ROMÂNIA
MINISTERUL APĂRĂRII NAŢIONALE
ACADEMIA TEHNICĂ MILITARĂ
„FERDINAND I”

FACULTATEA DE SISTEME INFORMATICE ȘI SECURITATE CIBERNETICĂ
Specializarea: Calculatoare și sisteme informatice pentru apărare și securitate națională

SISTEM DE INDEXARE, CLASIFICARE SI CAUTARE STIRI

CONDUCĂTOR ŞTIINŢIFIC:
Ș.L. dr. ing. Cristian CHILIPIREA

ABSOLVENT:
Stud. Sg. Maj. Ciprian-George PEȘU

Conține _____ file
Inventariat sub nr. _____
Poziția din indicator: ____
Termen de păstrare: _____

BUCUREŞTI

2022

ABSTRACT

The media plays an important role in shaping the public opinion. In today's society, it is known that media outlets report news in a biased way, affecting the beliefs of news consumers and changing their behavior. Because it is difficult to define standards by which bias can be measured, the public does not have a metric to make them aware that they are influenced.

Various organizations have tried over the years to quantify the confidence levels of publications, such scores are often given by human evaluators who are in turn influenced by their own subjectivity and political beliefs. In addition, recent events have shown that certain segments of the population are extremely likely to be influenced by very biased and false news, making them unable to assess the level of trust of a publication. This type of metric may cause the public to be skeptical about the results of the evaluations.

The solution proposed in this paper consists in presenting users with multiple sources of information from different points of the political spectrum and highlighting the differences between their ways of reporting different events. In order to eliminate the subjectivity introduced by human evaluators, the analysis and classification of data will be done through neural networks developed and tested in the paper.

REZUMAT

Mass-media joacă un rol important în formarea opiniei publice. În societatea curentă este cunoscut faptul că instituțiile de presă raportează știrile într-un mod părtinitor, afectând convingerile consumatorilor de știri și modificându-le comportamentele. Deoarece este dificil să se definească standarde prin care poate fi măsurată părtinirea, publicul nu are la dispoziție o metrică prin care să conștientizeze că este influențat.

Diferite organizații au încercat de-a lungul timpului să cuantifice nivelurile de încredere ale publicațiilor, astfel de scoruri sunt adesea atribuite de evaluatori umani ce sunt influențați la rândul lor de propria subiectivitate și convingeri politice. În plus, evenimentele recente au arătat că anumite segmente ale populației sunt extrem de susceptibile de a fi influențate de știri foarte părtinitoare și false, făcându-le incapabile să evalueze nivelul de încredere a unei publicații. Acest tip de metrică pot determina publicul să fie sceptic cu privire la rezultatele evaluărilor.

Soluția propusă in aceasta lucrare constă în prezentarea utilizatorilor cu surse multiple de informații provenite din puncte diferite ale spectrului politic și evidențierea diferențelor dintre modurile acestora de relatare a evenimentelor. Pentru a elimina subiectivitatea introdusă de evaluatorii umani, analiza și clasificarea datelor se va face prin intermediul unor rețele neuronale dezvoltate și testate în aceasta lucrare.

Cuprins

LISTĂ ABREVIERI

1.	BBC	British Broadcasting Corporation
2.	CNN	Cable News Network
3.	FOX	Fox News
4.	XML	Extensible Markup Language
5.	RSS	Really Simple Syndication
6.	CLI	Command Line Interface
7.	IP	Internet Protocol
8.	API	Application Programming Interface
9.	CPU	Central Processing Unit
10.	PV	Persistent Volume
11.	PVC	Persistent Volume Claim
12.	JSON	JavaScript Object Notation
13.	JDK	Java Development Kit
14.	HTML	HyperText Markup Language
15.	NLTK	Natural Language Toolkit
16.	LSTM	Long short-term memory
17.	BERT	Bidirectional Encoder Representations from Transformers
18.	RNN	Recurrent Neural Network
19.	DNN	Dense Neural Network
20.	CNN	Convolutional Neural Network
21.	GCC	GNU Compiler Collection
22.	REST	Representational State Transfer
23.	NLP	Natural Language Processing

LISTĂ FIGURI

LISTĂ TABELE

LISTĂ ECUAȚII

1 INTRODUCERE

Modul în care o publicație descrie un eveniment poate altera substanțial percepția publicului asupra acestuia. Polarizarea populație prin intermediul surselor media a devenit una dintre principalele amenințări la adresa stabilității și prosperității statelor dezvoltate. Cu toate acestea, mass-media este de multe ori părtinitoare și urmează adesea interesele companiilor private sau ale entităților statele care le controlează.

Articolele de știri online au început să înlocuiască mass-media tradițională scrisă sau radio, ca principala sursă de informații a populației. Datorită disponibilității și ușurinței de accesare a acestor publicații de știri, societatea este vulnerabila la răspândirea știrilor false sau denaturate.

Soluția propusa este reprezentata de o arhitectură de servicii pentru procesarea de știri în limba engleză publicate în internet, preluate prin intermediul unor fluxuri RSS și clasificate prin intermediul unor rețele neuronale. Principalele obiective sunt: construirea unui corpus de date uniform, extracția de entități, teme, sentiment și topice, clasificarea, indexarea într-un motor de căutare și realizarea diferitelor analize statistice. Un flux RSS (Really Simple Syndication) este pus la dispoziție de o publicație și conține un document în formatul XML, actualizat periodic, cu cele mai recente articole publicate.

Principala modalitate prin care sistemul propus combate polarizarea este prezentarea utilizatorului cu surse multiple de informații provenite din puncte diferite ale spectrului politic. La accesarea unei știri, în interfața grafica va fii expusă atitudinea publicației către evenimentul descris precum și relatări ale altor surse de informații despre același subiect sau subiecte similare.

Din punct de vedere arhitectural totalitatea serviciilor se distribuie în 3 subsisteme (de preluare, de clasificare și de vizualizare) interconectate prin intermediul unor cozi de mesaje. Clasificarea se face prin intermediul a doua rețele neuronale dezvoltate și prezentate în aceasta lucrare, ale căror predicții reprezintă scorurile de aparență la diferite categorii de text.

2 RELATED WORK

Soluții existente in domeniul Sentiment Analysis in English Texts

În această lucrare [1] , autorii au implementat și evaluat performanțele diferiților clasificatori algoritmici în clasificarea sentimentului tweet-urilor asupra seturilor de date echilibrate și neechilibrate. Clasificatorii algoritmici utilizați au fost Decision Tree, Naïve Bayes, Random Forest, K-NN, ID3 și Random Tree.

Setul de date inițial utilizat se compune din peste 14000 de tweet-uri distribuite în 3 etichete (pozitiv, negativ si neutru). Acesta a fost împărțit în șase seturi de date, fiecare incluzând tweet-uri referitoare la una dintre cele șase companii aeriene americane (United, Delta, Southwest, Virgin America, US Airways și American). Seturile de date echilibrate au fost obținute prin reducerea numărului de exemple la nivelul clasei cu cele mai puține.

Fiecare dintre seturile de date a fost împărțit în două părți. Prima parte conține 66% din numărul total de tweet-uri și este utilizată pentru a antrena clasificatorul, restul de 34% din tweet-uri au fost folosite pentru a testa predicțiile rezultate.

In primul rând, rezultatele obținute de clasificatoare au variat substanțial in funcție de setul de date utilizat. Toate implementările au obținut performantele cele mai mari pe setul de date US Airways (setul cu cel mai mare număr de exemple) în timp ce calificatoarele antrenare pe Virgin America (setul de dimensiunea cea mai mică) au prezentat cele mai slabe rezultate.

În al doilea rând Naïve Bayes și ID3 au fost în mare parte rezultate mai bune decât restul clasificatoarelor și au obținut aproape același nivel de precizie, iar performanța a fost mai bună pe seturile de date echilibrate. Clasificatori K-NN, Decision Tree, Random Forest și Random Tree au oferit o performanță mai bună pe seturile de date neechilibrate, aceștia obținând performante ridicate pe clasele cu un număr mai mare de exemple.

Fața de clasificatorii prezentați in lucrarea [1], noi urmărim obținerea unor performanțe mai ridicate utilizând un set de date echilibrat [5] de dimensiuni mult mai mari prin intermediul unor rețele neuronale complexe. Acest lucru va permite modelelor dezvoltare să generalizeze predicțiile pe toate tipurile de text, fața de clasificatoarele prezentate, specializate pe recunoașterea sentimentelor din tweet-urile referitoare la o companie aeriana.

Soluții existente in domeniul Classification of News

In aceasta lucrare [2] autorii urmăresc construirea și testarea unui model de clasificare a știrilor pe baza topicelor prezente in text. Setul de date utilizat [3] conține 125.000 de știri provenite de la HuffPost distribuite pe 31 de etichete.

Asupra setului de date este aplicat un proces de curățare si echilibrare, prin combinarea unor categorii de date similare (de exemplu, „Arts” și „Arts and Culture”, „Education” și „College” etc.) și eliminarea categoriilor "Comedy" și "Weird news". În urma acestor modificări, dimensiunea seului de date s-a redus la 113.342 de știri distribuite in 25 de categorii. Preprocesarea datelor a constat în procesele de eliminare ale cuvintelor stopwords, eliminarea semnelor de punctuație și trunchierea cuvintelor. Setul de date a fost împărțit în subseturi de antrenare, validare și testarea cu proporții de 80% , 10% si respectiv 10%.

Autorii au testat o serie de algoritmi de clasificare (Naive Bayes, Multinomial Logistic Regression, Kernel SVM, Random Forest) precum și patru tipuri de rețele neuronale bazate pe straturi convoluționale si LSTM (CNN si RNN). Dintre tehnicile tradiționale de clasificare cele mai mari performante au fost obținute de Logistic Regression cu o acuratețe de 64.10%.

Cele patru rețele neuronale analizate au fost:

- **CNN 2**
- **CNN1-RNN 100**
- **CNN1-RNN 200**
- **CNN2-RNN 200**

*Numărul după CNN denotă numărul de straturi convoluționale iar cel de după RNN denota numărul de neutroni din stratul LSTM.

Cele mai bune rezultate au fost obținute de modelul CNN1-RNN 200 cu o acuratețe de 68.34%. De asemenea, autorii concluzionează că o posibila cauză a performantelor scăzute ale modelelor poate fi reprezentată de subiectivitatea problemei prezentate (o știre este încadrata într-o singură categorie, situație necorespunzătoare cu realitatea). Aceștia consideră că un model bazat pe mai multe straturi LSTM s-ar putea adapta mai bine la acest tip de date.

Soluția noastă își propune sa obțină o acuratețe mai marea a predicțiilor prin utilizarea unei rețele neuronale bazate pe straturi LSTM bidirecționale, după cum a fost propus in lucrarea [2]. Setul de date utilizat reprezintă o versiune mai noua a setului de date prezentat anterior, iar pentru combaterea problemelor de subiectivitate a etichetelor, unei știri i se va atribuii una sau mai multe categorii in funcție de scorurile obținute la predicție.

AllSides

AllSides [10] este o companie americană care evaluează părtinirea politică a instituțiilor media și prezintă diferite versiuni ale știrilor similare din surse politice din dreapta, stânga și centru. Aceasta se concentrează pe publicațiile online și a reușit să evalueze peste 800 de surse de informații pe o scară de cinci puncte: Left, Leans left, Center, Leans right, and Right.

Fiecare sursă este clasificată de voluntari, supravegheați de doi membri ai personalului care au orientări politice diferite. Aceste evaluări sunt verificate ulterior de angajații companiei.

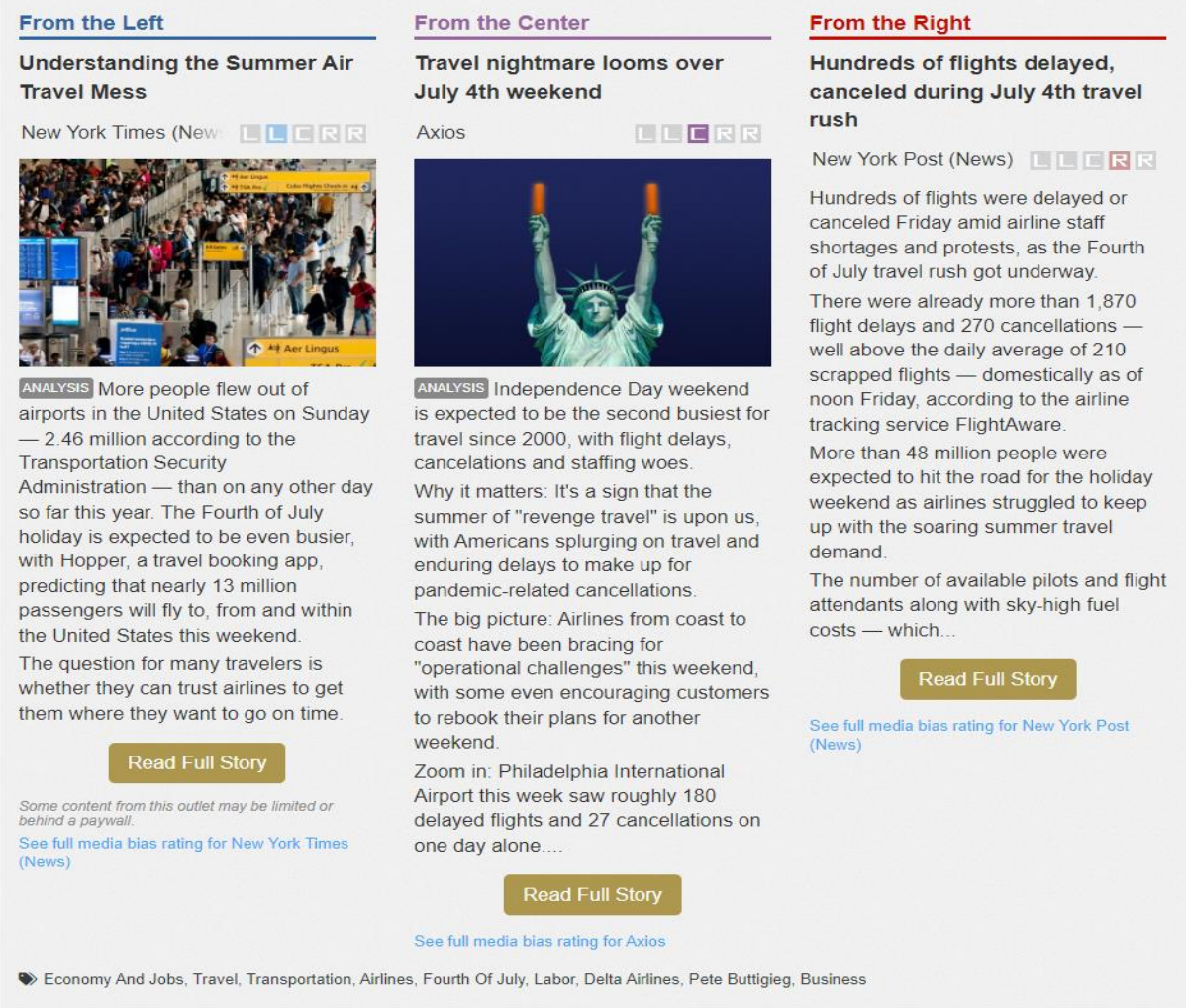

Figura 2.1 Reprezentarea știrilor de către AllSides

Fața de implementarea produsă de noi, AllSides se confruntă cu problema evaluatorilor umani ce sunt influențați de propria subiectivitate și convingeri politice. De asemenea, știrile sunt clasificate după sursă, nu după conținut. Rețelele neuronale utilizate in sistemul propus își bazează predicțiile pe textul extras din știri.

3 TEHNOLOGII UTILIZATE

3.1 ElasticSearch

ElasticSearch este un motor de căutare ce poate indexa date provenite dintr-o varietate de surse, inclusiv log-uri și aplicații web. Acesta permite stocarea, căutarea și analizarea unor volume mari de date, oferind răspunsuri rapide la interogări. Eficiența motorului de căutare se datorează modului in care stochează datele, folosind documente in format JSON și indici.

Documentele sunt unitatea de bază de informații care poate fi indexată în ElasticSearch. Acestea pot fi reprezentate de orice structură codificată în format JSON precum text, numere, vectori sau date binare. Fiecare document are un ID unic (generat de ElasticSearch) și un anumit tip de date ce descrie ce fel de entitate este acesta.

Un index este o colecție de documente cu caracteristici similare. Acesta reprezintă entitatea de cel mai înalt nivel ce poate fi interogată în ElasticSearch. Elementele dintr-un index sunt de obicei legate logic similar cu un tabel dintr-o baza de date relațională.

Definirea si configurarea unui index se face prin intermediul unui Index Template ce permite setarea câmpurilor și a formatului acestora, a priorităților și nivelului de redundanță a datelor. Un tamplate se atribuie unui index pe baza unui structuri regex ce se aplică asupra numelui indexului, numită „index_patterns” și definită la generarea template-ului.

Elasticsearch folosește o structură de date numită index inversat, care este concepută pentru a permite căutări foarte rapide în text. Un index inversat listează fiecare cuvânt unic care apare în orice document și identifică toate documentele în care apare fiecare cuvânt.

Kibana este o interfață web proiectată să funcționeze împreună cu ElasticSearch pentru a valorifica fluxuri de date complexe. Acesta permite analizarea mai rapida a documentelor prin generarea de grafice, diagrame, tabele, histograme și hărți.

In infrastructura prezentata anterior, Kibana are rolul de interfața de management prin intermediul căreia se definesc, se configurează, se vizualizează și se valorifica datele stocate în indecșii implementați în cadrul Elasticsearch și este expusa administratorilor de sistem în rețeaua locala.

3.2 Apache Kafka

Fața de alte soluții existente, precum sockets, Kafka prezinta o modalitate de transmitere a datelor scalabilă și redundantă prin utilizarea mai multor noduri ce lucrează in paralel. Un alt beneficiu al integrării acestei soluții in arhitectura dezvoltată este reprezentat de persistența mesajelor in cazul in care serviciul destinatar nu este disponibil, în implementările precum sockets, mesajele sunt șterse în momentul pierderii conexiunii.

Apache Kafka facilitează transmiterea de mesaje de tipul cheie-valoare ce pot provenii de la mai multe procese numite „producători” către o alta serie de procese numite „consumatori”, proces exemplificat în figura 3.2.

