Universitatea Babeș-Bolyai din Cluj-Napoca
Facultatea de Științe Economice și Gestiunea Afacerilor
Școala Doctorală de Științe Economice și Gestiunea Afacerilor

Rezumat Teză Doctorat

Contribuții la Dezvoltarea Resurselor Lingvistice pentru Prelucrarea Textelor Scurte în Limba Română Folosind Învățarea Automată

Doctorand: Dan-Claudiu NEAGU

Conducător de doctorat: Prof. Univ. Dr. Dorina LAZĂR

Cluj-Napoca, 2025

Curpins

Cuprinsul tezei de doctorat

Lista publicațiilor proprii

1. Neagu, D.C.; Rus, A.B.; Grec, M.; Boroianu, M.A.; Bogdan, N.; Gal, A. *Towards Sentiment Analysis for Romanian Twitter Content*. Algorithms 15(10), pp. 357, 2022 https://doi.org/10.3390/a15100357

2. Neagu, D.C.; Rus, A.B.; Grec, M.; Boroianu, M.A.; Silaghi, G.C. *Topic Classification for Short Texts*. In International Conference on Information Systems Development, Cham: Springer International Publishing, pp. 207-222, 2023, https://doi.org/10.1007/978-3-031-32418-5_12

3. Neagu, D.C. *BERTweetRO: pre-trained language models for Romanian social media content*. Studia Universitatis Babeș-Bolyai Oeconomica, Volume 70, Issue 1, pp. 83-111, 2025, https://doi.org/10.2478/subboec-2025-0005

Capitolul 1

Introducere

Supravegherea platformelor social media devine o modalitate eficientă de monitorizare având în vedere creșterea explozivă a numărului de utilizatori la nivel mondial: în februarie 2025 Statista[1] a estimat că aproximativ 5.56 miliarde de persoane au acces la internet și îl folosesc în mod regulat, ceea ce reprezintă $\approx 67.9\%$ din populația globală. Din acest total, 5.24 miliarde, sau $\approx 63.9\%$ sunt utilizatori ai rețelelor de socializare.

România are o populație de puțin peste 19 milioane de locuitori, însă a atins un număr impresionant de 18 milioane de utilizatori de internet la începutul anului 2024, dintre care aproximativ 17.3 milioane sunt activi pe rețelele de socializare. Acest număr reprezintă aproximativ 90% din populația țării și este considerabil mai mare decât media globală[2]. Prin analizarea datelor create și consumate în spațiul online românesc, pot fi identificate informații valoroase despre opiniile publice, tendințe și interese personale, care pot fi utilizate în scopuri științifice sau comerciale.

Procesarea limbajului natural (NLP – *Natural Language Processing*) este un subdomeniu al inteligenței artificiale care oferă computerelor capacitatea de a procesa date codificate în limbaj natural, utilizând metode bazate pe reguli, statistică sau de tip învățare automată (ML – *Machine Learning*), pentru a aborda o gamă largă de sarcini precum [34]: recunoaștere vocală, clasificarea textului, înțelegerea si generarea limbajului natural, etc.

Platformele de microblogging precum Twitter (redenumit ca "X"), Instagram, Facebook sau TikTok inspiră întrebări motivante, deoarece ele prezintă provocări lingvistice rareori întâlnite în textele literare. Eisenstein [23] se referă la acestea ca fiind un *limbaj impropriu*, incluzând emoticoane, abrevieri frazale precum *lol*, *smh*, si *ikr*, alungirea expresivă a cuvintelor (de exemplu *cooool*), sau termeni scriși într-un format neconvențional, inclusiv greșeli de tastare, vocabular neregulat sau gramatică informală. Din diverse motive, care nu intră în domeniul de cercetare al acestui studiu, prezența limbajului impropriu în conținutul de pe rețelele sociale influențează în mod negativ performanța unui model NLP standard aplicat în acest context [51].

Analiza sentimentelor (SA – *Sentiment Analysis*) este o categorie consacrată în NLP, în cadrul căreia multe eforturi de cercetare sunt concentrate pentru a identifica metodele de ML care generează cele mai performante modele pentru o anumită problemă studiată. Manuale precum [39] sau recenzii ca [25, 76] prezintă în detaliu pașii recomandați care trebuie adoptați în mod specific pentru SA sau față de o anumită tehnologie aplicată în NLP în general. Dar, încă există un spațiu larg de cercetare deschis în domeniul

[1] `https://www.statista.com/statistics/617136/digital-population-worldwide/`

[2] `https://www.statista.com/topics/7134/social-media-usage-in-romania/`

SA, în cazul în care apar anumite condiții specifice problemei, cum ar fi cele legate de caracteristicile textelor generate în cadrul platformelor de microblogging, sau gestionarea inputului utilizatorilor de pe dispozitive mobile, etc.

Vizând în mod specific platforma Twitter, realizarea analizei de tip SA pe tweet-uri într-o limbă diferită de cea engleză este considerată o provocare în principal din cauza dificultății de a colecta suficiente date etichetate în limba țintă [10]. Seturile de date etichetate pot fi găsite mult mai ușor pentru limbile populare din lume. Ca exemplu, pentru engleză putem menționa BERTweet [51], un model lingvistic de scară largă antrenat pe un corpus de 850M de tweet-uri, care poate fi utilizat împreună cu *fairseq* [56] sau *transformers* [75] pentru sarcini de clasificare a textului, inclusiv SA. În Franța, provocările DEFT desfășurate între 2014 și 2018 s-au concentrat pe extragerea opiniei și analiza sentimentelor din postările de pe Twitter [54], oferind echipelor participante acces la seturi de date etichetate. În Spania, atelierul TASS[3], organizat anual din 2012 în cadrul congresului SEPLN[4], a furnizat un set de date cu tweet-uri adnotate [19], incluzând variații translingvistice ale limbii spaniole.

Totuși, puține resurse pot fi găsite pentru limbile mai puțin populare ale lumii, cum ar fi limba română. Ciobotaru și Dinu [16] au realizat detectarea emoțiilor pe un set de date de aproximativ 4,000 de tweet-uri în limba română. Textele au fost etichetate manual de către ei, însă setul de date nu a fost făcut public. Istrati și Ciobotaru [29] au colectat și etichetat manual un set de date cu tweet-uri în limba română despre mărci comerciale și au creat un model de SA destinat monitorizării brandurilor. Din păcate, nici acest set de date etichetat manual nu este disponibil publicului. Din câte cunoaștem noi, setul de date recent propus *LaRoSeDa* [70] este primul și singurul set de date public dedicat pentru SA în limba română.

O parte din motivația noastră personală include dezvoltarea unei capacități de predicție a sentimentului în 3 clase ("negativ", "neutru", "pozitiv"), adaptată conținutului specific platformelor de social media. Acest aspect face ca LaRoSeDa să fie un candidat nepotrivit pentru cercetarea noastră, deoarece sentimentul este etichetat într-un mod binar ("negativ" și "pozitiv"), iar textele se referă la recenzii de produse colectate de pe site-uri de cumpărături online, nu de pe platforme de social media.

Descoperirea temelor abstracte care apar într-o colecție de texte sau documente poate fi realizată fie prin Clasificarea Tematică (TC – *Topic Classification*), fie prin Modelarea Tematică (TM – *Topic Modeling*). TM este o tehnică nesupervizată [72], care nu necesită date etichetate, în timp ce TC este o tehnică supervizată, ce presupune existența unor date etichetate pentru antrenarea modelelor.

Pentru acest studiu am ales TC în detrimentul TM din mai multe motive. În primul rând, TM ridică anumite provocări din cauza naturii sale nesupravegheate și a lipsei unor teme predefinite. Spre deosebire de TC, unde numărul temelor este determinat de datele de antrenament, TM poate genera un număr indefinit de teme latente ceea ce îngreunează interpretarea, evaluarea și aplicarea lui în scenarii din viața reală. În al doilea rând, TM necesită intervenție umană pentru a înțelege și eticheta temele latente nou generate, ceea ce adaugă complexitate și un grad de subiectivitate. Pe de altă parte, TC este o abordare directă care asociază documentele cu clase deja cunoscute, permițând astfel obținerea unor concluzii mai concrete.

