal. Fiecare kernel are proprii sai hyperparametri, cum ar fi gradul pentru kernelurile polinomiale sau coef0 pentru kernelurile sigmoidale, care controlează comportamentul acestora.

Rezultate:

|  |  |  |
| --- | --- | --- |
| ***SVC - rbf*** | **accuracy** | **macro f1** |
| **Tf-Idf** | 0.91 | 0.83 |
| **CBoW** | 0.88 | 0.75 |

1. **Linear SVC**

Algoritm de clasificare folosind masini cu vectori suport. Se ocupă cu căutarea hyperplanelor care separă clasele de date într-un spațiu dimensional mai mare. Este asemanator cu RBF SVC, însă folosește un kernel linear, iar hyperplanul rezultat este o linie dreaptă.

Rezultate:

|  |  |  |
| --- | --- | --- |
| ***Linear SVC*** | **accuracy** | **macro f1** |
| **Tf-Idf** | 0.93 | 0.88 |
| **CBoW** | 0.87 | 0.74 |

1. **MLP Classifier**

Constă din mai multe straturi de neuroni artificiali. Orice rețea neuronală are un strat de intrare. Pe lângă acestea doua, un MLP are un număr variat de straturi ascunse. Fiecare neuron primește input de la un alt neuron dintr-un strat precedent, îl modifică și îl trimite mai departe până la stratul de ieșire, unde se obține rezultatul final al rețelei.

Parametrii:

* **solver**: Algoritmul care rezolvă problema de optimizare
* **alfa**: Puterea termenului de regresie Ridge
* **hidden\_layer\_sizes**: Numărul de neuroni din straturile ascunse
* **max\_iter**: Numărul maxim de iterații
* **random\_state**: Determină generarea de numere aleatoare pentru inițializare

Rezultate:

|  |  |  |
| --- | --- | --- |
| ***MLP Classifier*** | **accuracy** | **macro f1** |
| **Tf-Idf** | 0.93 | 0.88 |
| **CBoW** | 0.85 | 0.74 |

1. **XGBoost**

Extreme Gradient Boosting este un algoritm proiectat pentru a optimiza performanța modelelor de gradient boosting ce folosește combinații de tehnici de boosting și regularizare. XGBoost îmbunătățește algoritmii tradiționali de gradient boosting prin introducerea mai multor inovații cheie: Optimizarea bazată pe gradient, regularizarea, tăierea arborilor, tratarea valorilor lipsa. Este eficient în rezolvarea problemelor de clasificare, regresie și clasificare a rangului.

Parametrii:

* **eta** (sau learning\_rate) are rolul de a preveni overfittingul
* **gamma** (sau min\_split\_loss) specifica reducerea minima a pierderii necesara pentru a produce o divizare
* **max\_depth**: adâncimea maximă a unui arbore. Este folosit în controlul overfitting-ului, intrucat adâncimea mai mare va permite modelului sa invete relații foarte specifice unui sample particular. Crescand aceasta valoare modelul devine mai complex și este foarte probabil sa produca overfitting.
* **subsample**: rata de subsampling a instanțelor de învățare. Valori mai mici fac algoritmul mai conservator si previn overfittingul, insa valori prea mici pot conduce la underfitting.
* **colsample\_bytree** este rata de subsampling a coloanelor cand se construiește fiecare arbore. Subsampling-ul are loc odată pentru fiecare arbore construit.
* **n\_estimators**: numărul de estimatori

Rezultate:

|  |  |  |
| --- | --- | --- |
| ***XGBoost*** | **accuracy** | **macro f1** |
| **Tf-Idf** | 0.91 | 0.84 |

1. **Deep Neural Network**

Retelele Neuronale, cunoscute si ca Retele Neuronale Artificiale sau Retele Neuronale Simulate, sunt “inima” algoritmilor de invatare in profunzime (deep learning). Structura si numele lor sunt inspirate de creierul uman, imitand modul in care neuronii transmit semnale de la unul la altul.

In compozitia lor, retelele neuronale artificiale au straturi de noduri (sau neuroni), dintre care unul este stratul de input, unul de output si celelalte sunt staturile ascunse. Fiecare nod este conectat la altul si are asociata o pondere si un prag. Daca outputul unui neuron depaseste valoarea pragului, neuronul este activat si trimite informatia la urmaorul neuron din retea, aflandu-se in urmatorul strat.

Retelele neuronale se bazeaza pe invatarea cu date de antrenare si isi imbunatatesc capacitatile treptat, insa odata rafinati, devin aparate puternice de mare viteza de clasificare si grupare in clustere a datelor.

Un model bazat pe invatarea in profunzime este Word2vec, recunsocut pentru utilizarile sale in NLP. El invata asocierile dintre grupuri mari de cuvinte, utilizand un vector numeric pentru retinerea fiecarui cuvant, redand similaritatea acestora. Printre algoritmii utilizati se afla skip-gram si CBOW.

Parametrii:

* **vector\_size**: Dimensiunea vectorilor de cuvinte
* **feature\_size**: Dimensiunea vectorului de feature-uri
* **workers**: Determina numarul de threaduri folosite pentru antrenarea modelului
* **epochs**: Numarul de epoci in care se itereaza textul
* **total\_examples**: Numarul total de propozitii

Rezultate:

|  |  |  |
| --- | --- | --- |
| ***Deep Neural Network*** | **accuracy** | **macro f1** |
| **N/A** | 0.91 | 0.83 |

1. **Modele preantrenate**

* <https://huggingface.co/patrickquick/BERTicelli?text=Keep+up+the+good+hard+work>

Obtinut prin rafinarea modelului BERT(Bidirectional Encoder Representations from Transformers) cu datele de antrenament din acest set de date: [Offensive Language Identification Dataset](https://scholar.harvard.edu/malmasi/olid)

* <https://huggingface.co/MilaNLProc/bert-base-uncased-ear-mlma>

O alta rafinare a modelului BERT, pe acest dataset: [MLMA](https://aclanthology.org/D19-1474/), folosind Regularizarea atentiei bazata pe entropie

* <https://huggingface.co/facebook/roberta-hate-speech-dynabench-r4-target>
* <https://huggingface.co/Hate-speech-CNERG/bert-base-uncased-hatexplain>

Forma de baza a modelului BERT

* <https://huggingface.co/Hate-speech-CNERG/dehatebert-mono-english>

O alta rafinare pentru limba engleza a modelului BERT

**Referințe**

* [Entropy-based Attention Regularization Frees Unintended Bias Mitigation from Lists](https://aclanthology.org/2022.findings-acl.88.pdf)
* [Learning from the Worst: Dynamically Generated Datasets to Improve Online Hate Detection](https://arxiv.org/pdf/2012.15761.pdf)
* [HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection](https://arxiv.org/pdf/2012.10289.pdf)
* [Deep Learning Models for Multilingual Hate Speech Detection](https://arxiv.org/pdf/2004.06465.pdf)
* [Whose Opinions Matter? Perspective-aware Models to Identify Opinions of Hate Speech Victims in Abusive Language Detection](https://arxiv.org/pdf/2106.15896.pdf)