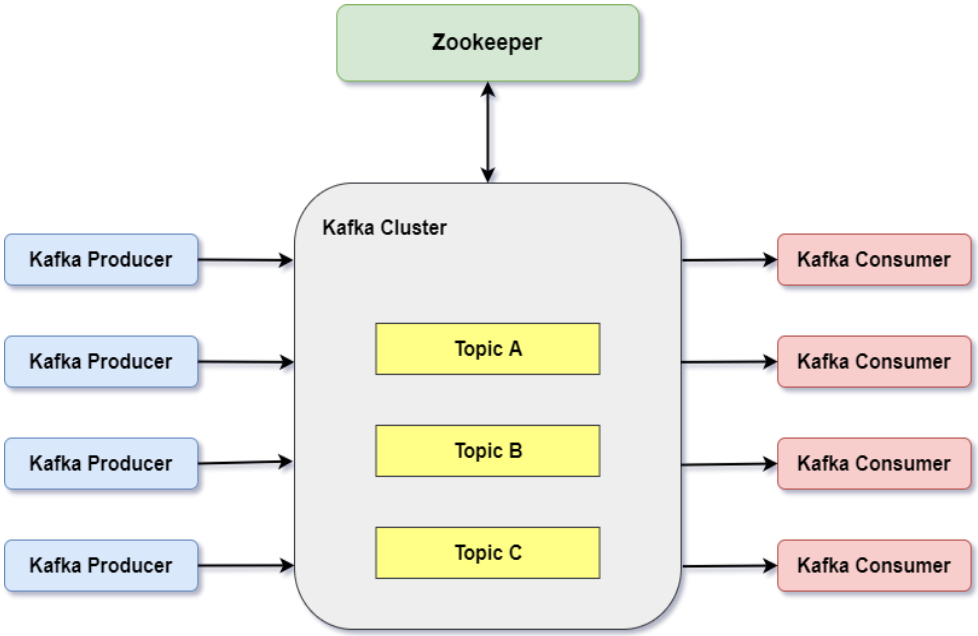

Figura 3.1 Arhitectura de funcționare Apache Kafka

Datele pot fi împărțite în diferite partiții în cadrul diferitelor subiecte (topics). În cadrul unei partiții, mesajele sunt ordonate strict în funcție de offset (poziția unui mesaj în cadrul unei partiții) și sunt indexate și stocate împreună cu un timestamp.

Kafka rulează pe un cluster de unul sau mai multe servere (numite brokeri), iar partițiile tuturor subiectelor sunt distribuite între nodurile clusterului. În plus, partițiile sunt replicate pe mai mulți brokeri pentru a sporii redundanța datelor. Această arhitectură îi permite lui Kafka să livreze fluxuri mari de mesaje într-un mod scalabil.

Topicele Kafka sunt categoriile folosite pentru organizarea mesajelor. Fiecare topic are un nume unic în întregul cluster. Acestea sunt mulți-abonați, ceea ce înseamnă că un subiect poate avea zero, unul sau mai mulți consumatori ce preiau datele scrise în acesta.

Pentru a implementa un cluster Kafka scalabil, cu mai mulți brokeri ce lucrează simultan pe aceleași topice, este necesara și o instanța de Zookeeper pentru a sincroniza brokerii.

Apache ZooKeeper este un serviciu ce este specializat pe configurarea și sincronizarea sistemele distribuite. In arhitectura propusa Zookeeper ține evidența stărilor nodurilor, topicelor și a partițiilor clusterului Kafka. Acesta se asigura că mai mulți consumatori sau producători pot lucra simultan cu aceleași date din topicele clusterului Kafka fără a modifica ordinea in care sunt stocate acestea.

3.3 Docker Engine

Containerizarea este una dintre cele mai eficiente metode de virtualizare disponibile. Containerele elimină nevoia de sisteme de operare virtualizate sau hipervizoare. Spre deosebire de mașinile virtuale, care se bazează pe nucleul lor virtual, containerele folosesc nucleul sistemului de operare gazdă. Acest lucru reduce drastic utilizarea resurselor.

Docker Engine este continer runtime-ul implicit al Kubernetes. Acesta creează un proces demon pe partea de server care găzduiește imagini, containere, rețele și volume de stocare. De asemenea, oferă o interfață de linie de comandă (CLI) la nivelul clientului, care permite utilizatorilor să interacționeze cu demonul prin intermediul API-ului Docker Engine.

O imagine de container este un pachet software, independent și executabil, care include tot ceea ce este necesar pentru a rula o aplicație: cod, sistemului de operare și dependențele necesare pentru a rula codul**.**

Docker registries reprezintă locul in care sunt stocate imaginile de container. In mod implicit, Docker Engine caută imagini in registrul public Docker Hub însă poate fi configurat sa lucreze cu registre private.

Containerele dispun de propriul sistem de fișiere, memorie, spațiu de stocare și pot comunica între ele prin canale bine definite. Acestea sunt create dintr-o imagine de continer de către Docker Engine

3.4 Kubernetes

Kubernetes este un sistem open-source pentru automatizarea, scalarării și gestionării aplicațiilor containerizate. Acesta distribuie aplicațiile într-un cluster și asigura nevoile de rețea ale containerelor in mod dinamic.

Un cluster Kubernetes este format din noduri de lucru (Worker node) ce rulează aplicații containerizate și un nod de management (Master node). Un nod poate fi reprezentat de o mașina fizica sau una virtuala și poate deține rolurile de Worker si Master simultan.

Pentru a permite rularea continerelor fiecare nod din clasterul Kubernetes trebuie sa aibă instalat un **continer runtime (Docker Engine)**. Kubernetes are la baza următoarele componente:

- **Pods** — sunt obiecte de bază ale Kubernetes responsabile de încapsularea containerelor, resurselor de stocare și IP-urilor de rețea.
- **Delpoyments** — sunt obiecte Kubernetes care gestionează podurile. Acestea permit precizarea numărului de pod-uri pe care dorim să le pornim cât și argumentele acestora.
- **Services** — reprezintă o modalitate abstractă de a expune o aplicație care rulează pe un set de Pods ca serviciu de rețea.
- **Persistent Volume (PV)** — este o resursa de stocare dintr-un cluster ce poate fi furnizată manual de către un administrator sau dinamic folosind Storage Class. PV-urile au un ciclu de viață independent fată de podul de care este utilizat, permițând astfel ca datele din acesta să rămână disponibile și după ștergerea sau repornirea podului.
- **Persistent Volume Claim (PVC)** — este o solicitare de spațiu de stocare a unui utilizator. Pod-urile pot solicita niveluri specifice de resurse (CPU, memorie sau spațiu de stocare) iar un PV se poate instanța automat pentru a satisface necesitățile sistemului.

MicroK8s este un sistem open-source pentru automatizarea implementării, scalarării și gestionării aplicațiilor containerizate. Oferă principale funcționalități oferite de Kubernetes cu un consum mai mic de resurse. Datorita resurselor limitate pe care le avem la dispoziție am ales sa folosesc Microk8s pentru arhitectura de servicii descrisă.

4 INPLEMENTARE

Soluția propusă este reprezentată de o arhitectură de servicii pentru procesarea de știri în limba engleză publicate în internet, preluate prin intermediul unor fluxuri RSS și clasificate prin intermediul unor rețele neuronale. Aceasta este susținuta de către orchestratorul de containere Kubernetes folosind Docker Engine in rolul de contiher runtime iar componentele necesare sunt expuse în rețeaua locală prin intermediul serviciilor.

4.1 Arhitectura generala a sistemului

Pentru a putea asigura colectarea și procesarea unui volum mare de date, în cadrul dezvoltării arhitecturi am integrat o serie de soluții scalabile. Transmitea datelor se face prin intermediul topicelor Kafka cu ajutorul coordonatorului Apache ZooKeeper, acest lucru permite distribuirea necesităților computaționale pe mai multe noduri de lucru. Componenta de stocare a informațiilor este reprezentată de ElasticSearch alături de interfața de management oferită de către Kibana, o soluție standard în industrie pentru gestionarea de volume mari de date.

După cum este prezentat in figura 4.1, arhitectura este formată din trei subsisteme cu rolul de a colecta, clasifica și vizualiza datele, alături de serviciile aferente necesare pentru stocarea și transmiterea datelor.

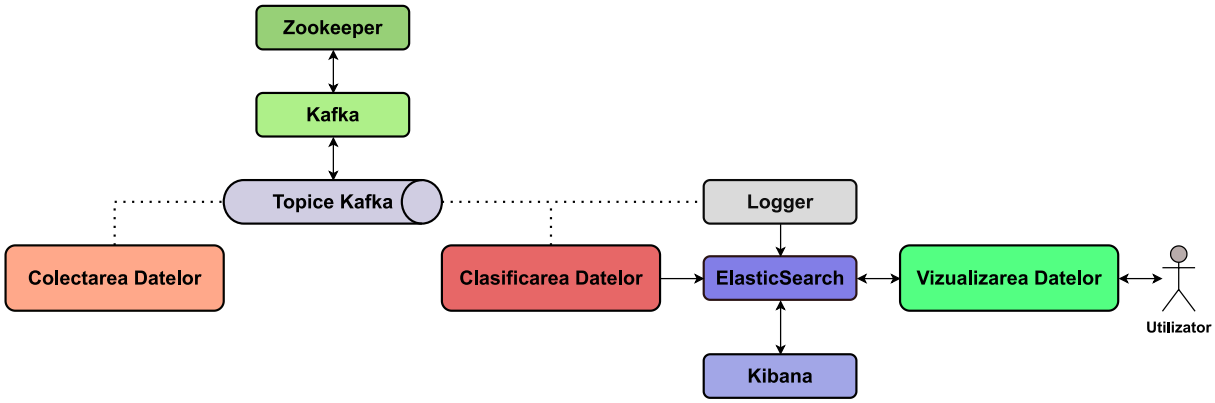

Figura 4.1 Arhitectura generala a sistemului

Fiecare dintre cele 3 subsisteme rezolva una dintre problemele legate de influenta mass-media asupra opiniei publice. Subsistemul de colectarea a datelor preia știrile dintr-o multitudine de surse definite de administrator. Subsistemul de clasificare se folosește de doua rețele neuronale pentru a eticheta informațiile, eliminând astfel nevoia de evaluatori umani ce pot fi influențați. Subsistemul de vizualizarea permite utilizatorului să înțeleagă și sa exploreze știrile și informațiile adunate.

4.1.1 Subsistemul de colectare a datelor

Subsistemul de colectare a datelor are rolul de a extrage informațiile provenite din fluxurile RSS ale publicațiilor disponibile in mediul online. Fiecare flux RSS definit este preluat prin intermediul unei aplicații containerizate pentru a permite administratorului sistemului să adauge sau să elimine surse de informații dinamic, fără a opri funcționarea întregului subsistem.

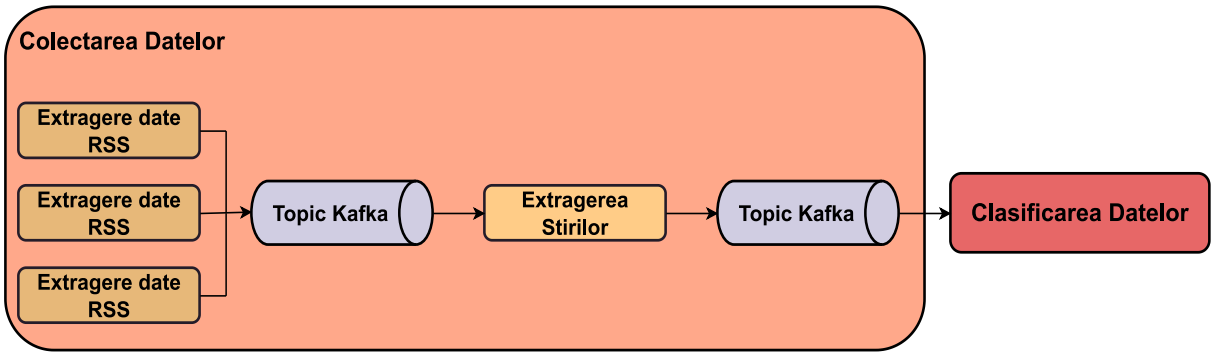

Figura 4.2 Subsistemul de colectare a datelor

Un flux RSS (Really Simple Syndication) încapsulează titlul, rezumatul, linkul, data și ora publicării știrilor in format XML. Acest conținut este distribuit în timp real, astfel încât primul rezultate din fluxul RSS să fie întotdeauna cel mai recent conținut publicat. In cadrul sistemului am ales preluarea datelor din trei fluxuri RSS aparținând unor publicații de renume pe plan mondial:

- **BBC - http://feeds.bbci.co.uk/news/rss.xml**
- **CNN - http://rss.cnn.com/rss/edition.rss**
- **FOX - http://feeds.foxnews.com/foxnews/latest**

Subsistemul de colectare a datelor este format, după cum este prezentat și in figura 4.2, dintr-un set de pod-uri cu rolul de a extrage datele din fluxurile RSS și un pod pentru a prelua conținutul text al știrilor. Acestea comunică prin intermediul unui topic Kafka denumit „rss_data”. Serviciul de extragere a textului transmite datele colectate către subsistemul de clasificare prin intermediul topicului „crawled_news”.

4.1.2 Subsistemul de clasificare a datelor

Subsistemul de clasificarea a datelor preia documentele JSON de la subsistemul de colectare a datelor prin intermediul unui topic Kafka. Acesta are rolul de a clasifica conținutul text al știrilor și de a introduce rezultatul obținut in ElasticSearch.

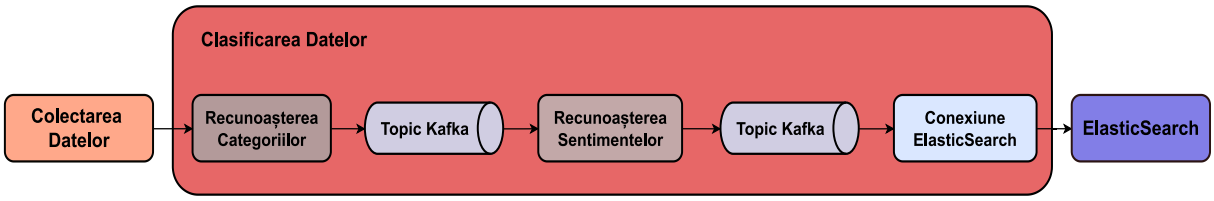

Figura 4.3 Subsistemul de clasificare a datelor

După cum este prezentat si in figura 4.3, acest subsistem este format din trei componente distincte, două aplicații containerizate bazate pe rețele neuronale dezvoltate si un serviciu numit „Conexiune ElasitcSearch” cu rolul de a introduce știrile clasificate in baza de date ElasticSearch. Aceste componente sunt interconectate prin intermediul topicelor Kafka.

Cele două aplicații implementate sunt bazate pe rețele neuronale, astfel:

- **Recunoașterea Categoriilor** – atribuie textului intre 0 și 4 categorii in funcție de subiectul știrii.
- **Recunoașterea Sentimentelor** – atribuie textului o valoare intre 0 și 1 data de pozitivitatea conținutului.

Seturile de date utilizate pentru antrenarea si testarea rețelelor neuronale sunt in limba engleza. Am decis acest lucru, in defavoare dezvoltării pentru limba română, datorita lipsei de seturi de date etichetate corespunzător și a modelelor preantrenate pentru procesele de embedding, tokenizare, trunchiere și lematizare pentru limba română.

Pe parcursul dezvoltării acestui subsistem au fost testate mai multe categorii de rețele neuronale si soluții algoritmice pentru ambele aplicații containerizate. Pentru a putea evalua performantele fiecărui model trebuie sa definim un set de metrici de evaluare specifice pentru fiecare problema in parte in funcție de sterile de date utilizate si de tipul de clasificare.

Metrici de evaluare ale modelelor

O **matrice de confuzie** este o reprezentare matricială a rezultatelor predicției care este adesea folosită pentru a descrie performanța modelului de clasificare pe un set de date de testare pentru care sunt cunoscute etichetele corecte. Fiecare predicție se poate încadra in una din patru categorii, în funcție de modul în care se potrivește cu valoarea adevărata:

- **True positives (tp)** – reprezintă numărul de intrări ce **sunt** desemnate ca aparținând clasei A iar modelul le încadrează in clasa A.
- **False positives (fp)** – reprezintă numărul de intrări ce **nu sunt** desemnate ca aparținând clasei A iar modelul le încadrează in clasa A.
- **True negatives (tn)** – reprezintă numărul de intrări ce **nu sunt** desemnate ca aparținând clasei A iar modelul **nu** le încadrează in clasa A.
- **False negatives (fp)** – reprezintă numărul de intrări ce **sunt** desemnate ca aparținând clasei A iar modelul **nu** le încadrează in clasa A.

Pe baza datelor obținute din matricea de confuzie se pot calcula o serie de valori mai mult sau mai puțin relevante pentru performantele modelelor testate.

Acuratețea (4-1) reprezintă numărul de intrări clasificate in mod corect de către model raportat la numărul de date introduse spre clasificare, această metrică de evaluare este folositoare dacă exemplele sunt distribuite uniform in setul de date, iar etichetele au o importanță similară. Problema de clasificare binară **Recunoașterea Sentimentelor** se încadrează in criteriile necesare pentru a asigura relevanta acestei metrici pentru evaluarea performantelor modelului dezvoltat.

$$Accuracy = \frac{tp + tn}{tp + tn + fp + fn} \quad \textbf{(4-1)}$$

Precision calculează de cate ori modelul este corect când atribuire o clasa prin formula (4-2). Aceasta metrica este utilă în dezvoltate modelelor pe baza unor seturi de date dezechilibrate, în acest caz acuratețea devine o metrica inadecvată.

$$Precision = \frac{tp}{tp + fp} \quad \textbf{(4-2)}$$

Recall calculează de cate ori o clasa este atribuita corect, fața de cate ori acesta era prezenta in setul de date pentru testare prin formula (4-3). Precum percision, aceasta metrica este utila in dezvoltate modelelor pe baza unor seturi de date dezechilibrate

$$Recall = \frac{tp}{tp + fn} \quad \textbf{(4-3)}$$

Scorul F1 (4-4) reprezintă media armonică dintre precison si recall. Când scorul F1 atinge valoare de 1, modelul a atins performanța maximă posibilă pe clasa analizată, iar valoarea de 0 reprezintă cazul cel mai nefavorabil. Această metrica este relevantă in cadrul unui sistem de clasificare pe etichete multiple distribuite inegal in setul de date, precum **Recunoașterea Categoriilor**.

$$F1 = \frac{tp}{tp + \frac{1}{2}(fp + fn)} \quad \textbf{(4-4)}$$

Funcția de cost are rolul de a reduce cu toate aspectele modelului într-un singur număr (numit **Cost**), astfel încât îmbunătățirile acestui număr să fie semnul unui model mai bun. In cadrul dezvoltării modelelor au fost folosite doua funcții de cost :

- **Categorical crossentropy** este o funcție de cost care este utilizată în problemele de clasificare cu mai multe clase. Acestea sunt reprezentate de cazuri în care un exemplu poate aparține doar uneia dintre multe categorii posibile, precum **Recunoașterea Categoriilor.** Categorical crossentropy este definita de ecuația (4-5) unde N reprezintă numărul de valori din vectorul de predicție, $\hat{y}_i$ valoare cu numărul i din vector și y_i valoare cu numărul i din vectorul ținta.

$$\text{Cost} = -\sum_{i=1}^{N} y_i \, (\log \hat{y}_i) \quad \textbf{(4-5)}$$

- **Binary crossentropy** este un caz special de categorical crossentropy in care modelul prezice o valoare binară și nu un vector de probabilități. Aceasta funcție de cost de poate implementa pe problema **Recunoașterea Sentimentelor** deoarece modelul trebuie să prezică daca o știre face partea din clasa pozitiv sau negativ. Binary crossentropy este definita de ecuația (4-6) iar parametrii au aceiași însemnătate ca și la categorical crossentropy.

$$\text{Cost} = -\frac{1}{n}\sum_{i=1}^{N} y_i \log \hat{y}_i \quad + \quad (1 - y_i) \log(1 - \hat{y}_i) \quad \textbf{(4-6)}$$

4.1.3 Subsistemul de vizualizare a datelor

După cum este prezentat în figura 4.4, subsistemul de vizualizare a datelor are rolul de a colecta si analiza datele din ElasticSearch și a le expune intru-un mod intuitiv si plăcut utilizatorului.