În ceea ce privește TC pentru limba română, menționăm studiul realizat de Vasile et al. [71], în care au fost utilizate mai multe modele tradiționale pentru a clasifica 219

[3] `http://tass.sepln.org/`

[4] `http://www.sepln.org/workshops/neges2019/`

postări de pe bloguri în 9 clase tematice distincte. Algoritmii Sequential Minimal Optimization (SMO) și Complement Naive Bayes (CNB) au obținut cele mai bune rezultate cu o acuratețe de 77.8%, k Nearest Neighbors (k-NN) a obținut un scor puțin mai mic de 73.3%, iar modelul clasic Naive Bayes (NB) a avut cea mai slabă performanță cu o acuratețe de 68.9%.

Alte lucrări de cercetare privind conținutul de tip microblogging în limba română lipsesc în ciuda beneficiilor sociale și economice care pot fi obținute din utilizarea sistemelor TC. Această lipsă poate fi explicată de caracteristicile problematice ale textelor de pe rețelele sociale, însă, datorită progreselor recente din domeniul ML, pot fi utilizate modele mult mai complexe, precum cele de tip Transformers, pentru a aborda aceste provocări. Un alt demers care ar putea încuraja cercetarea în această direcție ar fi colectarea și adnotarea unor seturi de date noi de mari dimensiuni.

Integrând componentele SA și TC, ne propunem să dezvoltăm un sistem NLP capabil să proceseze date extrase de pe platformele de social media românești din două perspective. Componenta SA va avea rolul de a prezice polaritatea globală a sentimentelor din texte, în timp ce componenta TC se va concentra pe clasificarea acestora în funcție de un set de teme de discuție predefinite. Cu ajutorul acestor funcționalități, cercetătorii sau entitățile private pot obține, aproape în timp real, o înțelegere mai clară a opiniilor publice și a subiectelor de interes actual.

Obiectivul nostru principal este de a aborda problema resurselor lingvistice limitate care pot fi utilizate pentru antrenarea unor modele fiabile de SA sau TC pentru textele din social media-ul românesc. Traducerea automată a seturilor de date populare în diferite limbi a fost prezentată în studii precum [7, 6], sugerând că anumite modele oferă performanțe similare, indiferent de setul de date utilizat pentru antrenare.

Ca obiectiv secundar, ne propunem să oferim un studiu comparativ cuprinzător între diferite abordări ML. Prin antrenarea și testarea unui spectru larg de modele ML, sperăm să răspundem unor întrebări de cercetare importante: Este fezabilă traducerea automată din engleză în română pentru clasificarea sentimentului sau a temelor de discuție? Ce metode de codificare și ce modele ar trebui alese? Îmbunătățește optimizarea hiperparametrilor performanța modelelor? Cum se compară modelele în ceea ce privește acuratețea și viteza de execuție?

Al treilea nostru obiectiv se referă la crearea unor modele de tip Transformer special concepute pentru textele în limba română. Pentru a realiza acest lucru este necesar să identificăm, colectăm și să curățăm un set de date care să conțină un număr semnificativ de tweet-uri în română, neetichetate. Mai precis, dorim să construim de la zero mai multe variante ale modelului RoBERTa, folosind corpusul nostru personalizat, iar aceste variante vor fi denumite BERTweetRO. După procesul de pre-antrenare, vom adapta variantele BERTweetRO pentru sarcinile de SA și TC folosind date traduse, iar apoi vom compara performanța celor mai bune variante față de Multilingual BERT, modele ML clasice și modele ML de tip deep learning.

Ultimul nostru obiectiv este de a evalua și valida performanța SA a celor mai bune modele dezvoltate prin compararea acestora cu Sentimetric[5], o soluție comercială pentru SA în limba română. Pentru a avea o comparație echitabilă, cu implicații practice, vom colecta manual un set mic de date format din tweet-uri reale în limba română, iar cu ajutorul unor voluntari umani vom eticheta fiecare tweet ca fiind negativ, neutru sau pozitiv. Fiecare voluntar va primi instrucțiuni clare privind modul de etichetare, pentru a ne asigura că acest nou set de date poate servi drept un reper de evaluare de încredere.

[5] `sentimetric.ro`

Prin compararea modelelor în fiecare limbă separat vom putea să le ierarhizăm în funcție de performanță, iar prin compararea modelelor antrenate pentru limba engleză cu cele antrenate pentru limba română vom putea evalua în ce măsură modelele se adaptează la texte traduse. În ceea ce privește TC, dorim să evidențiem beneficiile utilizării acestei abordări de invățare supervizată, în contrast cu abordarea TM care este mult mai frecvent utilizată. În plus, noul nostru model BERTweetRO poate contribui la avansarea domeniului procesării limbii române, oferind cercetătorilor posibilitatea de a-l adapta pentru alte sarcini NLP sau de a servi drept ghid pentru pre-antrenarea unor modele de tip BERT personalizate, chiar și în condiții de date limitate.

Capitolul 2

Studiul literaturii de specialitate

SA presupune aplicarea tehnicilor de NLP pentru a măsura emoțiile și conținutul subiectiv din texte, fiind utilizată pe scară largă prentru recenzii de produse, analiza sondajelor, monitorizarea rețelelor sociale și asistența medicală. Aplicațiile sale variază de la business intelligence și analiza feedback-ului clienților până la progrese în cercetarea medicală [27].

Din punct de vedere conceptual, există două abordări principale care pot fi utilizate pentru clasificarea textelor în funcție de sentimentul exprimat. În abordarea bazată pe baze de cunoștințe, cuvinte precum "fericit", "trist", "speriat" sau "plictisit" sunt considerate ca desemnând categorii afective [55]. Cuvintele cu opinie pozitivă sunt folosite pentru a exprima o stare dorită, în timp ce cele negative exprimă o stare nedorită. Unele baze de cunoștințe nu se limitează doar la listarea cuvintelor evidente, ci atribuie și cuvintelor arbitrare o "afinitate" probabilă față de anumite emoții [68]. Listele de cuvinte-opinie sunt, de obicei, create manual, dar pot fi extinse în mod automatizat cu ajutorul dicționarelor prin identificarea de sinonime și antonime [45].

Analiza de sentiment statistică integrează metodele ML, inclusiv modele Bag-of-Words (BoW), Latent Semantic Analysis (LSA), Pointwise Mutual Information for Semantic Orientation, algoritmi de codificarea cuvintelor, etc. Aceste metode oferă rezultate superioare comparativ cu abordările bazate pe cunoștințe, deoarece pot gestiona date textuale mai complexe, însă pentru crearea modelelor ML este nevoie de seturi de date etichetate. Printre algoritmii clasici de ML, alegeri populare [43] sunt Bernoulli Naive Bayes (NB) [42], Support Vector Machines (SVM) [32], Random Forest (RF) [13] sau Logistic Regression (LR) [50]. În ceea ce privește învățarea profundă (DL - *Deep Learning*), toate variantele importante de rețele neuronale precum Deep Neural Network (DNN) [53], Convolutional Neural Network (CNN) [31] sau Long Short-Term Memory (LSTM) [59] sunt raportate ca având performanțe bune în clasificarea textelor.

Metodele clasice de ML și rețelele neuronale standard sunt de obicei aplicate pe reprezentări la nivel de document, precum TF-IDF [66] sau Doc2Vec [37]. Rețelele DL de tip CNN sau LSTM sunt, în general, aplicate pe reprezentări la nivel de cuvânt, cum ar fi Word2Vec [44]. TF-IDF conduce la probleme de dimensionalitate ridicată, motiv pentru care unii autori recomandă aplicarea unor metode de reducere a dimensionalității pentru a îmbunătăți eficiența. În cadrul lucrării noastre vom experimenta cu aceste metode, în căutarea unei combinații potrivite care să ne satisfacă nevoile.

Google a propus modelul Bidirectional Encoder Representations from Transformers (BERT) [18] ca un model pre-antrenat de ultimă generație pentru numeroase sarcini de NLP. Multilingual BERT, de asemenea pre-antrenat pentru limba română, este raportat că funcționează bine în transferul de cunoștințe interlingvistice [62]. Totuși, după cum

indică practica [40], BERT implică costuri semnificative de timp pentru antrenarea și ajustarea fină a modelului, chiar și pe calculatoare performante.