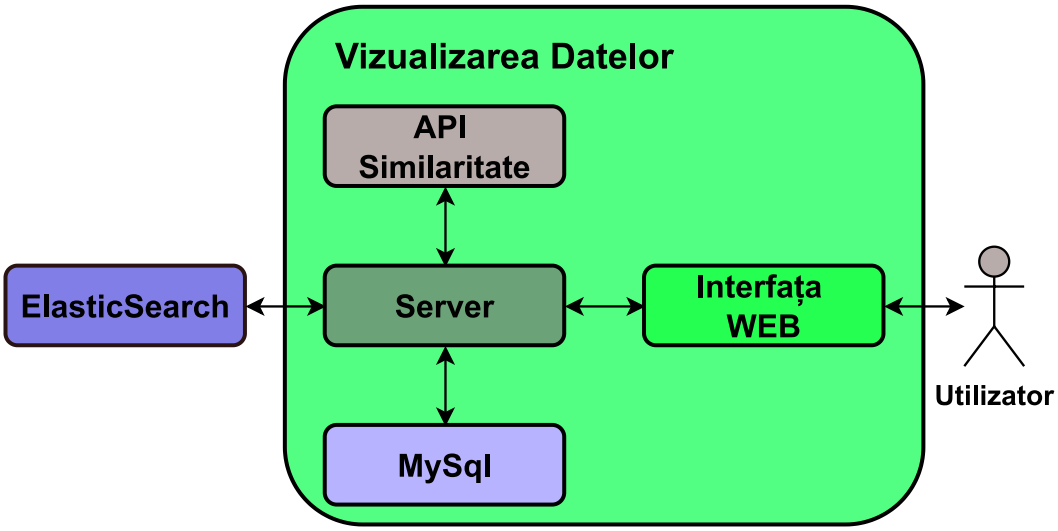

Figura 4.4 Subsistemul de vizualizare a datelor

Acesta este format din 4 componente :

- **Similaritate REST API –** calculează și întoarce cele mai similare știri pe baza unui ID
- **Baza de date MySql –** stochează date personale ale unui utilizator
- **Server –** asigura comunicația dintre restul elementelor precum si un mecanism de sesiuni
- **Interfața web –** interfața grafica prin intermediul căreia utilizatorul interacționează cu sistemul

Interfața grafică a fost dezvoltată utilizând biblioteca React pentru limbajul de programare JavaScript. Aceasta a fost concepută astfel încât să permită atât utilizatorilor autentificați cât si celor neautentificați să acceseze toate funcționalitățile oferite de arhitectura prezentata.

Server-ul a fost construit utilizând NodeJS pentru a putea beneficia de mecanismele de asincronitate . Scopul principal al acestuia este să răspundă la cererile trimise de către utilizator prin interfața grafică, interogând componentele de stocare și analiză disponibile

4.2 Serviciul de extragere a datelor din fluxurile RSS

Serviciul de extragere a datelor din fluxurile RSS este implementat cu ajutorul API-ului **PhantomJS** [8]. Acest API este folosit pentru a deschide pagini web, a face capturi de ecran și pentru a rula cod JavaScript în contextul paginii. Datele ce pot fi extrase din fluxurile RSS sunt reprezentate de titlul și descrierea știrii, data si ora publicării precum și adresa către pagina web ce conține textul complet al știrii.

Pentru a putea generaliza o aplicație containeriză astfel încât sa preia date din orice flux RSS dorit de utilizator, executabilul rezultat in urma compilării programului preia doua variabile de mediu ce trebuie setate :

- **RSSLink** – adresa către fluxul RSS dorit
- **RSSTag** – un string cu rolul de identificator pentru știrile extrase

Programul se lansează in execuție o dată la 10 minute, pentru a evita o utilizare ineficientă a resurselor computaționale. Aceasta limitare a fost impusa deoarece un flux RSS nu este actualizat constant ci la intervale predefinite de către administrator. De exemplu, fluxul ales anterior ce aparține BBC se actualizează o data la 15 minute.

Executabilul preia codul sursă al paginii web aflate la adresa indicată de **RSSLink** si reface fișierul XML ce conține informațiile despre știri. Programul iterează prin stirpe conținute in fișier si verifică dacă acestea sunt deja introduse in componenta de stocare, prin interogarea ElasticSearch. . In cazul in care au fost detectate date adiționale, programul le preia și le trimite topicul-ui Kafka responsabil alături de câmpul **RSSTag**, in format JSON.

Deoarece acest serviciu reprezintă punctul de intrare a datelor in pipeline-ul nostru de servicii, el nu are rolul de consumator pentru nici unul dintre topicele Kafka însă are rolul de producător pentru topicul „rss_data”, figura 4.5.


```
{
    "link": "https://www.bbc.co.uk/sport/football/59033038 ",
    "description": "Wolves' impressive Premier League form continues as they inflict a
    defeat on Everton which will surely worry beaten manager Rafael Benitez.",
    "source": "BBC News - Home",
    "title" : "Wolves' fine run continues as sloppy Everton punished again",
    "pubDate": "20211101",
    "pubTime": "22:32:52",
    "RSSTag": "BBC",
}
```


Figura 4.5 Document Json trimis către topicul "rss_data"

Imaginea de container utilizata in cadrul arhitecturii a fost creată dintr-o imagine de Ubuntu in interiorul căreia a fost copiat fișierul jar rezultat din compilarea programului precum și librăria PantomJS. Pentru a porni un serviciu de extragere a datelor din fluxurile RSS este necesară declararea unui deployment care să utilizeze imaginea generată anterior și să seteze cele doua variabile de mediu „RSSLink" și „RSSTag" prin intermediul unui fișier „yaml".

In cadrul arhitecturii se pot pornii mai multe astfel de deployment-uri cu un număr personalizat de replici pentru a facilita preluarea de știri din mai multe fluxuri RSS. Acest mod de implementare a fost ales pentru a permite utilizatorului să pornească sau să oprească extragerea de date de la fluxuri RSS individuale.

4.3 Serviciul de extragere a știrilor

Programul are ca scop preluarea conținutului text al știrilor din paginile web are publicațiilor fără a include reclamele sau alte elemente irelevante pentru utilizatorul soluției dezvoltate. Serviciul își începe execuția când recepționează un mesaj prin intermediul topicul Kafka populat de către serviciul de extragere a datelor din fluxurile RSS. Acesta verifica daca la adresa web specificata in documentul JSON primit se afla o pagina disponibila ce conține știrea in format text, iar in caz afirmativ demarează procesul de extragere a informației.

Aplicația containerizată accesează pagina web, extrage codul HTML, elimină tag-urile și identifică începutul și finalul știrii propriu-zise, iar apoi trece fiecare paragraf printr-un proces de filtrare a reclamelor. După extragerea textului dorit, acesta este adăugat la documentul primit inițial și trimis către topicul Kafka „crawled_news".

Imaginea de container utilizată in cadrul arhitecturii a fost creată dintr-o imagine de Ubuntu in care a fost copiat scriptul de Python . Prin intermediul repository-ului standard și al utilitarului **apt-get** au fost instalate interpretorul de Python3, programul de instalare a pachetelor **pip** și dependințele.

Pentru a porni un serviciu de extragere a știrilor este necesara declararea unui deployment care sa utilizează imaginea generata anterior prin intermediul unui fișier „yaml". Acesta poate fi setat sa pornească un număr personalizat de replici sporind astfel redundanța și scalabilitatea sistemului.

4.4 Rețeaua neuronală pentru recunoașterea categoriilor

Aceasta rețea neuronală reprezintă un model de inteligență artificială dezvoltat cu rolul de a atribuii știrilor intre 0 și maxim 4 categorii, in funcție de topicul textului, dintr-un total de 15 categorii posibile. Datorită concluziilor si recomandărilor expuse in articolul Classification of News Dataset [2], am ales implementarea si testarea unei rețele neuronale bazata pe multiple straturi LSTM.

Pentru antrenarea, validarea și testarea modelului se va utiliza o versiune mai nouă a setului de date prezentat in articolul [2], format din știri obținute de la publicația HuffPost. Acesta este împărțit din 41 de etichete cu număr diferit de exemple, însă numărul de clase distincte a fost redus la 15 printr-un proces de echilibrare si curățare a datelor.

Valorile de intrare ale rețelei neuronale se obțin prin concatenarea titlului și a descrierii unei știri și aplicarea ulterioara a pașilor de preprocesare și tokenizare. Aceste date sunt preluate din topicul Kafka „crawled_news”.

Valorile de ieșire sunt reprezentate de un vector de 15 valori cuprinse in intervalul [0,1], reprezentative pentru scorul de apartenență la fiecare dintre categorii. Toate categoriile ce corespund unei valori mai mari de 0.25 sunt considerate ca fiind atribuite știrii asupra căreia a fost făcută predicția. A fost ales pragul de 0.25 pentru a permite unei știți sa se încadreze in maxim 4 categorii simultan (4 categorii cu un procent de apartenenta la fiecare de 25%). Valorile obținute in urma predicției sunt atașate documentului JSON corespunzător știrii și trimise către topicul „categorised_news”.

Deoarece știrile extrase de subsistemul de colectare a datelor au dimensiuni mult mai mari decât exemplele utilizate in dezvoltarea modelului, acestea vor fi împărțite în propoziții și paragrafe și introduse în rețeaua neuronala individual. Rezultatele obținute astfel sunt agregate prin intermediul unei medii ponderate astfel încât să devină relevante pentru întreaga știre.

Modelul ce va fi integrat in serviciul de analiza sentimentala a știrilor va fi determinat pe baza metricilor de evaluare specifice definite anterior. Deoarece avem o problema de clasificare multipla pe un set de date cu etichetele distribuite inegal, valoare cea mai relevanta este scorul F1.

4.4.1 Data analysis

Setul de date utilizat reprezintă o versiune mai nouă a setului de date prezentat in articolul [2] și este format din știri obținute de la HuffPost. Prin utilizarea acestuia s-a urmărit obținerea unui model cu performante mai ridicare ca rețelele neuronale testate in articolul [2]. Fiecare știre prezinta următoarele câmpuri:

- **category** - categoria din care face parte articolul
- **headline** - titlul știri
- **authors** - autorul articolului
- **link** - adresa web unde este găsita știrea respectiva
- **short_description** - o scurta descriere a știrii
- **date** – data publicării articolului

In cadrul dezvoltării sistemului câmpurile **headline și short_description** vor fi folosite ca intrări in procesele de antrenare, validare și testare, iar câmpul **category** va fi utilizat drept etichetă.

Setul de date conține 200.853 de știri din 2012 până în 2018. Titlurile știrilor au o lungime medie de 9.53 de cuvinte pe titlu și o lungime maxima de 44 cuvinte iar, descrierile au o lungime medie de 19.72 cuvinte pe descriere și o lungime maxima de 243 cuvinte.

Pentru a putea evalua corect performanța modelului**,** setul de date a fost împărțit în trei parți distincte, respectiv in seturile de antrenare, validare și testare, fiecare fiind utilizat într-o etapă diferită a procesului de dezvoltare. Seturile de validare și testare au câte 10% din volumul inițial de exemple iar setul de antrenare 80%.

Distribuția claselor in cele 3 subseturi urmează distribuția din setul de date inițial după procesul de curățare. Acest lucru este necesar pentru a asigura ca modelul dezvoltat cu ajutorul subseturilor se poate aplica cu rezultate similare pe setul de date complet.

Ca termen comparativ pentru modelul dezvoltat oferim un sistem de recunoaștere aleatoriu. Asupra setului de date de testare au fost utilizate două DummyClassifier oferite de către biblioteca „scikit-learn”:

Dummy Most Frequent alege clasa cea mai frecventă din setul de date ca fiind predicția pentru toate datele de intrare. In cazul prezentat, clasa „POLITICS” este cea mai frecventa.

Dummy Uniform alege etichetele in mod uniform. Fiecare știre are probabilități egale sa fie distribuita in orice clasa.

4.4.2 Curățarea setului de date

Câmpurile **headline** și **short_description** prezintă informații necesare in vederea problemei de clasificarea a topicului unei știri. Pentru a putea utiliza aceste date simultan**,** la setul de date a fost adăugat un câmp denumit X format din concatenarea celor 2 coloane. În setul de date inițial câmpul X are o lungime maxima de 245 de cuvinte și o lungime medie de 29.14 de cuvinte.

După cum este evidențiat si in figura 4.6, setul de date este inițial format din 41 de etichete cu un număr diferit de exemple. Pentru a simplifica problema și a sporii performantele modelului, numărul de categorii trebuie micșorat.

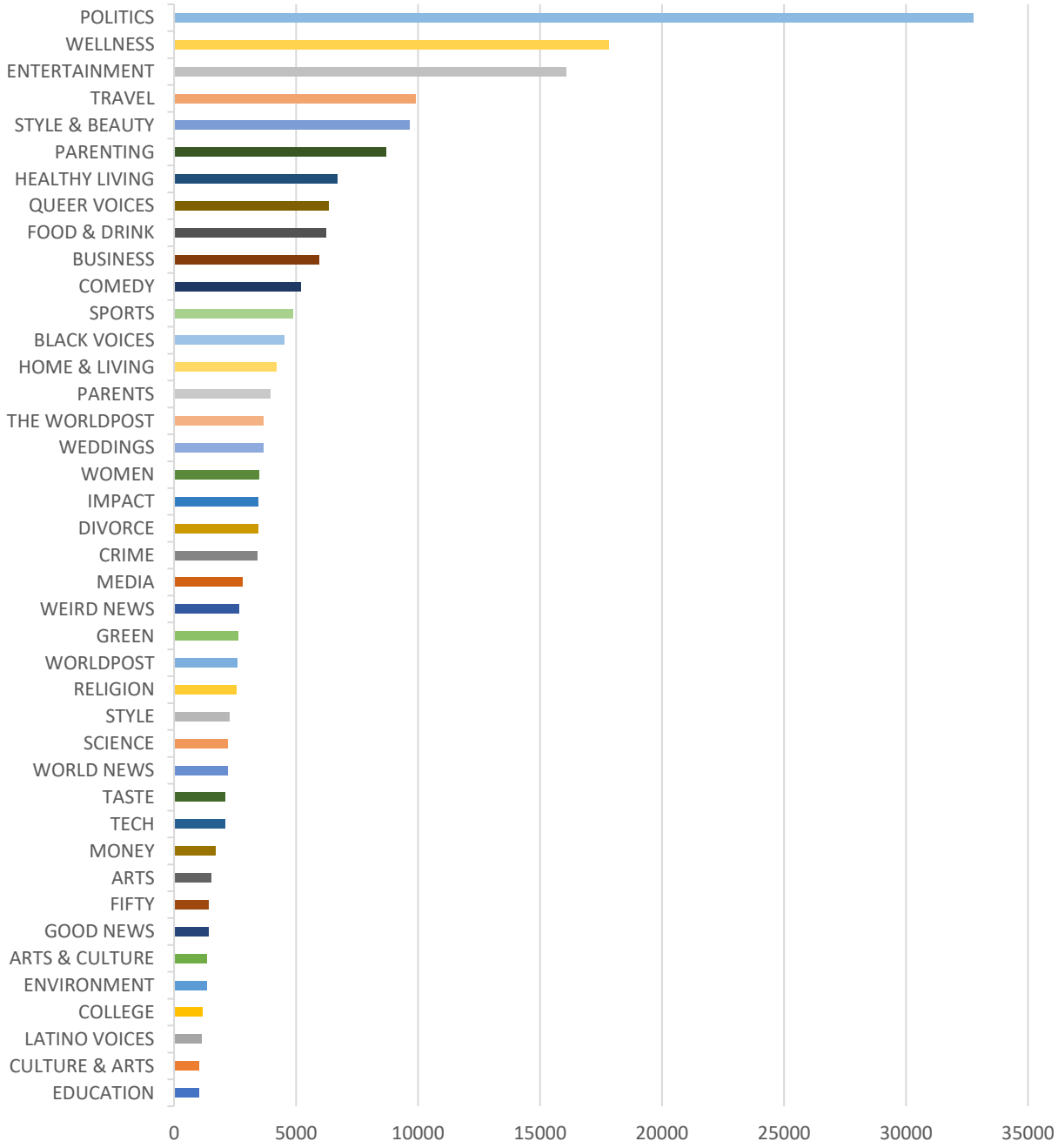

Figura 4.6 Distribuția claselor in setul de date HuffPost inițial

În interiorul setului de date există categorii distincte cu știri similare („ARTS”, „ARTS & CULTURE”, „CULTURE & ARTS”) sau categorii ce nu descriu conținutul textului („BLACK VOICES”, „ THE WORLDPOST”). Prin eliminarea sau comasarea acestora am ajuns la o variantă finală a setului de date cu 125.597 de știri distribuite in 15 clase, acesta reprezintă o scădere semnificativă de la numărul de 25 de etichete prezente în setul de date folosit in articolul [2]. Distribuția acestor clase este evidențiata în figura 4.7 .

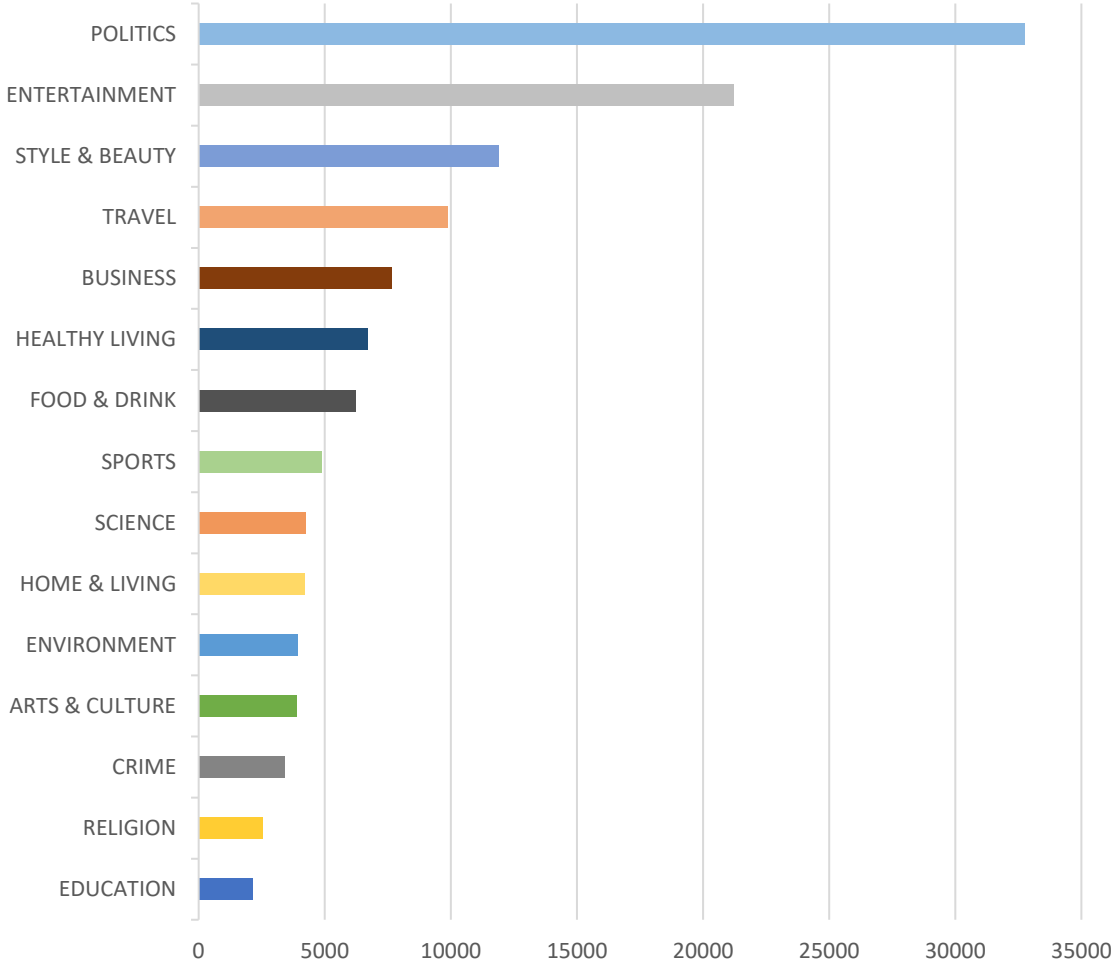

Figura 4.7 Distribuita claselor in setul de date HuffPost după curățare

În setul de date rezultat există inegalități între numărul de exemple prezente in fiecare clasa. Categoria „POLITICS” are 32.739 de știri iar „EDUCATION” 2.148, un factor de 15.24. Această diferența poate introduce prejudecăți(biases) în modelul dezvoltat, scăzându-i performanțele.

Dimensiunea medie a câmpului X a scăzut cu 1,8 cuvinte în urma procesului de curățare, ajungând la 27.36 cuvinte pe știre. Lungimea maxima a rămas la valoarea de 245 de cuvinte.

4.4.3 Preprocesarea datelor

Preprocesarea reprezintă totalitatea metodelor prin care textul este curățat și pregătit pentru a fi introdus in model. Acest pas este efectuat atât înainte de procesele de antrenare și evaluarea a modelului cat și înainte de clasificarea unui text si urmează structura evidențiata in figura 4.8 .

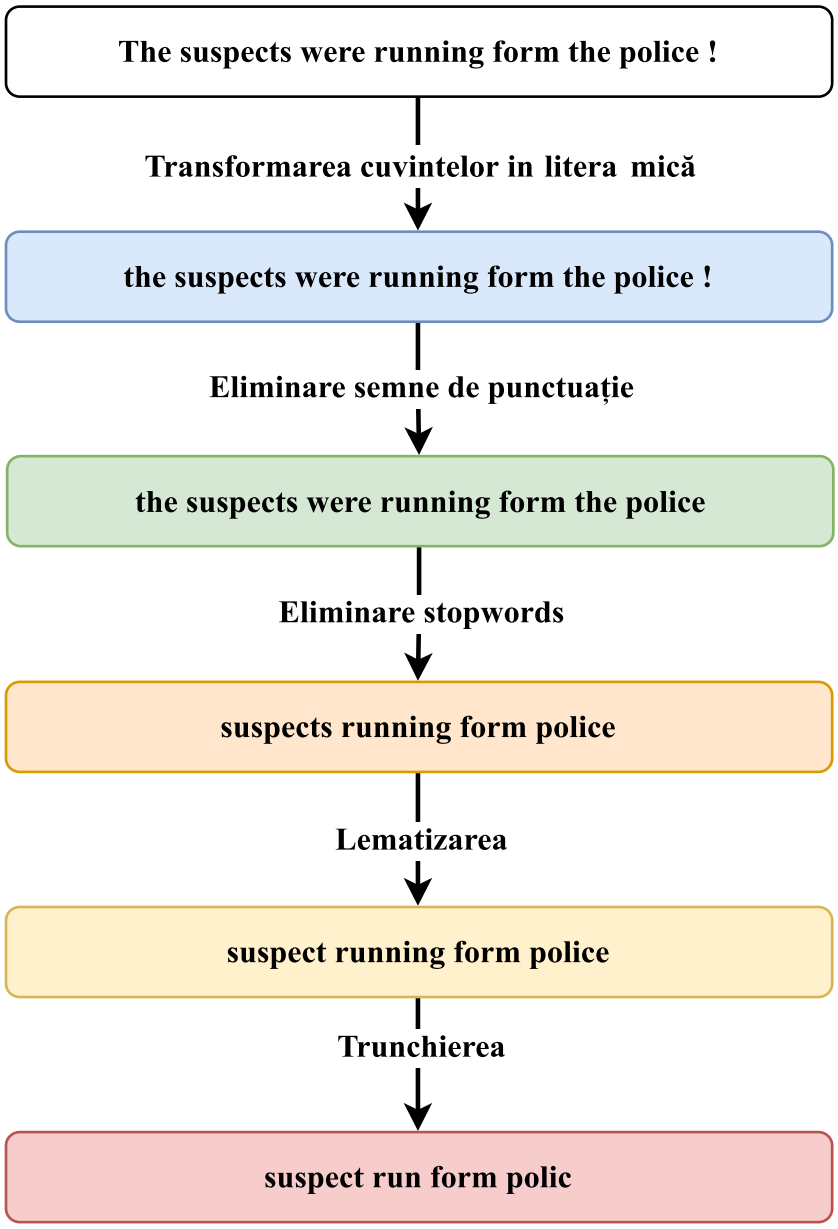

Figura 4.8 Pașii urmați in procesul de preprocesare a setului de date HuffPost

Prin intermediul preprocesării se urmărește păstrarea unui nivel de informații cât mai mare in textul dat, obținând însă o dimensiune cât mai mică a datelor pentru a optimiza procesele de antrenare, evaluare si clasificare.

Transformarea cuvintelor in litera mica

Transformarea in litera mica reprezintă procesul de a transforma caracterele scrise cu litera mare in caractere scrise cu litera mica. Acest proces ignora semnele de punctuație sau caracterele speciale. Scopul acestui pas este de a da o oarecare uniformitate datelor, necesară pentru ulterioarele metode de preprocesarea aplicate.

Eliminarea semnelor de punctuație

In contextul dat vom defini un semn de punctuație, conform cu modulul standard „string” aferent limbajului de programare Python 3.0, ca orice caracter prezent in următorul sir : **!"#$%&'()*+,-./:;<=>?@[\]^_`{|}~** .

Eliminarea acestor caractere reduce dimensiunea textului fără a afecta informațiile necesare pentru deducerea categoriilor din care face parte știrea analizata.

Eliminarea cuvintelor stopwords

“Stopwords” sunt cuvintele care nu oferă nicio informație utilă pentru a determina în ce categorie ar trebui să se încadreze textul. Aceasta caracteristică se poate datora faptului că nu au sens (prepoziții, conjuncții, etc). In contextul dat vom defini un stopword ca orice cuvânt prezent în șirul din anexa 1.

Prin extragerea acestor cuvinte, eliminăm informațiile de nivel scăzut din text pentru a ne concentra mai mult pe informațiile importante. Cu alte cuvinte, putem spune că ștergerea acestor cuvinte nu va avea niciun impact negativ asupra modelului pe care îl antrenăm.

Lematizarea

Lematizarea reprezintă procesul de transformare a formelor flexionare ale unui cuvânt in forma acestuia din dicționar (numita lemă), astfel încât acestea să poată fi analizate ca o singură entitate. Pentru procesul de lematizare am ales folosirea implementării WordNetLemmatizerdin din modulul Natural Language Toolkit (NLTK).

Trunchierea

Trunchierea cuvintelor reprezintă procesul de eliminare a sufixelor și prefixelor cuvintelor cu scopul de a le aduce la forma lor de bază. Pentru preprocesarea seturilor de date am ales folosirea implementării algoritmului de trunchiere Porter din modulul Natural Language Toolkit (NLTK).

Algoritmul lui Porter a fost dezvoltat și prezentat de Martin Porter în articolul [7]. Avantajele acestuia față de alți algoritmi, cum ar fi algoritmul Lancaster, sunt timpul redus de rulare și simplitatea.

Tokenizarea datelor

Procesul de tokenizare a datelor constă in transformarea cuvintelor din cadrul știrilor in valori numerice pentru a putea fi procesate de modelul dezvoltat. În cadrul rețelei neuronale pentru recunoașterea categoriilor, tokenizarea se face cu ajutorul bibliotecii „Keras” prin modulul antrenat pe setul de date avut la dispoziție.

In urma procesului de tokenizare se efectuează un proces de extindere a datelor cu valori de 0 astfel încât fiecare știre sa se reprezinte printr-un vector de 128 de valori.

Embedding

Procesul de embedding este reprezentat de transformarea valorilor din cadrul vectorului reprezentativ pentru știre in vectori de numere reale. Pentru acest proces a fost folosita implementarea Word2Vec din biblioteca Gensim. Fiecare valoare a fost extinsa la un vector de 350 de numere reale.

Valorile obținute in urma procesului de embedding vor fi utilizate în generarea stratului Embedding din cadrul modelului bidirecțional LSTM. Acesta are rolul de a transforma vectorul de 128 de întregi obținut in procesul de tokenizare într-o matrice de numere reale cu dimensiunile 128 si 350.

4.4.4 Testarea algoritmilor de inteligenta artificiala

Decision Tree Classifier generează un arbore de decizie bazat pe caracteristicile extrase din setul de date de antrenare. Fiecare caracteristică extrasă generează întrebări „da/nu” ce izolează proprietățile fiecărei clase, aceste întrebări sunt reprezentate ca noduri în structura arborescenta generata prin acest proces. In urma parcurgerii arborelui fiecărei știri din setul de testare îi este atribuita o etichetă.

Extremely Randomized Trees Classifier utilizează un număr de arbori de decizie antrenați pe bucăți aleatorii din setul de date. Fiecare arbore are dimensiunea maxima de 1 nivel. Principiul de funcționare al acestui clasificator este reprezentat de faptul ca o serie de clasificatori cu performante scăzute, generează împreuna o acuratețe mai ridicată. Predicția clasificatorului este data de eticheta care a fost desemnată de cei mai mulți arbori de decizie.

Random Forest Classifier generează un set de arbori de decizie. Fiecare arbore este antrenat pe o parte aleatorie din setul de date, astfel ne asiguram că fiecare copac își bazează predicțiile pe date diferite. Procesul de predicție este similar cu Extremely Randomized Trees Classifier.

Extreme Gradient Boosting (XGB) este o bibliotecă open-source care oferă o implementare eficientă a algoritmului gradient boosting. Acest algoritm este bazat pe un proces iterativ de învățare prin care modelul ține cont de predicțiile făcute corect la iterația anterioară cu ajutorul unor mecanisme numite ansambluri. Ansamblurile sunt construite din arbori de decizie ce sunt adăugai pe rând pentru a corecta erorile făcute de generațiile anterioare ale modelului. Acesta tip de model de învățare automată bazat pe ansambluri este denumit boosting.

Extreme Gradient Boosting Random Forest Classifier (XGBRF) aplică tehnicile iterative ale XGB asupra arborilor de decizie din Random Forest Classifier. In cadrul XGBRF, asamblările sunt formate din clasificatoare de timpul random forest menite sa compenseze defectele din iterațiile anterioare ale modelului.

Linear Support Vector Classifier (LinearSVC) reprezintă un algoritm matematic de clasificare supervizat bazat pe o funcție liniară. SVM mapează exemplele din setul de date de antrenament în puncte din spațiu, astfel încât să maximizeze distanța dintre grupurile de datele etichetate diferit. Predicțiile noi sunt mapate în același spațiu, iar categoria atribuită este bazată pe grupul de etichete care se afla la cea mai mica distantă.

Toate modelele exceptând **LinearSVC** au prezentat rezultate mai bune decât clasificatoarele Dummy. Cel mai performant a fost **XGB,** iar scorurile acestuia ar trebui depășite de către modelul dezvoltat de noi pentru a justifica folosirea lui. Rezultatele obținute de aceste clasificatoare sunt expuse in tabelul 4.1 .

Tabel 4.1 Rezultate clasificatoare algoritmice Classification Neural Network

Nume model	Acuratețe	Medie scoruri F1
Decision Tree	28%	17%
Random Forest	35%	15%
Extremely Randomized Trees	17%	10%
XGBRF	30%	8%
XGB	**37%**	**18%**
LinearSVC	9%	4%
KNeighbors	21%	9%
Dummy Most Frequent	25%	3%
Dummy Uniform	6%	6%

4.4.5 Dezvoltarea modelului bidirecțional LSTM

Modelul este format din 6 straturi, distribuite in modul expus de tabelul 4.2. Acestea înglobează un total de 20.019.859 de parametri, dintre care 902.159 sunt antrenabili.

Tabel 4.2 Straturile modelului bidirecțional LSTM

Nume strat	Tip strat	Dimensiune ieșire	Număr de parametrii
Input	InputLayer	[(None, 128)]	0
Embeddings	Embedding	(None, 128, 350)	19.685.050
LSTM-1	Bidirectional LSTM	(None, 128, 256)	490.496
LSTM-2	Bidirectional LSTM	(None, 256)	394.240
Dense-1	Dense	(None, 64)	16.448
Output	Dense	(None, 15)	975

Primul strat este de tipul InputLayer, fără parametri, și are funcția de a introduce date în model. Mărimea de ieșire a stratului reprezintă forma datelor de intrare, un vector de 128 de valori (date obținute din procesul de tokenizare). Am ales folosirea unei intrai de 128 de valori deoarece aceasta este dimensiunea maxima identificată a exemplelor după pasul de preprocesare.

Al doilea strat este format cu ajutorul valorilor obținute în pasul de embedding. Acesta prezinta 19.685.050 de parametrii neantrenabili. Dimensiunea de ieșire reprezintă transformarea unui vector de 128 de întregi într-o matrice de dimensiuni 128 și 350.

Următoarele 2 straturi sunt de tipul Bidirectional LSTM. Acestea lucrează împreună și își modifică greutățile in funcție de datele ce au fost, sunt sau vor fi introduse spre procesare. Acestea prezintă 490.496 si respectiv 394.240 de parametrii antrenabili cu o rata de dropout de 20%.

Penultimul strat este de tipul Dense. Fiecare neuron al acestuia este conectat la fiecare neuron din LSTM-2. Acest strat reduce dimensiunea datelor de la un vector de 256 de valori la 64 prin intermediul a 16.448 de parametrii antrenabili. Funcția de activare folosită a fost „relu”.

Ultimul strat este tot unul de tipul Dense cu un număr de 15 neuroni. Acesta are ca ieșire vectorul de 15 valori ce reprezintă predicția modelului. Funcția de activare folosită a fost „softmax”.

Modelul a fost compilat utilizând funcția de cost „Sparse Categorical Crossentropy” si optimizatorul „Adam” cu valoarea 0.001.

4.4.6 Antrenarea modelului

Modelul a fost antrenat pe setul de antrenament si validat la finalul fiecărei epoci pe setul de validare, acestea reprezentând 80% si respectiv 10% din setul de date complet. Au fost folosite argumentele batch_zise=256 si shuffle=True. Procesul s-a întins pe 15 epoci si este evidențiat in figurile 4.9 si 4.10 .

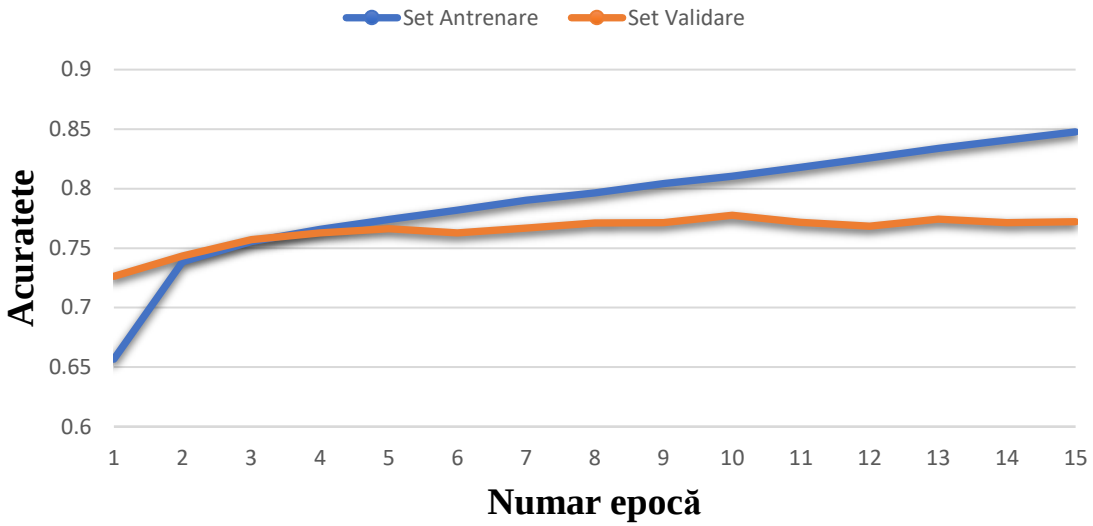

Figura 4.9 Acuratețea modelului LSTM bidirecțional in timpul antrenării

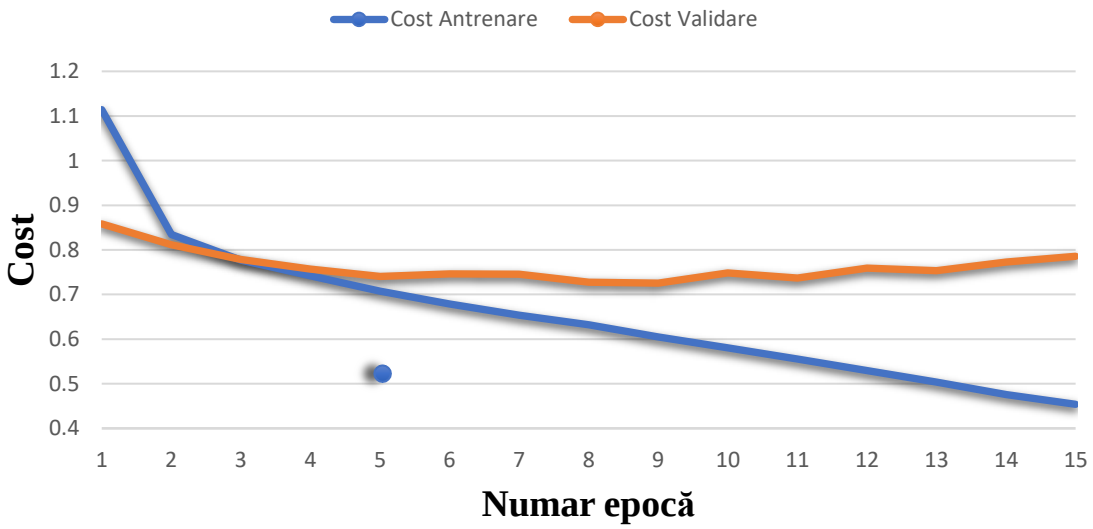

Figura 4.10 Costul modelului LSTM bidirecțional in timpul antrenării

In urma examinării datelor rezultate din procesul de antrenare, prezentate in tabelul 4.4, am decis să utilizez modelul după procesarea epocii cu numărul **9**. Aceasta decizie a fost luată deoarece începând cu epoca 10, costul setului de validate începe un proces de creștere iar modelul intră intre-un proces de overfitting pe setul de antrenare.