Realizarea SA pe conținutul din social media este considerată o sarcină dificilă [10], deoarece implică gestionarea limbajului neadecvat [23]. Totuși, pentru limbile populare precum engleza, spaniola sau franceza, există numeroase resurse lingvistice care pot fi aplicate pentru a îmbunătăți SA în acest contex. Pentru limba engleză, menționăm BERTweet [51], care a fost ajustat pentru SA și a obținut o acuratețe de 72% pe setul de testare SemEval2017-Task4A [65], depășindu-i pe concurenții RoBERT și XLM-R. Barbieri et al. [9] raportează că BERTweet reprezintă ultima generație pe benchmarkul TweetEval[1], cu un recall mediu de 73%. Pota et al. [63] au aplicat modele bazate pe BERT pentru SA pe date de pe Twitter în limba engleză și italiană, subliniind importanța preprocesării personalizate a textului pentru a extrage informații ascunse — un aspect pe care îl vom lua în considerare în cadrul lucrării noastre.

Pentru limba română, stadiul actual al SA aplicate conținutului din microblogging este mai puțin avansat. În sectorul privat, Technobium[2] a dezvoltat Sentimetric, un serviciu web dedicat analizei de sentiment pentru texte în limba română, care oferă și un demo gratuit online[3]. Din câte cunoaștem, un set de date dedicat conținutului de microblogging în română, similar cu BERTweet, nu este încă disponibil.

LaRoSeDa (Large Romanian Sentiment Dataset) pare să fie singurul set de date disponibil public pentru SA în limba română. Acesta conține 15,000 de recenzii de produse, dintre care 7,500 sunt etichetate ca pozitive și 7,500 ca negative. Din cauza acestei structuri, toate lucrările care folosesc această resursă raportează performanțele obținute pentru SA într-un mod binar. De exemplu, [22] a obținut un scor F1 de 54%, în timp ce lucrarea care a introdus LaRoSeDa a raportat o acuratețe de $\approx$ 91% ca reper [70]. Mai recent, menționăm și corpusul român DistilBERT[4], care ar putea fi utilizat pentru SA binară aplicată pe texte standard. Autorii [5] au raportat o acuratețe de clasificare binară de 98% pentru LaRoSeDa, reprezentând astfel un nou reper de performanță. În ceea ce privește SA multinomială aplicată pe texte din social media în limba română, nu am reușit să identificăm nicio lucrare publicată care să poată fi folosită ca referință de comparație în contextul nostru.

Cu toate acestea, Banea et al. [8] au răspuns afirmativ la întrebarea dacă putem "prezice în mod fiabil subiectivitatea la nivel de propoziție în alte limbi decât engleza, folosindu-ne de un set de date adnotat manual în limba engleză", prin antrenarea unor clasificatori Naive Bayes în 6 limbi, pornind de la un set de date original în engleză cu articole de presă traduse folosind motoare automate. Astfel, acest rezultat motivează eforturile noastre de a folosi traducerea automată pentru a obținere seturile de date de învățare necesare pentru procesarea limbajului natural în limba română.

În prezent, descoperirea subiectelor abstracte de discuție care apar într-o colecție de documente se poate realiza fie prin Clasificarea Tematică (TC), fie prin Modelarea Tematică (TM). TM este un instrument statistic utilizat pe scară largă pentru extragerea variabilelor latente din seturi mari de date, fiind foarte potrivit pentru analiza datelor textuale [72]. Printre cele mai utilizate metode pentru TM se numără Probabilistic Latent Semantic Analysis (PSLA) și Latent Dirichlet Allocation (LDA), care presupun că un document este un amestec de subiecte, unde un subiect este considerat a transmite

[1] `https://huggingface.co/datasets/tweet_eval`

[2] `https://technobium.com/`

[3] `http://sentimetric.ro/`

[4] `https://github.com/racai-ai/Romanian-DistilBERT`

o semnificație semantică printr-un set de cuvinte corelate, reprezentate de obicei ca o distribuție de cuvinte din vocabular. În esență, aceste modele convenționale identifică temele dintr-un corpus textual prin captarea implicită a tiparelor de co-apariție a cuvintelor la nivel de document [74, 12].

Cu toate acestea, aplicarea directă a acestor modele pe texte scurte nu este eficientă datorită tiparelor rare de co-apariție a cuvintelor în documentele individuale [28]. Unele soluții propuse pentru a atenua această problemă presupun agregarea textelor scurte în pseudo-documente mai lungi, însă rezultatele variază în funcție de setul de date utilizat [4]. Modelul Biterm Topic Model (BTM) [15], care modelează componentele tematice folosind perechi de cuvinte neordonate (biterms), tinde să ofere performanțe superioare în cazul textelor scurte, comparativ cu alte abordări.

Principalul avantaj al metodelor TM este că nu necesită date etichetate, ceea ce face ca procesul de colectare a datelor să fie mai accesibil și posibil de realizat într-un mod complet sau parțial automatizat. În ciuda popularității sale, TM este predispus la probleme legate de optimizare, sensibilitate la zgomot și instabilitate, ceea ce poate duce la rezultate nesigure [2]. Unele tehnici nu reușesc să reflecte fidel relațiile din datele reale [11], adesea din cauza unor ipoteze puternice impuse asupra parametrilor cheie. De exemplu, determinarea numărului optim de subiecte este o sarcină dificilă, iar intervenția umană este necesară pentru a atribui etichete relevante subiectelor identificate.

Atunci când sunt disponibile date de antrenare etichetate, TC poate fi folosită pentru identificarea subiectelor, evitând astfel multe dintre limitările TM. În acest caz, algoritmii ML populari tratează identificarea subiectelor ca pe o sarcină standard de clasificare, folosind caracteristici sintactice și lingvistice. Având un set de antrenament $D = X_1, X_2, \ldots, X_N$, unde fiecare înregistrare X_i (document, paragraf, propoziție sau cuvânt) este etichetată cu una dintre k clase tematice, obiectivul este de a antrena modele care să generalizeze din aceste tipare pentru a prezice corect subiectele textelor nevăzute. TC este mai transparentă și mai ușor de interpretat, făcând evaluarea performanței și comparațiile între modele mai directe.

Zeng et al. [77] au propus o abordare hibridă care combină TM cu TC. Mai întâi au extras cele mai relevante caracteristici latente folosind TM, apoi le-au introdus în modele ML supervizate, precum SVM, CNN și LSTM. În experimentele lor au folosit un set de date Twitter furnizat de TREC2011[5], ce conține aproximativ 15,000 de tweet-uri, etichetate semi-automat în 50 de clase tematice. Cea mai mare acuratețe obținută, în jur de $\approx 9.5\%$, a fost atinsă de CNN și este una modestă în cel mai bun caz. Mai mult, autorii au concluzionat că componenta TM nu a îmbunătățit în mod semnificativ capacitățile de învățare ale clasificatorilor.

Spre deosebire de SA, care se concentrează pe polaritatea textului, TC implică adesea un număr mare de clase, care se pot suprapune uneori [25, 39]. Pentru a gestiona această situație, unii autori [26, 52] utilizează acuratețea Top-K în locul celei standard. În loc ca un text să fie clasificat într-o singură clasă, modelul va produce cele mai probabile K clase, iar dacă eticheta corectă se regăsește printre acestea, clasificarea este considerată corectă. În cadrul acestei lucrări, vom ține cont de acest aspect și vom raporta atât acuratețea standard (Top-1), cât și valorile pentru Top-2 și Top-3.

În ceea ce privește TC pentru limba română, putem menționa doar lucrarea lui Vasile et al. [71], care a evaluat capabilitățile unor modele clasice de ML aplicate pe conținut de tip blog. Datele utilizate în studiul lor au fost extrase din 219 bloguri, fiecare instanță fiind etichetată cu o singură temă dintr-un total de 9: "Activism", "Afaceri și Finanțe",

[5] `http://trec.nist.gov/data/tweets`

"Artă", "Călătorii", "Gastronomie", "Literatură", "Modă", "Politică" și "Religie și Spiritualitate". Modelele SMO și Complement Naive Bayes (NB) au obținut cele mai bune rezultate, ambele atingând o acuratețe de aproximativ 77.8%. Un scor ușor mai scăzut de 73.3% a fost obținut de k-NN, în timp ce modelul standard NB a avut cea mai slabă performanță cu un scor de 68.9%. Este important de menționat că autorii au folosit un set de date foarte mic în experimentele lor, ceea ce ridică probleme privind reproductibilitatea acestor rezultate pe seturi de evaluare mai mari.