4.4.7 Testarea Modelului

După cum se poate vizualiza în tabelul 4.3, rezultatele obținute pe seturile de date de testare și de validare sunt similare, obținând o acuratețe de aproximativ 77%. Astfel modelul depășește substanțial performantele implementării algoritmice XGB (37%) si justifica utilizarea acestuia în rolul de clasificator

Tabel 4.3 Rezultate testare model LSTM bidirecțional pe seturile de date

Set de date	Acuratețe	Cost
Antrenare	0.8041	0.6048
Validare	0.7714	0.7255
Testare	0.7736	0.7733

Scorurile F1 obținute pentru diferitele etichete din setul de test, prezentate in tabelul 4.4, variază între 0.55 si 0.85 având o valoare medie ponderată de 0.76. Fata de clasificatoarele de tip Dummy se poate observa o creștere semnificativă a performantelor, Dummy Most Frequent a obținut o medie a scorurilor F1 de 0.03 iar Dummy Unifrom de 0.06.

Tabel 4.4 Rezultate testare model LSTM bidirecțional pe etichete

Eticheta	Precision	Recall	Scor F1	Support
ARTS & CULTURE	0.55	0.64	0.59	395
BUSINESS	0.62	0.66	0.64	760
CRIME	0.78	0.62	0.69	336
EDUCATION	0.65	0.49	0.55	232
ENTERTAINMENT	0.82	0.76	0.79	2132
ENVIRONMENT	0.66	0.63	0.64	404
FOOD & DRINK	0.77	0.90	0.83	627
HEALTHY LIVING	0.60	0.70	0.65	669
HOME & LIVING	0.84	0.74	0.79	434
POLITICS	0.80	0.85	0.83	3220
RELIGION	0.70	0.67	0.68	265
SCIENCE	0.59	0.56	0.58	439
SPORTS	0.74	0.79	0.76	494
STYLE & BEAUTY	0.88	0.83	0.85	1187
TRAVEL	0.83	0.77	0.79	966

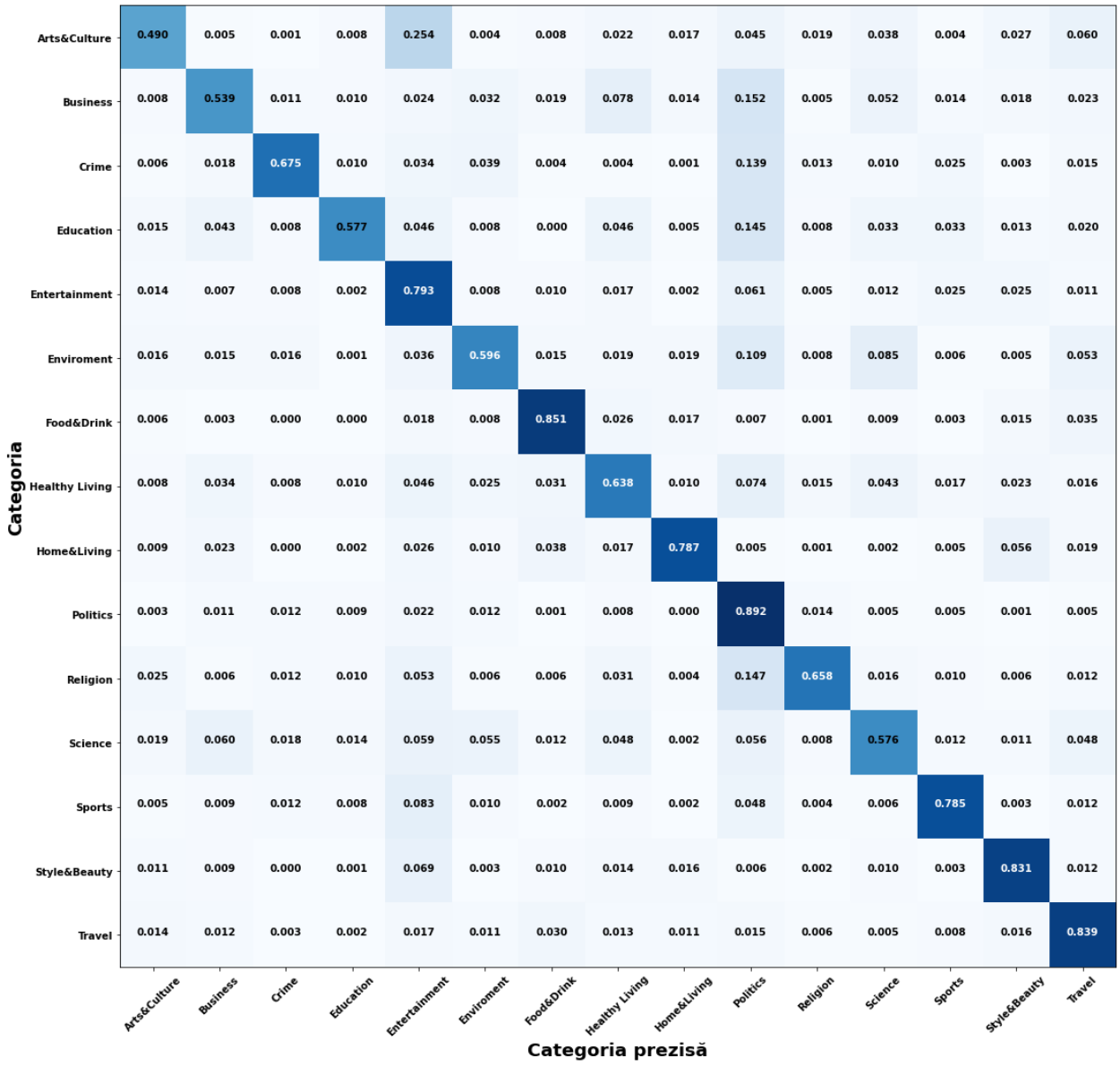

Figura 4.11 Matricea de confuzie normalizata a modelului LSTM bidirecțional

Din matricea de confuzie, reprezentată in figura 4.11, se evidențiază faptul că modelul confundă clasa „Arts&Culture” cu clasa „Entertaiment” în aproximativ 25% din cazuri. Acest lucru se poate datora modului în care este alcătuit setul de date, fiecărei știri îi este atribuită o singură etichetă in timp ce in situațiile reale o știre poate face parte din mai multe categorii simultan.

O altă observație este tendința de a plasa in clasa „Politics” o serie de știri din alte categorii. Un număr de 7 clase au peste 10% dintre știri plasate greșit în categoria „Politics”. Acest lucru se poate datora numărului mare de exemple utilizate pentru eticheta „Politics” sau a faptului că acele subiecte au de multe ori caracteristici politice si se încadrează în mai multe categorii simultan.

4.5 Rețeaua neuronală pentru recunoașterea sentimentelor

Aceasta rețea neuronală reprezintă un model de inteligentă artificială dezvoltat cu rolul de a atribuii știrilor un scor de pozitivitate cuprins între 0 si 1 (0 reprezentând o știre foarte negativă iar 1 o știre foarte pozitivă). Datele procesate de model provin de la topicul Kafka „categorised_news" iar predicțiile sunt atașate acestora și înaintate către topicul „procesed_news".

Pentru a determina tipul de model ce se pretează cel mai bine pe problema noastră de recunoaștere a sentimentelor am dezvoltat mai multe tipuri de rețele neuronale :

- Dense neural network (DNN)
- Convolutional neural network (CNN)
- Bidirectional Encoder Representations from Transformers (BERT)
- Long short-term memory recurrent neural network (LSTM)

Fiecare neuron dintr-o rețea neuronală calculează o valoare de ieșire aplicând o funcție specifică valorilor de intrare primite de la stratul anterior. Funcția care este aplicată valorilor de intrare este determinată de un vector de greutăți și prejudecați. Învățarea constă în modificarea iterativă a acestor greutăți și prejudecați. Cele mai des întâlnite dificultăți in dezvoltarea rețelelor neuronale sunt procesele de **underfitting** si **overfitting**

Overfitting reprezintă procesul prin care un model se specializează să recunoască doar setul de date de antrenarea. Acestea devine incapabil să generalizeze rezultatele obținute, acuratețea pe setul de văitarea începe sa scadă sau să stagneze iar costul prezinta valori din ce in ce mai mari. Câteva soluții pentru combaterea acestui fenomen sunt folosirea unui set de date mai mare si mai complex, introducerea unor noi straturi de dropout sau modificarea valorii dropout din straturilor ce prezintă o astfel de posibilitate.

Underfitting este procesul în care o rețea neuronală nu este capabilă să observe relația dintre variabilele de intrare și de ieșire, generând performante scăzute atât pe setul de antrenament, cât și pe cel de testare. Precum și in cazul procesului de overfitting, o soluție pentru ameliorarea acestei probleme este utilizarea unui set de date cu mai multe exemple. O alta opțiune este reprezentată de creșterea numărului de parametrii și a complexității modelului utilizat.

Modelul ce va fi integrat in serviciul de analiză a sentimentelor știrilor va fi determinat pe baza metricilor de evaluare specifice definite anterior. Deoarece avem o problemă de clasificare binară pe un set de date echilibrat, valoare cea mai relevantă este acuratețea.

4.5.1 Data analysis

Am vrut să antrenam și să testam modelele dezvoltate pe un set de date cât mai apropiat de cazul de utilizare din arhitectura propusă. Pe parcursul căutării am constatat ca nu există un set de date cu știți in limba engleză, suficient de complex și etichetat după cerințele noastre. Pentru a ameliora această problemă am ales utilizarea unuia dintre cele mai bine documentate seturi de date pentru problema de recunoaștere a sentimentelor și a unui set de date etichetat manual pentru a testa performantele modelului pe știrile in limba engleză.

Amazon Review Data (2018):

Acest set de date conține 233.1 milioane de recenzii de pe Amazon ce se întind pe o perioadă de 22 ani intre 1995 si 2018. Recenziile includ informații despre produse și utilizatori, evaluări (valori de la 1 la 5) și o recenzie în text.

Setul de date utilizat in dezvoltarea acestui model a fost construit etichetând recenziile cu scorurile de 1 și 2 ca fiind negative iar recenziile cu scorurile de 4 și 5 ca fiind pozitive. Recenziile cu scorul 3 sunt ignorate. În setul de date, clasa 0 reprezintă recenziile negative iar clasa 2 cele pozitive. Fiecare clasă are 20.000.000 de mostre.

Pentru a putea evalua corect performanța modelului dezvoltat**,** trebuie să împărțim setul de date in trei parți distincte, respectiv seturile de antrenare, testare si validare, fiecare fiind utilizat într-o etapă diferită a procesului de dezvoltare. Deoarece avem o problemă de clasificare iar cele doua clase (pozitiv si negativ) conțin un nivel egal de informație, este important să utilizăm un set de date echilibrat pentru a nu introduce in model prejudecați (biases).

Subsetul de antrenare este folosit pentru a antrena modelul să facă predicții corecte. În fiecare epocă datele de antrenament sunt transmise rețelei neuronale în mod repetat, iar modelul continuă să învețe caracteristicile acestora. Acesta este constituit din 32.000.000 de recenzii (80% din setul de date) cu o lungime maxima de 255 cuvinte si un număr mediu de 74.16 cuvinte pe recenzie, fapt evidențiat în figura 4.12.

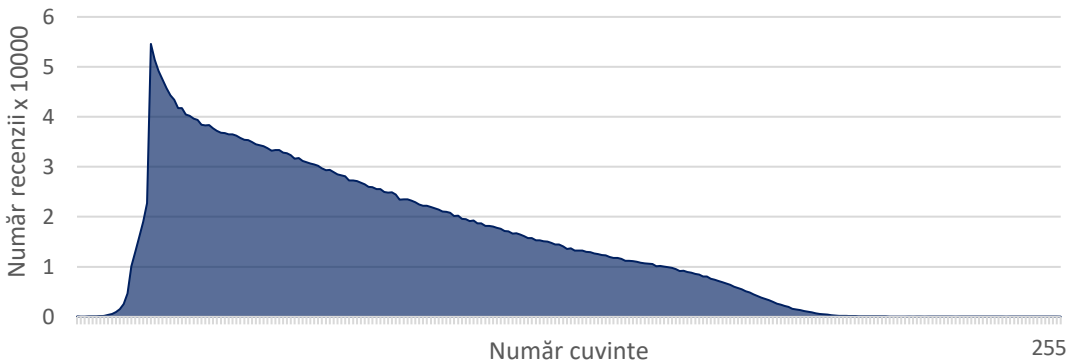

Figura 4.12 Lungimea recenziilor din subsetul de antrenare

Între cele 2 clase prezente setul de antrenament exista o diferența de 8.3% intre lungimea medie a recenziilor pozitive (71.18 de cuvinte) și lungimea medie a recenziilor negative (77.15) , după cum este prezentat in figura 4.12 . Această proprietate este prezentă in toate cele 3 subseturi de date și poate reprezenta o intrare în rețelele neuronale dezvoltate sau o caracteristică la care acestea se pot adapta pentru a face o predicție cât mai corectă.

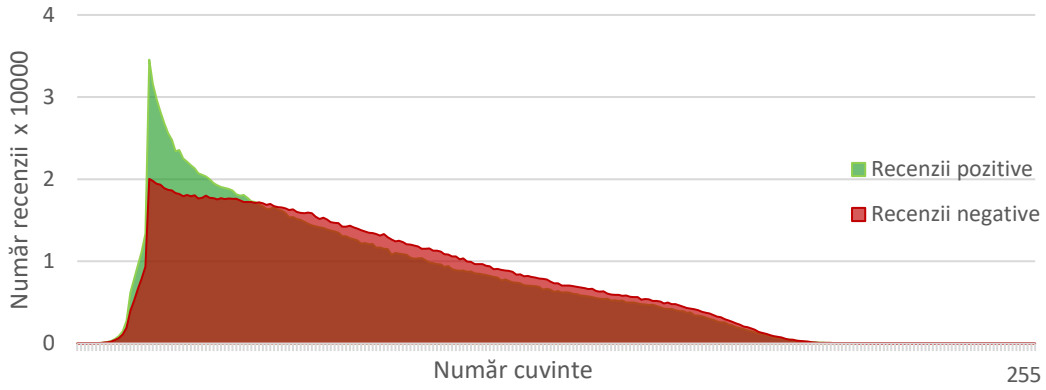

Figura 4.13 Lungimea recenziilor din subsetul de antrenare pe sentiment

Subsetul validare este utilizat pentru a verifica performanța modelului în timpul antrenamentului si reprezintă 10% din totalitatea recenziilor. La fiecare epocă modelul este antrenat pe setul de antrenament si evaluat cu setul de validare, acest proces oferă informații ce ne pot ajuta in reglarea parametrilor modelului. Acesta este constituit din 4.000.000 de recenzii cu o lungime maxima de 219 cuvinte si un număr mediu de 74.17 cuvinte pe recenzie. Fața de setul de antrenament lungimea maxima a scăzut cu 14.12% însă cea medie a rămas constanta, indicând posibilitatea prezentei unor recenzii “outliers” in setul de antrenare.

Subsetul de testare este utilizat pentru a testa modelul după finalizarea antrenării reprezentând 10% din totalitatea recenziilor. Pentru a putea oferii o imagine de încredere a performantelor modelului, recenziile trebuie să prezinte aceeași distribuție a claselor ca și setul de date de antrenament

Acesta conține 4.000.000 de recenzii cu o lungime maximă de 217 cuvinte și un număr mediu de 74.10 cuvinte pe recenzie. Lungimea maximă este in scădere cu 14.91% față de setul de antrenament, având o valoare aproximativ egală cu setul de validare, confirmând prezenta unor “outliers” in setul de antrenare.

Setul de date etichetat manual :

Pentru a testa performanțele modelului dezvoltat in cazul de utilizare specific arhitecturii prezentate, am etichetat 100 de știri provenite din fluxurile RSS ale BBC, CNN si Fox News. Fiecărei știri i-a fost atribuită o valoare binară, 0 reprezintă o știre cu limbaj negativ iar 1 o știre cu limbaj pozitiv. Acest set de date a fost utilizat doar în procesul de testare. El este format din 50 de știri etichetate cu 1 si 50 de știri etichetate cu 0.

Deoarece dimensiunea medie a unei știri este considerabil mai mare decât dimensiunea unei recenzii, acestea au fost împărțite in paragrafe de o dimensiune de maxim 128 de cuvinte (după aplicare pasului de preprocesare). Scorul final atribuit unei știri este reprezentat de media aritmetică ponderată a scorurilor prezise pe paragrafe în funcție de dimensiunea fiecărui paragraf.

Setul de date etichetat manual prezintă 50 de știri pentru fiecare dintre cele două etichete. Acestea sunt împărțite în 20 de știri provenite de la BBC, 20 de la CNN si 10 preluate de la FOX.