Nu am identificat alte lucrări de cercetare relevante care să vizeze în mod direct conținutul rețelelor sociale în limba română, iar seturile de date etichetate lipsesc, ceea ce înseamnă că va trebui să traducem un set de date potrivit din limba engleză pentru a crea datele de antrenament necesare experimentelor noastre de clasificare tematică.

Capitolul 3

Analiza sentimentelor pe texte românești din rețele sociale

Am căutat pe mai multe platforme online, inclusiv baze de date academice și depozite NLP (precum Kaggle), dar nu am reușit să găsim un set de date care să corespundă cerințelor noastre [49]. Prin urmare, am decis să folosim un set de date open-source în limba engleză, să îl traducem în limba română cu ajutorul Google Translate[1] și să îl utilizăm ca o resursă "surogat" în experimentele noastre.

Pentru această cercetare, am selectat setul de date Twitter US Airline Sentiment Tweets[2]. Datele au fost colectate în anul 2015, iar fiecare tweet a fost etichetat manual de către colaboratori externi cu polaritatea globală a sentimentului (pozitiv, negativ, sau neutru). Setul conține aproximativ 15,000 de tweeturi, cu următoarea distribuție a claselor: 63% negativ, 21% neutru și 16% pozitiv. Fiecare tweet este însoțit și de încrederea contributorului în eticheta atribuită, iar în cazul tweeturilor negative este inclus și motivul pentru evaluarea realizată.

Integritatea structurală, gramaticală și sintactică a oricărui text tradus automat este afectată într-o anumită măsură. Principala metrică utilizată în literatura de specialitate pentru a evalua calitatea unui traducător automat este scorul BLEU (Bilingual Evaluation Understudy) [58]. Acest scor variază între 0 și 100, unde valorile mai mari indică o traducere de calitate mai bună (100 reprezentând o traducere perfectă). În [3], texte generale în limba engleză au fost traduse în 50 de limbi diferite folosind Google Translate, iar scorul BLEU a fost utilizat ca metrica de evaluare. Scorul mediu obținut pentru toate traducerile a fost de aproximativ 76. Pentru traducerea din engleză în română s-a înregistrat un scor de 84, care ce este considerabil peste medie. Scorul maxim, de 91, a fost obținut pentru traducerea în portugheză, iar cel minim, de 55, pentru traducerea în hindi. Rezultate similare sunt raportate și în [67], unde traducerea din engleză în română a obținut din nou performanțe peste medie. Aceste constatări susțin faptul că abordarea noastră de a folosi traducerea pentru a crea un set de date destinat antrenării modelelor ML are șanse ridicate de succes.

Pentru experimentele noastre, setul de date a fost împărțit în seturi de antrenare și testare folosind o împărțire standard de 75–25%: ≈11,000 de instanțe pentru antrenare și ≈3,700 pentru testare, menținând distribuția claselor similară în ambele seturi. Mai mult, datele de antrenare și testare în limba engleză și română sunt identice din punct de vedere al conținutului, în sensul că includ același set de instanțe.

[1] `https://translate.google.com/`

[2] `https://www.kaggle.com/crowdflower/twitter-airline-sentiment`

În continuare, am dezvoltat un modul personalizat de preprocesare, care conține următorii pași, aplicați în această ordine specifică:

1. Eliminarea spațiilor albe redundante (independent de limbă).
2. Lematizarea și tokenizarea personalizată a cuvintelor (dependentă de limbă).
3. Identificarea și eliminarea adreselor URL (independent de limbă).
4. Identificarea și înlocuirea emoji-urilor (independent de limbă).
5. Identificarea și eliminarea mențiunilor din rețelele sociale (independent de limbă).
6. Eliminarea caracterelor consecutive redundante (independent de limbă).
7. Înlocuirea abrevierilor (dependentă de limbă).
8. Eliminarea cuvintelor de tip stop-word (dependentă de limbă).
9. Conversia tuturor caracterelor la litere mici (independent de limbă).
10. Eliminarea semnelor de punctuație (independent de limbă).

Pașii independenți de limbă pot fi aplicați în același mod atât în engleză, cât și în română. În contrast, pașii dependenți de limbă implică faptul că este necesară o cunoaștere specifică a limbii române sau engleze.

În pasul 1, toate spațiile consecutive care apar de mai mult de două ori sunt eliminate din tweet-uri, de exemplu "Hello world!" devine "Hello world!".

În pasul 2, am folosit SpaCy[3] pentru tokenizarea și lematizarea cuvintelor, datorită acurateței ridicate atât în engleză, cât și în română. Input-ul pentru acest pas este format din șiruri de caractere, iar output-ul generat este o listă de tokeni, unde fiecare token este fie un număr, cuvânt lematizat, sau simbol. Am ales lematizarea în detrimentul stemming-ului deoarece lematizarea poate identifica în mod corect partea de vorbire și sensul intenționat al cuvintelor.

Am modificat funcționalitatea implicită de tokenizare și lematizare oferită de SpaCy pentru a putea trata texte specifice rețelelor sociale, precum:

- Am instruit SpaCy să nu lematizeze cuvintele specifice rețelelor sociale de tagging (hashtag-uri și mențiuni) și să nu le separe în timpul pasului de tokenizare. În mod implicit, SpaCy ar transforma o intrare precum "#working" în următoarea listă de tokeni ["#", "work"]. Prin urmare, ne asigurăm că aceste cuvinte specifice rețelelor sociale sunt păstrate intacte și că tokenii cu hashtag sunt gestionați corespunzător.
- Cuvintele negate sunt esențiale pentru analiza de sentiment [25], așadar am implementat un mecanism care poate identifica cuvintele negate într-o propoziție și le atașează un prefix și un sufix special. De exemplu, "not happy" este transformat în următorii tokeni ["not", "|!|happy|!|"]. Cu această funcționalitate ne asigurăm că elementele de negație nu sunt pierdute după pasul de eliminare a cuvintelor de legătură (stop-words), întrucât listele de stop-word conțin, de obicei, cuvinte de negație precum "no", "not".

[3] `https://spacy.io/`

În pasul 3, identificăm care dintre tokeni sunt URL-uri (Uniform Resource Locator) folosind un tipar complex de expresii regulate cu peste 400 de caractere și le eliminăm din date, deoarece URL-urile nu sunt în limbaj natural și nu oferă informații sau perspective utile pentru clasificarea textului.

Pasul 4 este esențial pentru contextul nostru de microblogging, deoarece ne ocupăm de emoji-uri, un fel de "limbaj impropriu" care a devenit extrem de popular la nivel mondial în sursele textuale informale. Ca exemplu, Instagram a raportat în 2015 că aproape jumătate din textele de pe platforma lor conțineau emoji-uri [4]. Kralj Novak et al. [36] au raportat că aproximativ 4% din tweet-uri conțin emoji-uri, iar polaritatea sentimentului acestora nu depinde de limbă. În acest sens, ei au construit lexiconul *Emoji Sentiment Ranking* [5] care conține 751 dintre cele mai frecvent utilizate emoji-uri, fiecare fiind adnotat cu polaritatea sa de sentiment (negativă, neutră, sau pozitivă).

Prin urmare, în acest pas verificăm dacă un token se regăsește în *Full Emoji List* a Unicode[6] și dacă are o polaritate asociată în lexiconul Emoji Sentiment Ranking. Dacă da, tokenul va fi înlocuit cu polaritatea sa, împreună cu un prefix și un sufix special. De exemplu, "U0001F642" reprezintă codul Unicode pentru "față ușor zâmbitoare" și are polaritatea "pozitiv". Astfel, va fi transformat în "|^|positive|^|". Dacă un token este un emoji dar nu a fost etichetat cu o polaritate în lexicon, sau dacă codul Unicode nu reprezintă un emoji, atunci vom adăuga un alt prefix și sufix. De exemplu, "U00001D19" este codul Unicode pentru "R majuscul întors" și va fi transformat în "|*|U00001D19|*|".

La pasul 5, sunt identificate și eliminate mențiunile specifice rețelelor sociale, care fac referire la alte entități din cadrul rețelei. În cazul Twitter, căutăm simbolul "@" care este utilizat pentru a crea legături externe în tweeturi. Majoritatea mențiunilor fac referire la companii aeriene americane și am observat o corelație puternică între acestea și sentimentul general exprimat în tweet-uri. Prin urmare, am decis să eliminăm mențiunile deoarece existența lor ar putea crește artificial acuratețea clasificatorilor noștri (dorim să evaluăm polaritatea pe baza limbajului și nu pe baza unor entități denumite specifice).

Pasul 6 se ocupă de caracterele consecutive excesive. În textele microblogging, indiferent de limbă, este obișnuit să se accentueze un cuvânt prin adăugarea de caractere suplimentare. De exemplu, cuvântul "cool" poate fi scris cu un număr variat de litere "o". Pentru a rezolva această problemă, caracterele consecutive care apar de mai mult de 3 ori într-un anumit token sunt eliminate. Astfel, restricționăm apariția unui cuvânt fie la forma sa standard, fie la o singură instanță de scriere accentuată: de exemplu, "cool" poate apărea doar ca "cool" sau "coool", a doua fiind instanța accentuată.

În pasul 7, înlocuim unele abrevieri din text cu forma lor completă, folosindu-ne de două lexicoane construite special în acest scop. Versiunea în limba engleză conține aproximativ 400 de abrevieri, în timp ce versiunea în limba română conține în jur de 100 de abrevieri și a fost construită pe baza paginii Wikipedia pentru abrevieri românești[7]. După acest proces, fiecare abreviere este înlocuită cu tokenii derivați din forma sa completă. De exemplu, "brb" va fi înlocuit cu o listă de 3 tokeni: ["be", "right", "back"].

La pasul 8, cuvintele de tip stop-word sunt identificate și eliminate folosind dicționarele oferite de SpaCy, atât pentru limba engleză cât și pentru română. Eliminarea cuvintelor stop-word este o practică standard în procesarea textului, așa cum este indicat și în [39].

[4] https://instagram-engineering.com/emojineering-part-1-machine-learning-for-emoji-trendsmachine-learning-for-emoji-trends-7f5f9cb979ad

[5] `https://kt.ijs.si/data/Emoji_sentiment_ranking/`

[6] `https://unicode.org/emoji/charts/full-emoji-list.html`

[7] `https://ro.wiktionary.org/wiki/Wik%C8%9Bionar:Abrevieri`

La pasul 9, toți tokenii sunt transformați în litere mici, reducând astfel numărul de tokeni identificați pentru un anumit concept.

În pasul 10, eliminăm toate semnele de punctuație suplimentare din interiorul tokenilor, cu excepția celor marcați cu prefixele și sufixele speciale create de către noi.

Tabelul 3.1 prezintă două exemple de tweet-uri și reprezentarea acestora după aplicarea tuturor celor 10 pași din modulul nostru de preprocesare. Primul tweet este în limba engleză, iar al doilea este traducerea sa automată în limba română. "U00001F620" este codul Unicode asociat emoji-ului "angry face".

Language	Raw tweet	Preprocessed tweet
EN	"@united and don't hope for me having a nicer flight next time. REALY getting on my nerves U00001F620 #nothappy...."	["|!|hope|!|", "i", "nice", "flight", "next", "time", "really", "nerve", "|^|negative|^|", "|#|nothappy|#|"]
RO	"@unitate și sa nu sperați să am un zbor mai frumos data viitoare. Devine într-adevăr pe nervii mei U00001F620 #nothappy...."	["|!|sperat|!|", "zbura", "frumos", "data", "viitor", "deveni", "adevăr", "nerv", "|^|negative|^|", "|#|nothappy|#|"]

Table 3.1: Exemplu de preprocesare a unui tweet

Elementul esențial pentru orice sarcină de NLP este reprezentarea internă a documentului, adică selecția adecvată a caracteristicilor din textul brut și codificarea acestora în valori numerice, astfel încât reprezentarea să fie gestionabilă sau să fie îmbogățită cu semnificații lingvistice. Acest proces este obligatoriu deoarece algoritmii de analiză a textului necesită intrări numerice pentru a efectua calcule matematice. Datele textuale în forma lor brută conțin de asemenea multe caracteristici irelevante sau redundante, care trebuie tratate în această etapă deoarece modelele ML nu sunt foarte eficiente în a le gestiona în mod autonom [69].

Pentru studiul nostru am selectat cele mai populare abordări, așa cum sunt sugerate de literatura de specialitate în domeniul NLP [25, 39]: TF-IDF, Word2Vec și Doc2Vec.

Am antrenat un vectorizator TF-IDF pe setul de antrenament în limba engleză și un altul pe traducerea acestuia în limba română. TF-IDF a fost aplicat pe tweet-urile preprocesate, iar vocabularul a fost setat să conțină doar tokenii care apar de cel puțin 3 ori. Astfel, am eliminat un număr mare de tokeni infrecvenți sau care ar fi putut fi construiți eronat în etapa de preprocesare. Vectorizatoarele TF-IDF antrenate au fost apoi aplicate pe seturile de testare, după care am observat că vocabularul englezesc conținea aproximativ 3,100 de tokeni, iar cel românesc conținea aproximativ 4,000 de tokeni, datorită faptului că limba română este în general mai abundentă în cuvinte decât limba engleză.

Am utilizat biblioteca Gensim [64] pentru a învăța vectorii de embedding Word2Vec și Doc2Vec pentru datele noastre. La fel ca în cazul TF-IDF, a fost necesar să antrenăm două modele separate, unul pe setul de antrenament în limba engleză și altul pe setul de antrenament în limba română. Pentru Word2Vec am selectat arhitectura CBOW deoarece funcționează mai bine pe texte scurte. Pentru Doc2Vec am utilizat modelul DBOW împreună cu o arhitectură de tip hierarchical softmax. Această combinație permite prezicerea cuvintelor în contextul lor și îmbunătățește timpul de antrenare, ceea ce este ideal în cazul nostru. Vocabularul a fost, de asemenea, setat să includă doar tokenii care apar de cel puțin 3 ori.

Pentru ambele modele, Word2Vec și Doc2Vec, am setat dimensiunea vectorilor de embedding la 200, pentru a ne asigura că modelele pot capta suficientă informație contextuală și, în același timp, să menținem o performanță computațională eficientă. Fiecare

model a fost antrenat cu o rată de învățare (α) de 0.025, o dimensiune a ferestrei de 5 și pe parcursul a 5 epoci.

Reducerea dimensionalității în știința datelor se referă la procesul prin care numărul caracteristicilor sau dimensiunea lor este redusă, astfel încât informația disponibilă să păstreze proprietățile semnificative ale datelor originale [1]. Această abordare este utilă în unele sarcini NLP unde caracteristicile inițiale au o dimensionalitate extrem de ridicată, cum este cazul lui TF-IDF care e cunoscut pentru generarea de matrici mari și dispersate.

Odată cu aplicarea reducerii dimensionalității, ne propunem să reducem dimensiunea datelor ca apoi să analizăm în ce măsură performanța predictivă și computațională a modelelor ML este afectată. Am selectat următorii algoritmi pentru testare: Principal Component Analysis (PCA) [33], Non-negative Matrix Factorization (NMF) [60] și Latent Semantic Analysis (LSA) [20].

Am aplicat toți cei 3 algoritmi pe seturile de date TF-IDF și am redus numărul de caracteristici la 500. Aceasta înseamnă că reprezentarea redusă pentru limba engleză este de aproximativ 6.2 ori mai mică, iar pentru limba română de aproximativ 8 ori mai mică decât cea originală. Nu am aplicat reducerea dimensionalității asupra datelor Word2Vec și Doc2Vec deoarece dimensiunea dorită a vectorilor (200) a fost setată înainte de extragerea caracteristicilor. De asemenea, reducerea dimensiunii acestor reprezentări ar putea compromite semnificativ calitatea lor.