Știrile din subsetul de date etichetat pozitiv au o lungime medie de 808,8 cuvinte si o lungime maxima de 2.554 cuvinte, distribuite in medie pe 27.4 paragrafe. Obținem astfel un număr mediu de 29,5 cuvinte pe paragraf. Știrile din subsetul de date etichetat negativ au o lungime medie de 835,2 cuvinte si o lungime maximă de 2.984 cuvinte, distribuite in medie pe 29.1 paragrafe. Obținem astfel un număr mediu de 28,7 cuvinte pe paragraf.

Din datele prezentate anterior se demonstrează că toate subseturile de date extrase din același set au o structură similară si pot fi utilizate in procesele de antrenare, validare sau testare. De asemenea au fost identificate o serie de recenzii neconforme din punct de vedere al lungimii lor, acestea trebuie analizate si extrase in pasul de preprocesare a datelor.

Ca termen comparativ pentru modelul dezvoltat oferim un sistem de recunoaștere aleatoriu. Asupra setului de date **Amazon Review Data** au fost utilizate aceleași doua DummyClassifier ca și in cazul rețelei neuronale pentru recunoașterea categoriilor .

Dummy Most Frequent alege una dintre clase (in cazul prezentat, a fost ales „POZITIVE”) si atribuie aceea eticheta tuturor valorile, obținând un scor de 100% pe clasa aleasă si 0% cealaltă.

Dummy Uniform atribuie fiecărei valori una dintre clase cu o probabilitate de 50%. Astfel se obține o acuratețe de aproximativ 50% cu mici diferențe la fiecare execuție a clasificării.

4.5.2 Preprocesarea datelor

Pentru preprocesarea datelor s-au aplicat tehnici similare cu cele utilizate în dezvoltarea rețelei neuronale pentru recunoașterea categoriilor. Datorită particularităților seturilor de date utilizate a fost nevoie de utilizarea unor tehnici suplimentare de preprocesare

Transformarea cuvintelor in litera mică

Un prin pas in preprocesarea datelor reprezintă transformarea tuturor cuvintelor in litera mică pentru a da o oarecare uniformitate datelor, necesară pentru ulterioarele metode de preprocesarea aplicate.

Eliminarea semnelor de punctuație

Cele trei seturi de date utilizate prezinta o medie cuprinsa intre 6.88 si 6.93 cuvinte la fiecare semn de punctuație utilizat asigurând astfel menținerea unei uniformități a datelor după procesul de curățare.

Tabel 4.5 Numărul de semne de punctuație din subseturile de date

	Antrenament	Validare	Testare
Total	34,366,892	4,295,238	4,292,727
Subset pozitiv	16,434,458	2,056,272	2,054,143
Subset negativ	17,932,434	2,238,966	2,238,584

In tabelul 4.5 se pot observa numărul de semne de punctuație din fiecare set sau subset de date. In urma eliminării acestora, am redus lungimea medie a recenziilor din setul de antrenament cu 74.17, obținând o lungime medie de 73.92 cuvinte, scăzând astfel timpul si resursele necesare pentru antrenarea modelului .

Eliminarea stopwords

Datorită rolului specific pe care trebuie sa îl îndeplinească modelul dezvoltat am ales separarea listei oferite de modulul **NLTK** in doua liste distincte in funcție de conotațiile negative ale cuvintelor (“Stopwords neutre” si “Stopwords negative”). Acestea vor fii analizate in mod individual si eliminate/menținute in funcție de rezultatele obținute.

Stopwords negative :

In contextul dat vom definii un stopword negativ ca orice cuvânt prezent in următorul sir:

['no', 'nor', 'not', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't’ , 'don', ‘don't’]

Tabel 4.6 Procentul de stopwords negative subseturi

	Antrenament	Validare	Testare
Set	1.26%	1.26%	1.26%
Subset pozitiv	0.87%	0.87%	0.87%
Subset negativ	1.62%	1.62%	1.62%

In urma analizei efectuate asupra tabelului 4.6, se poate observa o diferența semnificativa între proporția cuvintelor din categoria stopwords negative din subseturile pozitive si negative , acestea devenind astfel un nou posibil input pentru antrenarea modelului. Am decis păstrarea cuvintelor aparținând acestei categorii deoarece deșii frecventa lor este crescuta, ele conțin un nivel mare de informație necesara procesului de antrenare .

Stopwords neutre :

In contextul dat vom definii un stopword neutru ca orice cuvânt prezent în următorul sir:

['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y']

Tabel 4.7 Procentul de stopwords negative din subseturi

	Antrenament	Validare	Testare
Set	46.05%	46.06%	46.02%
Subset pozitiv	46.14%	46.14%	46.12%
Subset negativ	45.97%	45.97%	45.93%

După cum se poate observa din tabelul 4.7, cuvintele din grupul examinat reprezintă o proporție aproximativ identică din cuvintele aflate în seturile și subseturile analizate. acestea pot fi extrase fără a influența substanțial rezultatele modelului dezvoltat.

Eliminarea cuvintelor aparținând categoriei “Stopwords neutre” definite mai sus va reduce dimensiunea subsetului de antrenament cu aproximativ 46.05%, scăzând astfel timpul de antrenament datorită numărului mai mic de cuvinte implicate in acest proces.

Trunchierea cuvintelor

In urma aplicării metodei de trunchiere a cuvintelor asupra setului de date de antrenare am redus numărul de cuvinte unice din acesta de la 2.062.631 de cuvinte la 1.807.388 , reducând astfel dimensiunea dicționarului necesar pentru transformarea datelor in intrări pentru modelul dezvoltat.

Eliminarea cuvintelor rar întâlnite

Deoarece setul nostru de date este format din recenzii provenite direct de pe Amazon, fără a fi corectate sau preprocesate in prealabil, am decis eliminarea cuvintelor ce se găsesc de mai puțin de 1.000 de ori in cele 40.000.000 de recenzii ale setului de antrenament. Acest pas va filtra eventualele greșeli gramaticale sau cuvinte provenite din limbi diferite de limba engleza.

Prin acest proces am scăzut numărul cuvintelor unice la 17.856, micșorând astfel dimensiunea dicționarului necesar pasului de tokenizare, prin eliminarea unor componente ce dețineau un nivel scăzut de informație pentru modelul dezvoltat.

Eliminarea recenziilor de tip „outlier”

In urma aplicării metodelor anterioare de preprocesare au rezultat 9 recenzii cu o lungime mai mare de 128 de cuvinte. Acestea au dimensiunile de 131, 131, 134, 135, 141, 148, 153, 205 și respectiv 223 cuvinte și sunt formate din utilizări repetate ale stopwords-urilor negative „no” sau „don't” .

Prin eliminarea acestora am limitat dimensiunea maxima a recenziilor la 128 de cuvinte, reducând astfel timpul si resursele necesare pentru antrenarea modelului.

Tokenizarea datelor

Acest proces este similar cu cel utilizat in dezvoltarea rețelei neuronale pentru recunoașterea categoriilor, tokenizatorul fiind antrenat pe setul de date Amazon Review Data. In urma acestui pas se efectuează un proces de extindere a datelor astfel încât recenziile să se reprezinte printr-un vector de 128 de valori.

Embedding

Procesul de embedding necesită transformarea valorile din cadrul vectorului reprezentativ pentru știre in vectori de numere reale. Fiecare valoare a fost extinsă la un vector de 128 de numere reale.

4.5.3 Dense neural network

Rețelele neuronale dense sunt modele secvențiale formate în principal din straturi total interconectate. Acestea nu necesită resurse de memorie sau computaționale sporite datorita numărului de parametrii relativ scăzut comparativ cu alte tipuri de rețele neuronale.

Tabel 4.8 Straturile modelului DNN

Nume strat	Tip strat	Dimensiune ieșire	Număr dparametrii
Dense-1	Dense	(None, 128, 128)	12416
Dropout-1	Dropout	(None, 128, 128)	0
Global_max_pooling1d	Global_	(None, 128)	0
Dense-2	Dense	(None, 32)	4128
Dropout-2	Dropout	(None, 32)	0
Dense-3	Dense	(None, 2)	66

După cum este prezentat in tabelul 4.8 ,modelul este format din 6 straturi cu 16.610 de parametrii antrenabili, ceea ce îi face cel mai rapid model din punct de vedere al vitezei de antrenare si predicție. Cele 3 tipuri de straturi ce formează rețeaua neuronală descrisa sunt:

- Straturile complet conectate de tipul „**Dense”** ce conectează fiecare neuron din stratul curent cu fiecare neuron din stratul precedent.
- Straturile de tip **Dropout** reprezintă o metoda de a ameliora procesul de overfitting. Acestea păstrează la fiecare pas din procesul de antrenare un procent din parametrii nemodificați. În cadrul acestui model a fost ales un procent de 30%.
- Stratul **GlobalMaxPooling1D** are rolul de a transforma o matrice bidimensionala într-un vector. Fiecare element al vectorului reprezintă valoarea maxima aflată pe linia corespunzătoare din matrice.

Straturile **Dense-1** si **Dense-2** utilizează funcția de activare „relu”. **Dense-3** reprezintă stratul de ieșire si se folosește de funcția „softmax” pentru a returna două valori intre 1 si 0 reprezentative pentru scorurile de pozitivitate , respectiv de negativitate atribuite intrării. Modelul a fost compilat utilizând funcția de cost „Categorical Crossentropy” si optimizatorul „Adam” cu valoarea 0.001.

Antrenare

Modelul a fost antrenat pe setul de antrenare si validat la finalul fiecărei epoci pe setul de validare. Au fost folosite argumentele batch_zise=32 și shuffle=True. Procesul s-a întins pe 10 epoci si este evidențiat in figurile 4.14 si 4.15 .

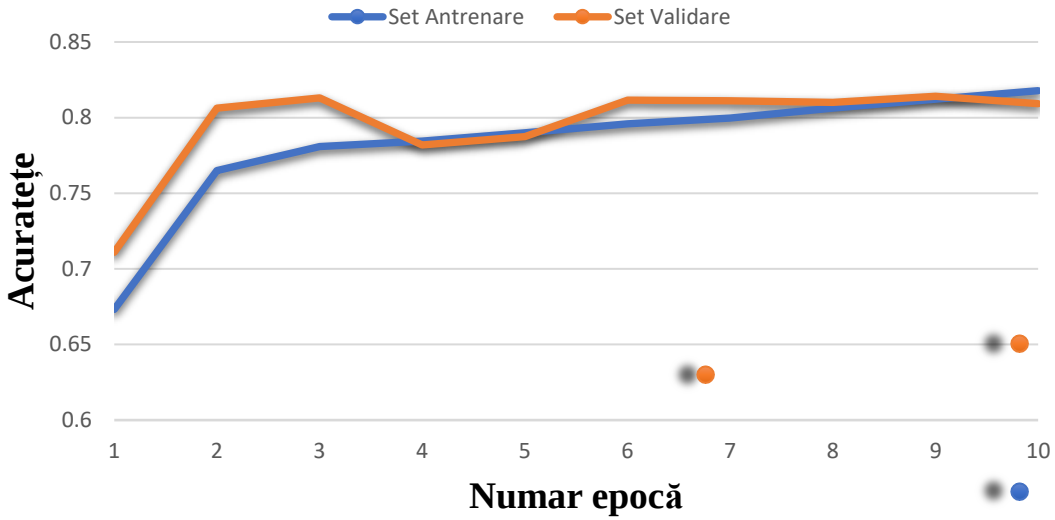

Figura 4.14 Acuratețea modelului DNN in timpul antrenării

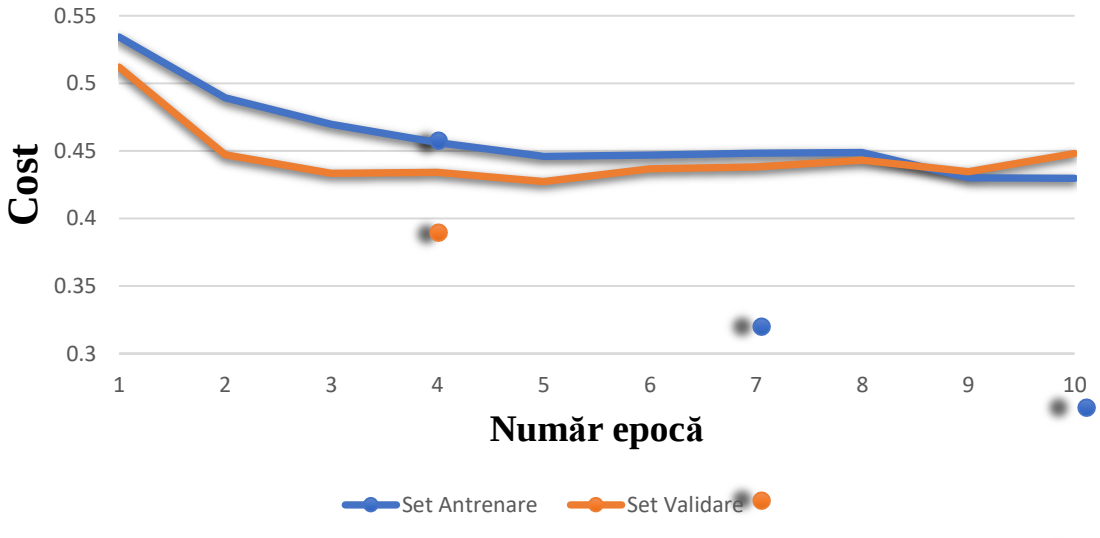

Figura 4.15 Costul modelului DNN in timpul antrenării

In urma examinării datelor rezultate din procesul de antrenare am ales utilizarea modelul după procesarea epocii cu numărul **9.** Aceasta versiune a modelului DNN va fi folosită în timpul testării.

Testarea

După cum se poate vizualiza in tabelul 4.9, modelul prezintă valori similare pentru acurateţe si cost pe seturile de date derivate din Amazon Review Data (2018) [5] , dar prezintă o scădere substanţială a performanţei pe setul de date etichetat manual. Acest lucru se poate datora faptului că modelul nu se poate adapta la limbajul utilizat in cadrul ştirilor sau datorită modului de calcul al valorii finale al prezicerii pe aceste ştiri.

Tabel 4.9 Rezultate testare model DNN pe seturile de date

Set de date	Acurateţe	Cost
Antrenare	0.8122	0.4301
Validare	0.8141	0.4346
Testare	0.8112	0. 4327
Etichetat Manual	0.6400	

Faţa de clasificatoarele de tip Dummy, ce au obţinut o acurateţe de 0.50, modelul DNN a învăţat din datele procesate si a reuşit să genereze predicţii cu o acurateţe de 0.81. De asemenea şi pe setul de date etichetat manual, modelul a obţinut o acurateţe de 0.64, acest lucru arată că reţeaua neuronală a reuşit să extragă caracteristicile fiecărei clase.

4.5.4 Convolutional neural network

Reţelele neuronale convoluţionale sunt în general utilizate pentru detecţia sau recunoaşterea obiectelor sau a persoanelor în imagini, însă pot fi specializate şi pe alte tipuri de probleme, de exemplu NLP.

In general acest tip de modele conţin straturi convoluţionale de diferite dimensiuni ce procesează intrări matriciale. Datorita acestui lucru deseori acestea trebuie utilizate alături de straturi de pooling ce au ca scop transformarea unei intrări bidimensionale într-o ieşire de forma vectoriala prin diferite tehnici.

După cum se poate vizualiza in tabelul 4.10, în total modelul prezintă 1.241.646 de parametrii antrenabili distribuiţi pe 12 straturi. Acesta a fost compilat utilizând funcţia de cost „Binary Crossentropy” şi optimizatorul „Adam” cu valoarea 0.001.

Straturile **Conv1D** sunt compuse dintr-un număr de filtre cu dimensiuni configurabile. Acestea efectuează o operaţie de convoluţie între un vector şi filtru, producând ca ieşire un vector nou de dimensiune egală cu număr de filtre.

Tabel 4.10 Straturile modelului CNN

Nume strat	Tip strat	Dimensiune ieșire	Număr parametrii
input_1	InputLayer	(None, 128)	0
Embedding-1	Embedding	(None, 128, 128)	1.048.576
conv1d_1	Conv1D	(None, 127, 100)	25.700
Global_max_pooling1d_1	Global_1d	(None, 100)	0
conv1d_2	Conv1D	(None, 126, 100)	38.500
Global_max_pooling1d_2	Global_1d	(None, 100)	0
conv1d_3	Conv1D	(None, 125, 100)	51.300
Global_max_pooling1d_3	Global_1d	(None, 100)	0
Concat	Concat	(None, 300)	0
Dense-1	Dense	(None, 256)	77.056
Dropout-1	Dropout	(None, 256)	0
Dense-2	Dense	(None, 2)	514

In cadrul modelului prezentat s-au utilizat 3 straturi de tipul Conv1D cu un număr de 100 de filtre si dimensiunea filtrelor egală cu 1, 2 și 3. Toate aceste straturi sunt conectate la starul Embedding-1, obținând astfel 3 dimensiuni de ieșire diferite : (None, 127, 100), (None ,126, 100), și (None, 125, 100).

Cele 3 straturi **Global_max_pooling1d** sunt conectate la câte un strat **Conv1D** reducând astfel cele 3 dimensiune diferite la un vector de 100 de valori. Valorile celor 3 straturi sunt apoi concatenate in stratul **Concat** pentru a putea fi procesate împreuna ca un strat cu dimensiunea de ieșire (None, 300).

Antrenare :

Modelul a fost antrenat pe setul de antrenare și validat la finalul fiecărei epoci pe setul de validare. Procesul s-a întins pe 10 epoci si este evidențiat in figurile 4.16 si 4.17 .