Așa cum este indicat în literatura de specialitate [25, 35, 39], pentru construirea modelelor SA pot fi aplicați algoritmi clasici de învățare automată (classic ML), metode de învățare profundă (deep learning) sau abordări bazate pe modele lingvistice moderne. Pentru experimentele noastre am decis să aplicăm următoarele metode de învățare:

- Classic ML:
  - Bernoulli Naive Bayes (Bernoulli NB)
  - Support Vector Machine with a linear kernel (Linear SVM)
  - Random Forest (RF)
  - Logistic Regression (LR)
- Deep Learning:
  - Deep Neural Network (DNN)
  - Long Short-Term Memory (LSTM)
  - Convolutional Neural Network (CNN)
- Advanced Language Model:
  - Multilingual BERT

Bernoulli NB, SVM, RF, LR și DNN vor fi asociate cu TF-IDF, TF-IDF cu dimensiune redusă și Doc2Vec. LSTM și CNN vor fi aplicate pe codificarea Word2Vec, deoarece acestea pot procesa și sunt specializate în date secvențiale multidimensionale. În acest caz, se încorporează un strat suplimentar de embeddings sub stratul de intrare, pentru a mapa fiecare token din text la reprezentarea sa Word2Vec corespunzătoare. Deoarece pentru limba română nu dispunem de o resursă globală de embeddings precum GloVe [61] care este disponibilă pentru limba engleză, am optat să învățăm reprezentările Word2Vec de la zero pentru ambele limbi, utilizând exclusiv seturile de antrenament.

Am implementat algoritmii clasici ML cu ajutorul bibliotecii *Scikit-Learn*[8], în timp ce pentru algoritmii deep learning am utilizat biblioteca *Keras*[9].

[8] `https://scikit-learn.org/stable/`

[9] `https://keras.io/`

Pentru BERT, am folosit modelul disponibil în cadrul bibliotecii Hugging Face Transformers[10], apelat cu varianta de bază multilingual uncased. Ceea ce diferențiază acest model de ceilalți clasificatori este faptul că utilizează propriul său mecanism de codificare, denumit Multilingual Tokenizer, și nu acceptă caracteristici sub formă de TF-IDF, Word2Vec sau Doc2Vec. Peste M-BERT am adăugat un strat ascuns de tip dens cu 75 de noduri și funcția de activare ReLU, urmat de stratul standard de clasificare cu 3 noduri, care produce clasa de sentiment. Optimizatorul selectat a fost Adam, cu o rată de învățare de 2×10^{-5} și $\epsilon = 10^{-8}$. Funcția de pierdere a fost setată la Categorical CrossEntropy.

Algoritmii evolutivi (EA – *Evolutionary Algorithm*) reprezintă o familie de metode de optimizare inspirate din selecția naturală. Aceștia funcționează prin îmbunătățirea iterativă a unui set de soluții candidate, utilizând o funcție de fitness ca metrică de evaluare. Procesele de selecție, încrucișare, mutație și reproducere sunt aplicate asupra candidaților (numiți și "indivizi") pentru a-i evolua de-a lungul mai multor generații, cu scopul final de a găsi cei mai buni indivizi conform funcției de fitness alese [73]. Algoritmul Genetic (GA – *Genetic Algorithm*) este cel mai popular și de bază tip de EA.

Am selectat această metodă probabilistică deoarece poate accelera considerabil procesul de optimizare a hiperparametrilor, oferind în același timp o combinație bună de valori pentru parametri. Există variante mai complexe ale algoritmului GA, cum ar fi GA termodinamic, însă am decis să folosim o versiune standard de GA, întrucât s-a demonstrat că aceasta depășește oricum optimizarea bayesiană [46]. Un alt avantaj important al optimizării evolutive este că funcționează în toate cele 3 tipuri de spații de căutare (continuu, discret și categorial), indiferent de clasificatorul asupra căruia se aplică optimizarea.

Am folosit biblioteca Sklearn-Genetic-Opt[11] pentru implementarea optimizării GA în raport cu clasificatorii selectați. Sklearn-Genetic-Opt utilizează framework-ul DEAP[12], care oferă numeroase variante de EA necesari pentru rezolvarea problemelor de optimizare.

GA-ul a fost proiectat după cum urmează. Având un număr N de parametri de optimizat, un individ/cromozom este reprezentat ca un vector $(p_1, p_2, \ldots, p_i, \ldots, p_N)$. În acest vector, fiecare p_i reprezintă valoarea selectată pentru hiperparametrul corespunzător N_i. O populație alcătuită din 20 de indivizi este evoluată pe parcursul a 40 de generații, cu o probabilitate de încrucișare de 80%, pentru a combina caracteristicile indivizilor, și o probabilitate de mutație de 10%, pentru a introduce variație în populație. Indivizii sunt selectați pentru generația următoare folosind un turneu elitist standard de dimensiune 3. Intern, fiecare individ este evaluat folosind acuratețea ca funcție de fitness, calculată prin validare încrucișată cu 3 fold-uri. În general, convergența se observă după 15–20 de generații, astfel încât evoluarea populației pe parcursul a 40 de generații este mai mult decât suficientă pentru a garanta o selecție eficientă a parametrilor.

În cazul algoritmilor ML clasici toți parametrii descriși în documentația oficială Sklearn au fost optimizați. În cazul rețelei DNN am luat în considerare, printre parametri, următoarele aspecte: capacitatea rețelei (numărul de straturi ascunse și numărul de unități per strat), funcția de activare, funcția de regularizare și rata de drop-out. Pentru CNN și LSTM am dorit să aplicăm aceeași logică, însă a apărut o problemă neașteptată. În cazul clasificării textelor, aceste două modele necesită ca parametrul "embedding weight" să fie sub forma unui tensor 2D, dar Sklearn-genetic-opt nu acceptă acest tip de date pentru parametrii. Ca urmare, a trebuit să modificăm codului sursă al bibliotecii pentru a putea transmite parametrii multidimensionali direct către DEAP.

[10] `https://huggingface.co/docs/transformers/en/model_doc/bert`

[11] `https://sklearn-genetic-opt.readthedocs.io/en/stable/api/gasearchcv.html`

[12] `https://deap.readthedocs.io/en/master/`

Pentru clasificatorul BERT, având în vedere că antrenarea unui singur model este foarte costisitoare din punct de vedere al timpului, am decis să nu aplicăm procedura de optimizare evolutivă. În loc de validare încrucișată, am rezervat 10% din setul de antrenare pentru validare și am lăsat procesul de învățare să optimizeze funcția de pierdere pe parcursul a câtorva epoci. Am observat că modelul supraînvață (overfits) rapid, astfel că am oprit procesul de antrenare după 2 epoci, atât pentru limba engleză cât și pentru limba română.

În continuare prezentăm arhitectura la nivel înalt a sistemului nostru de Analiză a Sentimentelor (SA), sumarizată în Figura 3.1.

În partea superioară a diagramei, traducerea automată a tweet-urilor din setul Twitter US Airline Sentiment din limba engleză în limba română este evidențiată ca primul și cel mai important pas necesar pentru a putea rula SA în limbi diferite, folosind un singur set de date unilingv drept sursă de informație.

Etapa de preprocesare constă într-o serie de proceduri variate, grupate în două fluxuri abstracte distincte. Fluxul din stânga generează texte compatibile cu tehnicile TF-IDF, Doc2Vec și Word2Vec. În fluxul din dreapta se aplică doar pașii 3 și 5 din modulul de preprocesare, împreună cu o tokenizare la nivel de propoziție, pentru a genera texte în formatul așteptat de encoderul BERT preantrenat.

Variantele TF-IDF și Doc2Vec sunt grupate împreună pentru a evidenția faptul că output-ul tuturor acestor metode are aceeași formă. Mai precis, un text preprocesat este transformat într-un vector de lungime N, pe când Word2Vec va transforma un text preprocesat într-o matrice NxM, unde numărul de rânduri este egal cu numărul de cuvinte din text, iar numărul de coloane este egal cu dimensiunea vectorului de embedding. Codificarea contextuală BERT va reprezenta textele prin intermediul mai multor vectori, cu ajutorul Multilingual Tokenizer.