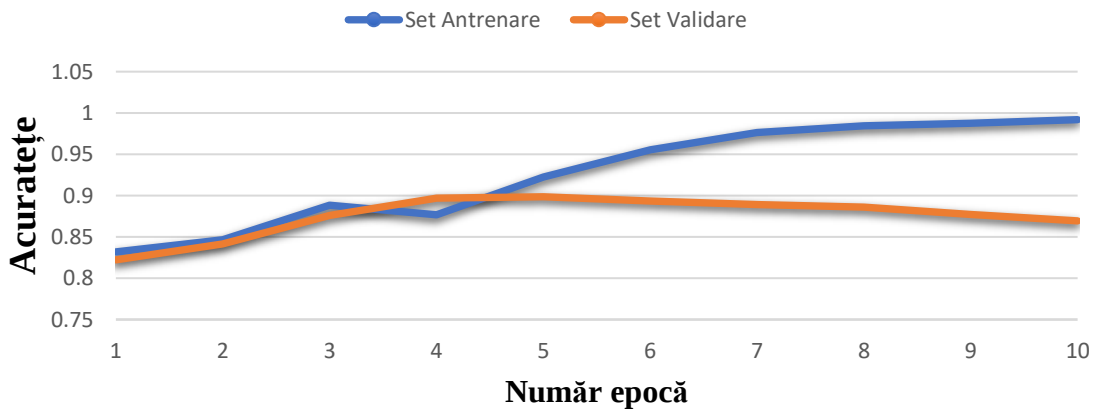

Figura 4.16 Acuratețea modelului CNN pe timpul antrenării

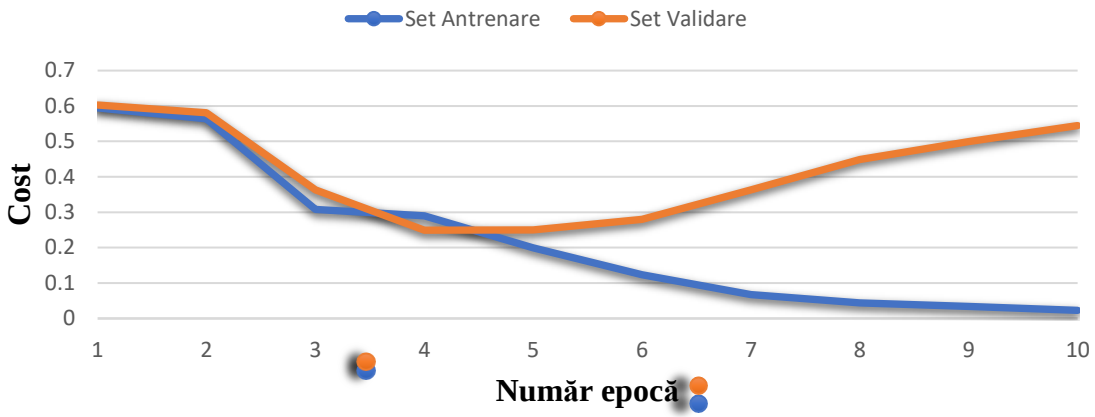

Figura 4.17 Costul modelului CNN pe timpul antrenării

In urma examinării datelor rezultate din procesul de antrenare, performantele cele mai bune s-au obținut la finalul epocii cu numărul 5. Acest lucru se datorează faptului că modelul intră intru-un proces amplu de overfitting începând cu epoca 6.

Testare :

Tabel 4.11Rezultate testare model CNN pe seturile de date

Set de date	Acuratețe	Cost
Antrenare	0.9225	0.1995
Validare	0.8986	0.2499
Testare	0.8987	0.2453
Etichetat Manual	0.7300	

Rezultatele obținute pe seturile de date de testate si de validare, evidențiate în tabelul 4.16, sunt similare, obținând o acuratețe de 89.86% si 89.87%. Se poate observa începutul unui proces de overfittig pe setul de date de antrenare ce devine evident in epocile ulterioare, acuratețea având o valoare de 92.25%.

Modelul prezintă performante mai ridicate fața de DNN pe toate seturile de antrenare, incluzând setul etichetat manual unde acuratețea a crescut cu 9%. Acest lucru denotă faptul că modelul generalizează mai bine decât modelul prezentat anterior.

4.5.5 Bidirectional Encoder Representations from Transformers

Un transformator reprezintă o arhitectura propusă de către Google in anul 2017 bazată pe un mecanism ce se folosește de măștii de atenție (un vector de 1 si 0 ce identifica cuvintele ce influențează modelul dezvoltat).

Măștile de atenție ne permit să trimitem date către transformator chiar și atunci când exemplele au lungimi variabile. Pentru a obține acest comportament, toate datele sunt aduse la aceeași lungime, apoi utilizând tensorul „attention_mask" putem identifica care dintre componente sunt padding.

BERT este un cadru open source de învățare automată pentru procesarea limbajului natural folosind transformatori bidirecționali. Acesta este conceput pentru a ajuta computerele să înțeleagă semnificația limbajului ambiguu în text, folosind textul din jur pentru a stabili contextul. BERT a fost antrenat în prealabil folosind text de pe Wikipedia și poate fi ajustat cu seturi de date specifice pentru a specializa modelul.

Distilarea reprezintă procesul de antrenare a unui model mai mic, numit student, să imite rezultatele obținute de un model mai complex, numit profesor. Acest pas este util când dorim să transferam un model ce rulează pe o componentă hardware cu resurse considerabile pe o mașina cu resurse limitate. Modelul student rulează mai repede și ocupă mai puțin spațiu.

DistilBERT reprezintă un model mai mic si mai rapid obținut prin procesul de distilare a modelului BERT ce conține cu aproximativ 40% mai puțini parametrii si păstrează peste 95% din performatele modelului original. Acesta are 6 straturi comparativ cu cele 12 ale modelului BERT si 66 de milioane de parametrii față de cei 110 milioane .

Antrenare :

Modelul a fost antrenat pe setul de antrenare si validat la finalul fiecărei epoci pe setul de validare. Procesul s-a întins pe 12 epoci si este evidențiat in figurile 4.18 si 4.19 .

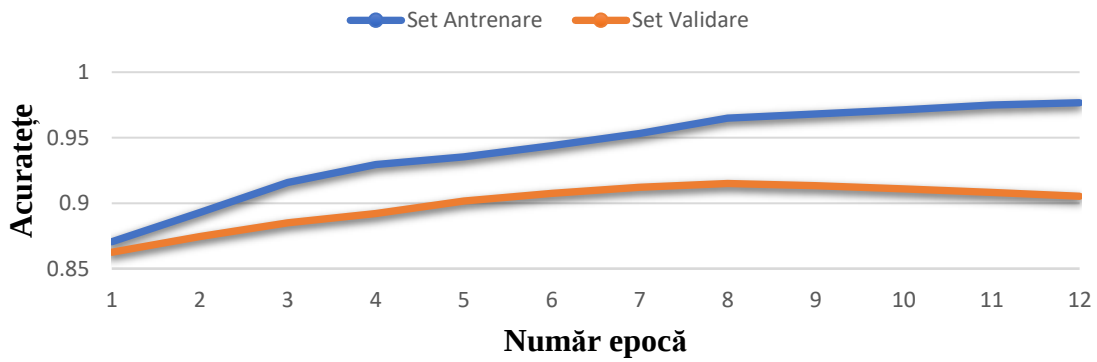

Figura 4.18 Acuratețea modelului DistilBERT pe timpul antrenării

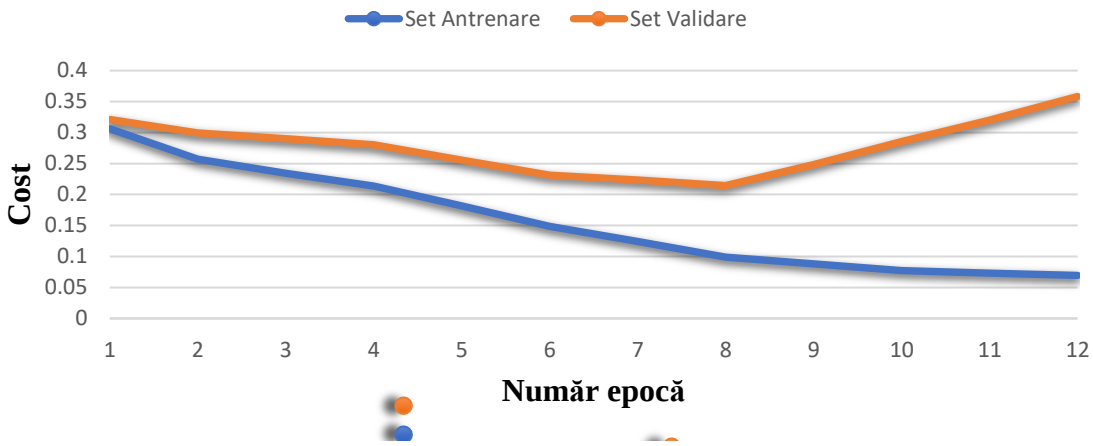

Figura 4.19 Costul modelului DistilBERT pe timpul antrenării

În urma examinării datelor rezultate din procesul de antrenare performantele cele mai bune s-au obținut la finalul epocii cu numărul 8, iar acel model se va folosii in procesul de testare.

Testare :

Rezultatele obținute pe seturile de date de testate si de validare sunt similare, obținând o acuratețe de 91.15% si 91.16% . Se poate observa un proces de overfittig pe setul de date de antrenare, acuratețea având o valoare de 96.48%.

Modelul s-a adaptat mai bine decât modelele prezentate anterior la setul de date etichetat manual. Acesta a obținut o acuratețe de 82% substanțial mai mare decât valorile obținute de DNN si CNN pe același set de date.

Tabel 4.12 Rezultate testare model DistilBERT pe seturile de date

Set de date	Acuratețe	Cost
Antrenare	0.9648	0.0992
Validare	0.9115	0.2144
Testare	0.9116	0. 2151
Etichetat Manual	0.8200	

4.5.6 Long short-term memory (LSTM)

Long Short Term Memory Network este un tip de rețea secvențială RNN care permite modelului să țină cont de datele procesate anterior in modificarea greutăților fiecărui neuron.

Am ales dezvoltarea a 3 modele ce au in componentă 1, 2 sau respectiv 3 straturi LSTM. Structurile straturilor acestora sunt prezentate în tabelurile 4.13, 4.14 si respectiv 4.15 .

In modul implicit un strat LSTM are o intrare bidimensionala iar ieșirea este reprezentata de un tensor vectorial. Pentru a putea înlănțuii mai multe straturi avem nevoie de setarea parametrului „return_sequences=True” ce va returna un vector al stărilor stratului ascuns pentru fiecare valoare de intrare, obținând astfel o ieșire bidimensionala.

Tabel 4.13 Straturile modelului LSTM-1

Nume strat	Tip strat	Dimensiune ieșire	Număr de parametrii
Embedding-1	Embedding	(None, 128, 256)	8.388.608
Dropout-1	Dropout	(None, 128, 256)	0
LSTM-1	LSTM	(None, 128, 128)	197.120
Dense-1	Dense	(None, 128)	165.12
Dropout-3	Dense	(None, 128)	0
Dropout-1	Dropout	(None, 256)	0
Dense-2	Dense	(None, 2)	258

Modelul LSTM-1 din 8 straturi dintre care 1 strat este de tipul LSTM. Acesta are 8,602,498 de parametrii antrenabili.

Tabel 4.14 Straturile modelului LSTM-2

Nume strat	Tip strat	Dimensiune ieșire	Număr de parametrii
Embedding-1	Embedding	(None, 128, 256)	8.388.608
Dropout-1	Dropout	(None, 128, 256)	0
LSTM-1	LSTM	(None, 128, 128)	197.120
Dropout-2	Dropout	(None, 128, 128)	0
LSTM-2	LSTM	(None, 128, 128)	131.584
Dense-1	Dense	(None, 128)	16.512
Dropout-4	Dense	(None, 128)	0
Dropout-1	Dropout	(None, 256)	0
Dense-2	Dense	(None, 2)	258

Modelul LSTM-2 din 10 straturi dintre care 2 straturi sunt de tipul LSTM. Acesta prezinta 8,734,082 de parametrii antrenabili.

Tabel 4.15 Straturile modelului LSTM-3

Nume strat	Tip strat	Dimensiune ieșire	Număr de parametrii
Embedding-1	Embedding	(None, 128, 256)	8.388.608
Dropout-1	Dropout	(None, 128, 256)	0
LSTM-1	LSTM	(None, 128, 128)	197.120
Dropout-2	Dropout	(None, 128, 128)	0
LSTM-2	LSTM	(None, 128, 128)	131.584
Dropout-3	Dropout	(None, 128, 128)	0
LSTM-3	Global_1d	(None, 128)	131.584
Dense-1	Dense	(None, 128)	16.512
Dropout-3	Dense	(None, 128)	0
Dropout-1	Dropout	(None, 256)	0
Dense-2	Dense	(None, 2)	258

Modelul LSTM-3 din 12 straturi dintre care 3 straturi sunt de tipul LSTM. Acesta are 8,865,666 de parametrii antrenabili.

Cele trei modele au fost compilate utilizând funcția de cost „Categorical Crossentropy” si optimizatorul „Adam” cu valoarea 0.001.

Antrenare:

Toate cele 3 modele au fost antrenate pe setul de antrenarea si validat la finalul fiecărei epoci pe setul de validare. Procesul s-a întins pe 5 epoci și este evidențiat in tabelele 4.16, 4.17 si 4.18 .

Tabel 4.16 Rezultate antrenare model LSTM-1

	Acurateţe Antrenare	Acurateţe Validare	Cost Antrenare	Cost Validare
1	0.8571	0.8361	0.3298	0.3658
2	0.8983	0.8772	0.2506	0.2825
3	0.9124	0.8724	0.2194	0.3189
4	0.9231	0.8819	0.1944	0.2892
5	0.9315	0.8762	0.1743	0.3312

Tabel 4.17 Rezultate antrenare model LSTM-2

	Acurateţe Antrenare	Acurateţe Validare	Cost Antrenare	Cost Validare
1	0.8585	0.8609	0.3299	0.3507
2	0.8983	0.8753	0.2588	0.2891
3	0.9118	0.8761	0.2211	0.2981
4	0.9218	0.8776	0.1990	0.3039
5	0.9298	0.8652	0.1794	0.3237

Tabel 4.18 Rezultate antrenare model LSTM-3

	Acurateţe Antrenare	Acurateţe Validare	Cost Antrenare	Cost Validare
1	0.8581	0.8475	0.3315	0.3660
2	0.8983	0.8753	0.2588	0.2891
3	0.8794	0.9111	0.2878	0.2488
4	0.9051	0.8803	0.2346	0.3055
5	0.9145	0.8845	0.2132	0.3071

Din datele evidenţiate prezentate anterior, am determinat epoca in care fiecare dintre modele a atins performantele optime. Astfel in cadrul procesului de testare vom utiliza modelele LSTM-1 și LSTM-2 după finalizarea epocii cu numărul 4 și modelul LSTM-3 aflat in starea după procesare epocii cu numărul 3.

Testare:

Tabel 4.19 Performantele modelelor LSTM pe subseturile de date

	LSTM-1	LSTM-2	LSTM-3
Antrenare	0.9231	0.9218	0.8794
Validare	0.8819	0.8776	0.9111
Testare	0.8794	0.8812	0.9034
Etichetat Manual	0.8100	0.8100	0.8400

După cum se poare observa din tabelul 4.19, cele mai bune performate, dintre toate modele dezvoltate, au fost obţinute de cate LSTM-3 cu 3 straturi LSTM.

În urma rezultatelor obţinute din testarea modelelor dezvoltate, expuse în tabelul 4.20, am ales utilizarea modelului LSTM in cadrul arhitecturii prezentate datorita acurateţei obţinute pe setul de date etichetat manual. DistilBERT a obţinut rezultate marginal mai bune pe setul de date de testare însă in cazul de utilizare necesar, setul de date etichetat manual este mai relevant.

Tabel 4.20 Acurateţea modelelor pe seturile de date

	Setul de Testare	Setul Etichetat Manual
DNN	0.8112	0.6400
CNN	0.9225	0.7300
DistilBERT	0.9116	0.8200
LSTM	0.9034	0.8400
Dummy Most Frequent	0.5000	0.5000
Dummy Uniform	0.5000	0.5000

4.6 MySql Database

Într-o bază de date relațională, toate datele sunt stocate în tabele. Un tabel este o structură bidimensională compusa din rânduri și coloane. O coloană reprezintă o proprietate a entității descrise de tabel iar un rând reprezintă o entitate definita pe baza proprietăților.

In cadrul arhitecturii am utilizat o baza de date relaționala MySql containerizata pentru a stoca informațiile legate de conturile de utilizator, preferințe sau istoricul accesării.

Acesta conține o singura tabela, denumita „users”, având următoarele câmpuri:

- **ID** – int AUTO_INCREMENT
- **Username** – varchar(128)
- **Password** – varchar(128)
- **Email** – varchar(128)
- **Preference** – varchar(512)
- **Visited** – varchar(512)

Adăugarea sau ștergerea din baza de date se face prin intermediul serverului NodeJs, in funcție de cererile primite de la client.

4.7 Server

Node.js este un mediu de rulare JavaScript open-source ce permite execuția de cod JavaScript în afara unui browser web. In cadrul arhitecturii propuse acesta are rolul de a răspunde cererilor trimise de către interfața grafica.

Serverul comunica in mod direct cu REST API-ul de similaritate și cele două baze de date MySql și ElasticSearch prin intermediul cărora colectează, inserează si procesează date.

Pentru o experiența mai plăcută a utilizatorului, serverul implementează un mecanism de sesiuni prin intermediul framework-ului **Express** astfel încât contul de utilizator al clientului sa rămână conectat la fiecare accesare a interfeței web.

Parola utilizatorului nu este păstrata in clar, aceasta are o funcție de hash aplicata iar rezultatul este introdus in câmpul Password din tabela „users”.

4.8 Similaritate REST API

Serviciul **Similaritate REST API** are rolul de a identifica știrile similare cu o anumită știre al cărei ID este primit ca argument întru-un HTTP Request și a fost dezvoltat utilizând framework-ul Flask pentru limbajul de programare Python.

Acesta preia direct din baza de date ElasticSearch titlul, descrierea, data publicării, câmpul RSSTag și conținutul știrii pe baza ID-ului recepționat. Având aceste informații, programul extrage toate știrile ce au fost publicate la o diferența de timp de maxim 2 zile față de data publicării știrii inițiale. A fost aleasă restricția de 2 zile pentru a sporii timpul de răspuns al serviciului si pentru a detecta știrile ce descriu același eveniment.

Prin intermediul modelului **Universal Sentence Encoder** oferit de către modulul **„spaCy”** toate aceste date trec printr-un proces de embedding. Am ales folosirea acestui model deoarece a fost antrenat special pentru a oferi embeddings relevante pentru sentence similarity.

Calculul similarității dintre doi vectori de embeddings se face prin intermediul **„Cosine Similarity”** definit prin următoarea formula 4-7 .

$$similarity(A,B) = \cos(\theta) = \frac{A*B}{||A||*||B||} \textbf{(4-7)}$$

Pentru a micșora timpul de răspuns al sistemului calculul similarității se face în două etape. Inițial se calculează similaritatea dintre titlurile știrilor, iar dacă acesta are o valoare mai mare sau egală cu 0.3 se calculează și similaritatea pe baza descrierii și a conținutului, in caz contrar știrea nu este procesată mai departe.

Acest serviciu este apelat la fiecare accesare a unei știri din interfața grafică de către utilizator. Clientul trimite o cerere către server iar acesta înaintează un HTTP Request către Similaritate REST API, așteaptă rezultatul obținut si întoarce un răspuns adecvat către client.

Serviciul returnează primele 5 știri cu scorul de similaritate cel mai mare cu știrea al care ID a fost primit, in format JSON.

4.9 Interfața web

Pagina principală

Pagina principală prezintă utilizatorului știrile aflate in baza de date în ordinea publicării acestora, după cum este prezentat in figura 4.20 . Acestea sunt distribuite in diferite card-uri colorate în funcție de publicațiile de proveniența ce expun titlul, descrierea si data publicării unei știri. Navigarea intre paginile cu știri se face prin intermediul butoanelor „Prev” si „Next” aflate pe bara de jos sau prin apăsarea pe butonul cu numărul corespunzător paginii dorite.

Filtrele sunt accesate prin apăsarea butonului „FILTERS” din partea stângă. Fiecare dintre acestea se poate extinde individual prin apăsarea butonului din partea dreapta a căsuței corespunzătoare.

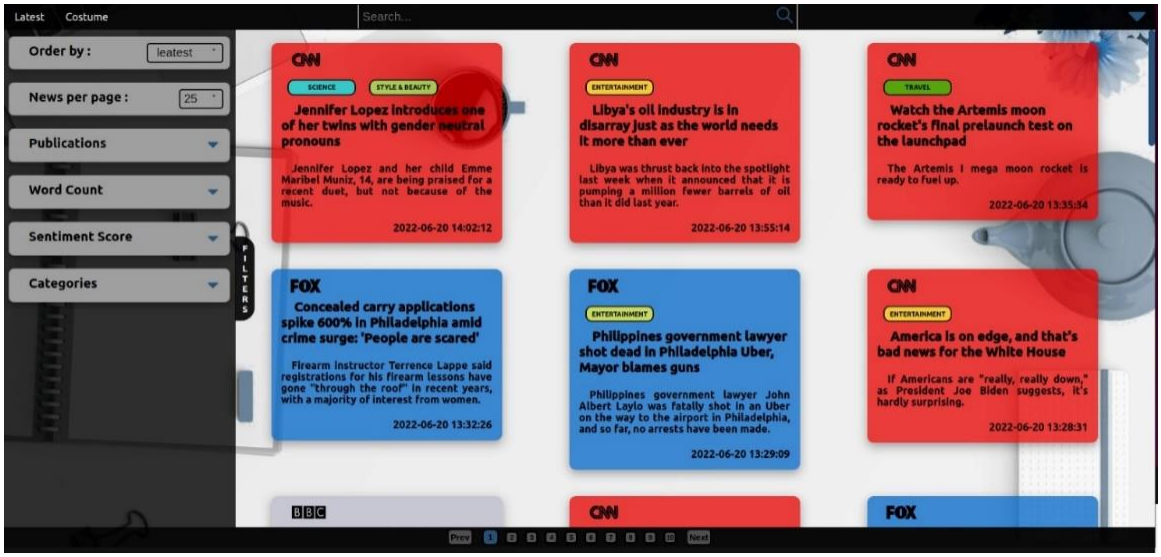

Figura 4.20 Meniul cu filtre

Pagina de înregistrare si autentificare

Pagina de înregistrare si autentificare este împărțita in două meniuri separate destinate pentru autentificarea într-un cont existent si înregistrarea unui nou utilizator . Navigarea între acestea este realizata prin intermediul butoanelor „Login” si „Register”.

Înregistrarea unui nou cont de utilizator se realizează prin introducerea unui nume de utilizator, o parola si a unei adrese de email urmat de apăsarea butonului „Register”.

Autentificarea unui utilizator se realizează prin introducerea numelui de utilizator si a parolei alese in pasul de înregistrare a unui nou cont urmat de apăsarea butonului „Login”.

Pagina de vizualizare a știrilor

Pagina pentru vizualizarea a unei știri se accesează prin apăsarea pe unul dintre card-urile cu știri din pagina principala sau pagina personalizata a utilizatorului. Modul de prezentare a acesteia se poate observa in figura 4.21.

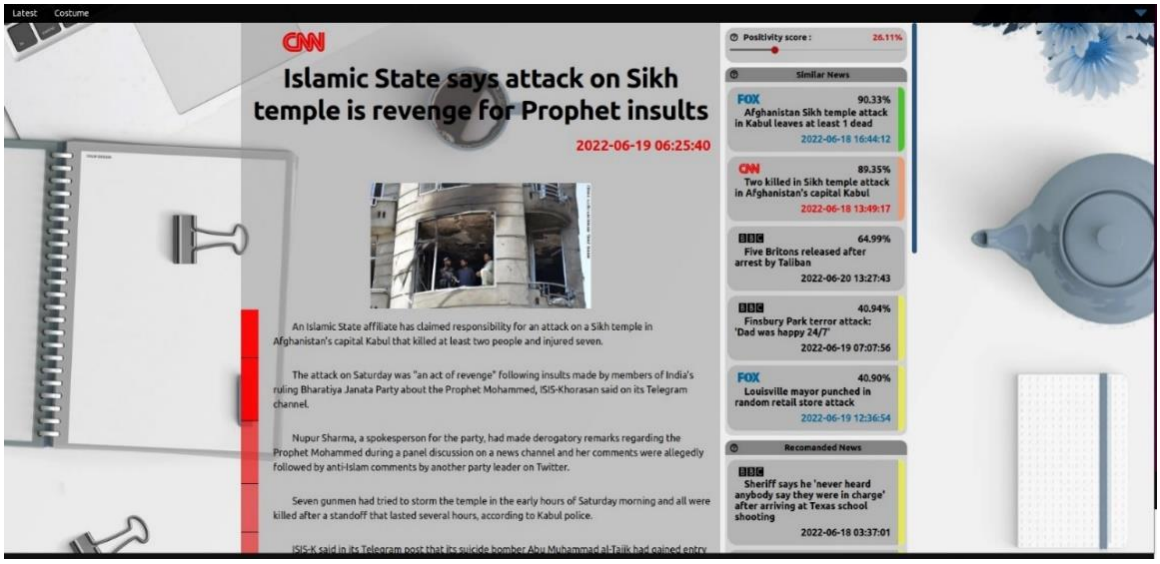

Figura 4.21 Pagina de vizualizare a știrilor

Aceasta prezinta in partea de mijloc a paginii publicația de proveniența, data publicării, categoriile din care face parte știrea si titlul alături de conținutul acesteia distribuit pe parafare.

Fiecare paragraf are in partea stânga o culoare ce reprezintă scorul acesteia dat de către rețeaua neuronală pentru recunoașterea sentimentelor (roșu – sentiment negativ; albastru – sentiment pozitiv) .

In partea dreapta sus a paginii se afla scorul de pozitivitate dat de către modelul dezvoltat pe totalitatea articolului urmat de secțiunea de știri similare indicate de către Similaritate REST API . Ultimul punct de interes din partea dreapta este reprezentat de secțiunea de știri recomandate pe baza categoriilor accesate in prealabil de utilizator.

Daca au fost detectate știri cu un scor de similaritate mai mare de 90% atunci în partea de jos a pagini va fi prezentă rubrica de „Story Timeline” , reprezentată in figura 4.30, care ordonează aceste știri in ordine cronologica incluzând si știrea actuala.

Aplicația web implementează un mecanism de popup-uri pentru informarea asupra eventualelor erori sau pentru a oferii feedback utilizatorului. Acestea sunt afișate in partea de dreapta jos a paginii și au o durata de viată de 5 secunde.

Pagina personalizata a utilizatorului

Pagina personalizată a utilizatorului este accesată prin apăsarea butonului „Costume” din partea stânga a barii de navigație a aplicației web.

Daca utilizatorul nu este autentificat pagina va afișa o eroare ce îndruma utilizatorul către pagina de autentificare și înregistrare.

Daca utilizatorul este autentificat pagina personalizata a utilizatorului nu mai întoarce o eroare si prezinta o pagina asemănătoare cu cea principala. Aceasta prezinta butoanele „LOAD” si „SAVE”, prezente in partea stânga a figurii 4.25, care permit salvarea filtrelor actuale sau încărcarea celor salvate anterior.

Pagina de vizualizare a statisticilor

Pagina de vizualizare a statisticilor este accesata din meniul DropDown și prezintă statistici asupra datelor din baza de date. Datele prezentate in aceasta se pot remarca in figura 4.22 .

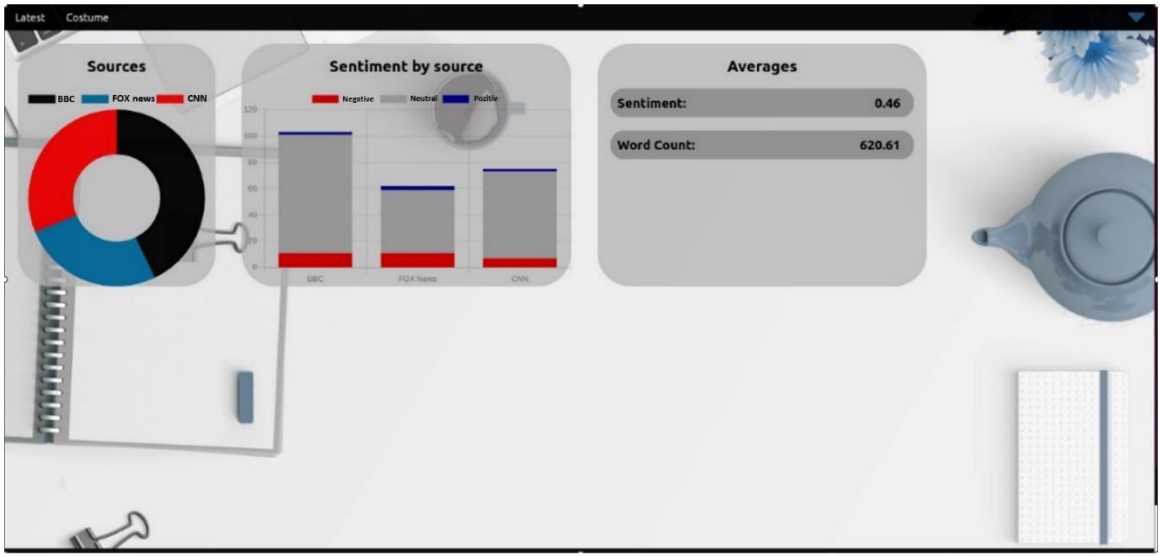

Figura 4.22 Pagina de vizualizare a statisticilor

4.10 Serviciul de colectarea al log-urilor

Serviciile de extragere a datelor din fluxurile RSS, de extragere al textului precum și rețelele neuronale pentru recunoașterea sentimentelor și a categoriilor transmit cate topicul kafka „logs” date in format JSON ce conțin un timestamp alături de știrea procesată. Aceste mesaje sunt ulteriori colectate de un serviciu containerizat ce le introduce într-un nou index in cadrul ElasticSearch.

5 CONCLUZII

Pe parcursul dezvoltării sistemului propus am întâmpinat o serie de provocări ce au limitat performantele acestuia sau au încetinit procesul de implementare. De asemenea am sesizat si o serie de direcții de dezvoltare posibile ale arhitecturii.

Provocările întâmpinate în implementarea proiectului.

În aplicarea rețelei neuronale pentru recunoașterea sentimentelor a apărut problema modului in care trebuie calculat scorul final atribuit fiecare știri. Textele extrase din paginile publicațiilor au dimensiuni mult mai mari fata de recenziile utilizate in pasul de antrenare a modelului. Astfel, acestea au fost împărțite în propoziții sau paragrafe de maxim 128 de cuvinte, după pasul de preprocesare, ce sunt introduse individual în procesul de predicție. Valorile rezultate trebuie agregate într-o singura valoare finala, reprezentativă pentru totalitatea textului.

Aceasta problemă a fost rezolvata prin introducerea setului de date etichetat manual si testarea a mai multor formule de calcul a rezultatului final. Metoda obținută a fost utilizarea unei medii ponderate a valorilor pe baza lungimilor individuale ale paragrafelor. Aceasta a adus o acuratețe de 84% pe setul de date etichetat manual utilizând modelul LSTM-3. O alta soluție a fost implementarea în interfața grafică a posibilității de vizualizare a scorului de sentiment pentru fiecare paragraf în parte.

Rezultate obținute

Rețeaua neuronală pentru recunoașterea categoriilor a obținut prin intermediul modelului bazat pe straturi LSTM bidirecționale o acuratețe de 77% pe setul de date de testare si o medie a scorului lor F1 pe diferitele etichete de 0.76. Aceste rezultate sunt substanțial mai favorabile decât valorile identificate prin utilizarea clasificatorilor Dummy.

In urma testării diferitelor tipuri de modele dezvoltate**,** rețeaua neuronală pentru recunoașterea sentimentelor a obținut o acuratețe de 90.34% pe setul de date de testate utilized modelul LSTM-3 SI 91,16% utilized DistilBERT. Acuratețea acestor modele pe setul de date etichetat manual a fost de 84% si respectiv 82%. Modelele de tipul DNN si CNN au obținut acurateței de sub 90% pe setul de testare.

Îmbunătățiri ale sistemului implementat

Există o multitudine de direcții prin care performantele sistemului se pot îmbunătăți.

Rețeaua neuronală pentru recunoașterea categoriilor se poate perfecționa prin utilizarea unui set de date uniformizat de dimensiune mai mare. Se mai pot explora modele cu un număr crescut de straturi bidimensionale LSTM , alte tipuri de RNN sau modele bazate pe transformatori precum BERT.

Rețeaua neuronală pentru recunoașterea sentimentelor poate beneficia de un proces de perfecționare al parametrilor rețelei LSTM-3. Se poate testa performanta modelului BERT pe întreg setul de date, daca sunt disponibile suficiente resurse computaționale.

In cadrul arhitecturii exista posibilitatea extinderii sistemului de log-are astfel încât să colecteze informații de la fiecare aplicație containerizată.

Funcționalități adiționale

Chiar daca proiectul a implementat toate funcționalitățile prezentate în cadrul cerințelor, arhitectura este proiectata astfel încât se pot adăuga aplicații sau servicii containerizate ușor.

In cadrul subsistemului de colectare a datelor se pot implementa aplicații ce asigura preluarea informațiilor de la publicațiile ce nu expun un flux RSS.

Subsistemul de clasificare poate beneficia de integrarea a multiple rețele neuronale specializate pe diferite problematici ce țin de NLP, precum detecția de știri false sau a emoțiilor .

Interfața web poate fi extinsa prin adăugarea a noi funcționalități menite să sporească interacțiunea cu utilizatorul. De exemplu, se poate implementa un sistem reclasificare a știrilor menit să permită reantrenarea ulterioară a modelelor pe un set de date mai mare. O alta direcție de dezvoltarea este reprezentata de un mod de apreciere a știrilor de către client astfel încât sistemul sa ofere recomandări pe baza acestora.

6 BIBLIOGRAFIE

[1] A. Al Shamsi, Arwa & Bayari, Reem & Salloum, Said. (2021). Sentiment Analysis in English Texts. Advances in Science Technology and Engineering Systems Journal. 5. 1683-1689. 10.25046/aj0506200.

[2] Olga Fuks (2018). Classification of News Dataset

[3] Kaggle News Category Dataset. https://www.kaggle.com/rmisra/news-category-dataset. Accessed: 2022-05-06.

[4] Jianmo Ni, Jiacheng Li, Julian McAuley. Empirical Methods in Natural Language Processing (EMNLP), 2019

[5] Amazon Review Data. https://nijianmo.github.io/amazon/index.html . Accessed: 2021-10-23

[6] Sanh, Victor & Debut, Lysandre & Chaumond, Julien & Wolf, Thomas. (2019). DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.

[7] Porter, M. "An algorithm for suffix stripping." Program 14.3 (1980): 130-137.

[8]PhantomJS - Scriptable Headless Browser. https://phantomjs.org/ Accessed: 2022-05-13.

[9] Kaur, Gurmeet & Bajaj, Karan. (2016). News Classification using Neural Networks. Communications on Applied Electronics. 5. 42-45. 10.5120/cae2016652224.

[10] AllSides. https://www.allsides.com/unbiased-balanced-news Accessed: 2022-03-02

7 ANEXE

Anexa 1 : Lista cuvinte stopwords

['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'no', 'nor', 'not', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't’ , 'don', ‘don't’]

Anexa 2 Rezultate antrenare model LSTM bidirecțional

	Acuratete Antrenare	Acuratete Validare	Cost Antrenare	Cost Validare
1	0.6567	0.7263	1.1142	0.8582
2	0.7385	0.7434	0.8345	0.8118
3	0.7555	0.7571	0.7768	0.7783
4	0.7657	0.7629	0.7419	0.7568
5	0.7741	0.7663	0.7072	0.7407
6	0.7819	0.7628	0.6788	0.7460
7	0.7902	0.7668	0.6534	0.7454
8	0.7964	0.7711	0.6320	0.7274
9	**0.8041**	**0.7714**	**0.6048**	**0.7255**
10	0.8105	0.7776	0.5802	0.7482
11	0.8178	0.7717	0.5554	0.7370
12	0.8258	0.7685	0.5292	0.7590
13	0.8339	0.7743	0.5035	0.7532
14	0.8408	0.7713	0.4755	0.7724
15	0.8477	0.7722	0.4539	0.7853

Anexa 3 Rezultate antrenare model DNN

	Acuratete Antrenare	Acuratete Validare	Cost Antrenare	Cost Validare
1	0.6732	0.7110	0.5343	0.5120
2	0.7650	0.8063	0.4892	0.4471
3	0.7809	0.8130	0.4696	0.4334
4	0.7845	0.7819	0.4562	0.4342
5	0.7899	0.7874	0.4460	0.4273
6	0.7959	0.8115	0.4470	0.4368
7	0.7997	0.8111	0.4483	0.4382
8	0.8063	0.8100	0.4489	0.4431
9	**0.8122**	**0.8141**	**0.4301**	**0.4346**
10	0.8178	0.8093	0.4297	0.4482

Anexa 4 Rezultate antrenare model CNN

	Acuratete Antrenare	Acuratete Validare	Cost Antrenare	Cost Validare
1	0.8319	0.8222	0.5946	0.6027
2	0.8466	0.8417	0.5637	0.5813
3	0.8882	0.8759	0.3077	0.3631
4	0.8766	0.8969	0.2900	0.2493
5	**0.9225**	**0.8986**	**0.1995**	**0.2499**
6	0.9555	0.8934	0.1235	0.2796
7	0.9762	0.8890	0.0677	0.3631
8	0.9844	0.8859	0.0439	0.4487
9	0.9876	0.8771	0.0344	0.4999
10	0.9919	0.8693	0.0229	0.5443

Anexa 5 Rezultate antrenare model DistilBERT

	Acuratete Antrenare	Acuratete Validare	Cost Antrenare	Cost Validare
1	0.8705	0.8625	0.3061	0.3211
2	0.8927	0.8745	0.2572	0.2996
3	0.9159	0.8850	0.2344	0.2903
4	0.9294	0.8920	0.2136	0.2802
5	0.9354	0.9016	0.1817	0.2553
6	0.9440	0.9075	0.1486	0.2310
7	0.9532	0.9122	0.1241	0.2235
8	**0.9648**	**0.9150**	**0.0992**	**0.2144**
9	0.9681	0.9134	0.0879	0.2480
10	0.9713	0.9116	0.0772	0.2854
11	0.9750	0.9083	0.0729	0.3202
12	0.9767	0.9053	0.0694	0.3582