În etapa de antrenare și ajustare a modelelor, se observă că pentru toate metodele ML selectate, cu excepția M-BERT, facem optimizarea hiperparametrilor cu algoritmi genetici, pentru a identifica cel mai bun set de parametri. Am ales algoritmii genetici în locul unei căutări simplistice de tip grilă (grid search) pentru a evita riscul de a cădea într-un minim local și pentru că acești algoritmii genetici sunt capabili să exploreze eficient spațiul de căutare chiar și pentru parametri cu valori continue. Din cauza timpilor de antrenare ridicați în cazul BERT, am optat pentru un proces clasic de antrenare folosind parametrii recomandați.

Bernoulli NB, Linear SVM, LR, RF și DNN sunt grupați împreună pentru a evidenția relația de tip mulți-la-mulți dintre acest grup de algoritmi și seturile de caracteristici derivate din TF-IDF și Doc2Vec. Acest lucru înseamnă că orice tip de caracteristici din acest grup poate fi utilizat de oricare dintre modelele menționate anterior. LSTM și CNN sunt grupați pentru a evidenția faptul că ambii folosesc caracteristicile Word2Vec, în timp ce clasificatorul BERT utilizează codificările specifice generate de propriul său Multilingual Tokenizer.

În partea de jos a diagramei este evidențiat obiectivul principal al lucrării noastre, și anume clasificarea textelor de intrare în 3 categorii: negativ, neutru sau pozitiv.

În continuare, prezentăm experimentele realizate și discutăm rezultatele obținute. Clasificatorii vor fi evaluați strict pe seturile de testare folosind 3 metrici frecvent utilizate în literatură: Macro F1, Weighted F1 și Acuratețe. În plus, vom prezenta și analiza timpii de execuție pentru fiecare model de învățare automată. Ca urmare a acestor rezultate, alți cercetători pot selecta mai ușor modelul care se potrivește cel mai bine nevoilor lor, în funcție de performanța predictivă așteptată și de resursele computaționale disponibile.

Figure 3.1: Arhitectura sistemului de Analiză a Sentimentelor

Toate experimentele, cu excepția ajustărilor fine pentru Multilingual BERT, au fost făcute pe un calculator de înaltă performanță cu următoarele specificații: 2 × Intel Xeon Gold 6230 CPUs, 128 GB RAM DDR4 și 8 × NVIDIA Tesla V100 GPUs cu 32 GB VRAM fiecare. Pentru BERT am utilizat un mediu de dezvoltare echipat cu TPU-uri (Tensor Processing Units), puse la dispoziție de Google. Nu am rulat BERT pe sistemul nostru performant deoarece, în urma unor investigații, am descoperit că putem obține timpi de execuție mai rapizi pe TPU-uri. Codul sursă a fost implementat în Python 3.9.

Am aplicat fluxul de procesare descris în Figura 3.1 pe setul de date original și pe cel tradus automat al tweet-urilor din Twitter US Airline Sentiment. Am utilizat exact aceeași metodologie pentru ambele limbi, atât pentru consistență, dar mai ales pentru a izola impactul traducerii automate de alte variabile.

Tabelul 3.2 prezintă performanțele de învățare ale clasificatorilor, iar pentru algoritmii care au fost asociați cu reprezentarea TF-IDF sunt oferite și rezultatele obținute după reducerea dimensionalității.

Encoding	Classifier	English original dataset			Romanian translated dataset		
		Acc.	Weighted F1	Macro F1	Acc.	Weighted F1	Macro F1
TFIDF	Bernoulli NB	77.37	77.25	70.7	78.2	78.2	71.91
	Linear SVM	77.41	76.67	69.77	78.36	77.47	70.54
	RF	77.45	75.9	68.47	77.81	76.45	69.04
	LR	66.7	55.21	39.2	65.2	54.71	38.17
	DNN	78.18	77.15	70.14	77.2	76.23	69.19
TFIDF+PCA	Bernoulli NB	68.01	62.82	50.74	67.78	62.49	50.4
	Linear SVM	75.89	75.94	69.52	76.2	74.71	66.94
	RF	75.78	74.79	67.28	75.73	73.47	65.05
	LR	67.52	58.95	45.39	63.66	57.16	42.17
	DNN	76.34	75.34	68.43	76.58	75.75	68.59
TFIDF+NMF	Bernoulli NB	72.44	71.7	63.43	72.93	72.22	64.43
	Linear SVM	74.14	74.13	67.1	73.16	69.88	60.5
	RF	74.77	73.27	65.89	74.99	73.88	66.05
	LR	62.55	48.01	25.62	65.41	55.17	39
	DNN	74.67	74.28	66.66	75.02	73.07	64.7
TFIDF+LSA	Bernoulli NB	68.26	63.97	52.23	67.01	60.77	47.43
	Linear SVM	76.3	75.05	67.31	76.9	75.65	68.18
	RF	76.12	74.29	66.33	76	74.3	66.23
	LR	67.81	68.72	61.72	64.1	57.87	43.15
	DNN	76.67	75.89	69.06	76.36	75.07	67.67
Word2Vec	CNN	78.21	76.66	70.33	77.69	76	68.67
	LSTM	77.5	76.35	69.4	78.17	77.98	71.39
Doc2Vec	Bernoulli NB	62.44	48.01	25.62	62.42	47.98	25.62
	Linear SVM	63.05	48.01	25.62	62.67	47.97	25.62
	RF	62.75	48	25.22	62.52	47.97	25.62
	LR	62.44	47.9	25.02	62.42	47.79	25.62
	DNN	62.9	48.01	25.62	62.44	47.98	25.62
Multilingual BERT Tokenizer	Multilingual BERT	83.02	82.57	77.48	80.99	80.5	74.81

Table 3.2: Performanța predictivă a clasificatorilor

Unul dintre cele mai importante aspecte evidențiate în Tabelul 3.2 este că performanța clasificării între limbi este surprinzător de consistentă. De exemplu, Bernoulli NB cu TF-IDF arată o ușoară creștere a acurateței, de la 77.37% în limba engleză la 78.2% în limba română. Linear SVM și DNN înregistrează, de asemenea, variații minore între cele două

limbi, sugerând că abordarea prin traducere automată este viabilă și pentru limba română. Variația în rândul tuturor combinațiilor de modele și scheme de codificare se situează în jurul valorii de ±2% în ceea ce privește acuratețea, iar această diferență nesemnificativă este menținută și pentru metricile Weighted și Macro F1.

Multilingual BERT stabilește un nou standard de performanță, atingând o acuratețe de 83% pe setul de date în limba engleză și 81% pe cel în limba română. În cazul algoritmilor clasici ML, Bernoulli NB și Linear SVM au obținut cele mai bune rezultate pe toate cele 3 metrici de evaluare, atingând o acuratețe de aproximativ 78% în ambele limbi. RF are o acuratețe comparabilă, însă înregistrează o scădere mai mare a performanței în raport cu Weighted și Macro F1. LR a fost considerabil cel mai slab din acest grup, indicând o incapacitate de a capta informațiile necesare pentru o clasificare corectă.

DNN, CNN și LSTM au performat similar cu Bernoulli NB și Linear SVM, având acurateți de ≈ 78% pe setul în engleză și între 77–78% pe cel în română. DNN cu TF-IDF și CNN cu Word2Vec au avut rezultate puțin mai bune pe setul englez, în timp ce LSTM cu Word2Vec a fost puțin mai bun pe cel românesc, dar diferențele sunt neglijabile.

Un alt aspect important pe care vrem să îl subliniem aici este impactul reducerii dimensionalității asupra vectorilor TF-IDF: metodele PCA, NMF și LSA au redus performanța de clasificare a modelelor în comparație cu aceleași modele care au utilizat vectorii originali. De exemplu, acuratețea Linear SVM pe setul în română a scăzut de la 78.36% cu TF-IDF la 76.2% cu TF-IDF+PCA, 73.16% cu TF-IDF+NMF și 76.9% cu TF-IDF+LSA. Un trend similar se observă și pe setul englez. Aceasta sugerează că, deși acești algoritmii pot îmbunătăți viteza de execuție, ei afectează negativ acuratețea prin pierdere de informație. Totuși, impactul nu este atât de drastic, ceea ce poate face ca un astfel de compromis să fie acceptabil în funcție de contextul aplicației.

Cele mai slabe performanțe au fost înregistrate de modelele care au folosit Doc2Vec ca mecanism de codificare. Spre deosebire de TF-IDF, care se bazează pe frecvența cuvintelor, și Word2Vec, care construiește un vector dedicat pentru fiecare cuvânt în funcție de context, Doc2Vec încearcă să genereze un singur vector dens pentru a reprezenta fiecare instanță din setul de date. Această tehnică poate funcționa bine în cazul textelor mai lungi, însă este problematică în cazul nostru, deoarece cu cât textele sunt mai scurte, cu atât riscul de a introduce erori în vectorii generați este mai mare. Acest lucru explică, cel mai probabil, performanța slabă a Doc2Vec. În orice caz, niciun model care a utilizat Doc2Vec nu a obținut rezultate predictive acceptabile pentru aplicații din lumea reală.

Prin această analiză observăm că, deși traducerea din engleză în română aduce anumite variații, performanțele tuturor clasificatorilor selectați au rămas stabile între limbi. Acest fapt confirmă că abordarea bazată pe traducere automată poate fi folosită pentru a crea resurse pentru SA în limba română.

În tabelul 3.3 sunt afișați timpii de execuție pentru optimizarea hiperparametrilor, antrenarea modelelor finale și testarea acestora. Cu ajutorul optimizării evolutive am reușit să creștem măsura F1 ponderată a modelelor cu 1–3%, obținând câștiguri ușor mai mari pentru acuratețe.

Ca primă observație, dorim să evidențiem timpii mai mari de optimizare și antrenare care au fost necesari pe setul românesc de date. Acest lucru nu este surprinzător, deoarece limba română tinde să fie mai expresivă decât engleza, ceea ce înseamnă că se folosesc un număr mai mare de cuvinte pentru a exprima aceleași idei. Din acest motiv, modelele pentru limba română au ajuns să aibă un vocabular mai extins decât modelele antrenate pe setul de date în limba engleză. Totuși, această creștere a timpilor de învățare pentru limba română nu este suficient de mare pentru a fi considerată un obstacol real în practică.

Encoding	Classifier	English original dataset			Romanian translated dataset		
		Opt. (s)	Train (s)	Test (s)	Opt. (s)	Train (s)	Test (s)
TFIDF	Bernoulli NB	1337	0.285	0.128	1645	0.368	0.147
	Linear SVM	920	0.36	0.02	1048	0.238	0.022
	RF	89603	8.502	0.024	3588	7.855	0.026
	LR	5735	1.59	0.085	6158	0.568	0.122
	DNN	11513	2.176	0.274	13551	2.23	0.44
TFIDF+PCA	Bernoulli NB	395	0.051	0.03	387	0.064	0.031
	Linear SVM	2450	2.089	0.016	2038	2.418	0.01
	RF	2113	14.044	0.02	2245	4.792	0.022
	LR	795	1.587	0.134	375	0.295	0.05
	DNN	10195	1.561	0.19	3558	1.015	0.166
TFIDF+NMF	Bernoulli NB	384	0.058	0.044	362	0.051	0.017
	Linear SVM	482	0.276	0.015	412	0.149	0.013
	RF	226	1.885	0.014	5125	16.644	0.019
	LR	567	0.479	0.077	449	0.404	0.03
	DNN	17842	5.109	0.207	9236	5.28	0.176
TFIDF+LSA	Bernoulli NB	389	0.064	0.016	385	0.062	0.032
	Linear SVM	8417	6.53	0.003	2724	2.353	0.009
	RF	2245	0.743	0.028	204	0.693	0.022
	LR	943	1.546	0.14	659	0.905	0.154
	DNN	10195	1.782	0.214	6585	2.567	0.261
Word2Vec	CNN	9143	1.46	0.281	16127	4.209	0.25
	LSTM	17172	16.865	1.32	62364	20.926	1.292
Doc2Vec	Bernoulli NB	271	0.028	0.01	274	0.021	0.015
	Linear SVM	654	0.19	0.013	580	0.152	0.014
	RF	484	1.595	0.015	428	0.936	0.01
	LR	712	1.171	0.14	263	0.523	0.111
	DNN	17842	0.912	0.144	11529	4.594	0.279
Multilingual BERT Tokenizer	Multilingual BERT	N/A	416.13	16.64	N/A	444.02	16.73

Table 3.3: Timpii de optimizare a hiperparametrilor, antrenare și evaluare ale clasificatorilor

Un alt aspect ce poate fi observat se referă la variațiile mari ale vitezelor de optimizare și antrenare între clasificatori și metodele de codificare. Este evident că antrenarea modelelor clasice este mai rapidă decât a modelelor deep learning, deoarece acestea din urmă sunt prin natura lor mai complexe. Chiar și așa, timpii de antrenare a modelelor finale sunt nesemnificativi, cu excepția modelului Multilingual BERT care a necesitat 7 minute pentru fiecare limbă. Al doilea cel mai lent, dar considerabil mai rapid, este LSTM cu 17 secunde pentru engleză și 21 de secunde pentru română. Cel mai rapid a fost Bernoulli NB, necesitând mai puțin de 1 secundă pentru toate combinațiile de limbă și metodă de codificare.

Problema apare în momentul căutării parametrilor optimi deoarece acest proces este foarte consumator de timp: complexitatea modelului conduce la un număr mai mare de parametri, un spațiu de căutare mai extins și timpi de antrenare mai lungi; factori care determină o creștere exponențială a duratei de optimizare. Pentru Bernoulli NB, fără reducerea dimensionalității, acest proces a durat aproximativ 27 de minute în limba română și 22 de minute în limba engleză. Linear SVM a avut timpi similari între cele două limbi, puțin peste 16 minute. Căutarea celor mai buni parametri și a capacității rețelei DNN a durat aproximativ 3 ore și 45 de minute pentru datele în limba română și 3 ore și 11 minute pentru cele în limba engleză. Pentru LSTM, acest proces a durat peste

17 ore pentru română și aproape 5 ore pentru engleză, în ciuda faptului că dimensiunea vectorului de embedding a fost doar 200. Dintre modelele clasice de ML, RF și LR au fost cele mai lente.

Timpii de testare sunt relativ scăzuți pentru toate perechile de modele și codificări ceea ce arată că, odată antrenați, clasificatorii pot furniza rapid un număr mare de predicții, indiferent de limbă. Multilingual BERT, în ciuda timpului său îndelungat de antrenare, a obținut timpi buni de evaluare de 16.64 și 16.73 secunde pentru seturile de date în limba engleză și română. Al doilea cel mai lent a fost LSTM, dar acesta a avut nevoie doar de aproximativ 1.3 secunde pentru a finaliza testarea. Celelalte modele (inclusiv CNN) au rezultate și mai bune, cu timpi de testare sub 0.5 secunde în ambele limbi.

Per ansamblu, acești timpi de execuție oferă informații utile despre cât de solicitanți sunt algoritmii ML selectați. Un model complex precum BERT are capacități predictive superioare, însă rulează mai lent, în timp ce modelele mai simple, precum Bernoulli NB sau Linear SVM, oferă un compromis rezonabil între viteză și acurateţe, ceea ce le face ideale în situațiile în care resursele hardware sunt limitate.

Capitolul 4

Clasificarea tematică pe texte românești din rețele sociale

Pentru clasificarea tematică avem nevoie de un set de date extins care să conțină texte specifice rețelelor sociale. Numărul de subiecte disponibile în setul de date ar trebui să acopere teme generale, dar relevante, iar eticheta tematică a fiecărei instanțe ar trebui să fie atribuită corect, de preferat prin adnotare manuală. Din păcate, nu am identificat niciun set de date public disponibil care să satisfacă toate cerințele cercetării noastre și nu dispunem de resursele necesare pentru a colecta și adnota unul nou de la zero. Prin urmare, am decis să traducem un set de date open-source din limba engleză în română cu traducere automată și să îl folosim ca "surogat" în experimentele noastre [48].

Am selectat setul de date News Category Dataset[1], care conține 202,372 de titluri de știri colectate între anii 2012 și 2018 de pe site-ul HuffPost[2]. Acest site oferă știri, satiră, bloguri, conținut original și acoperă o varietate de subiecte. Fiecare înregistrare a setului de date conține următoarele atribute: *category* (41 de categorii), *headline*, *short_description*, *authors*, *date* (data publicării) și *link* (URL-ul articolului).

Există o serie de motive pentru care am ales acest set de date ca reper pentru experimentele noastre: (i) Conține texte scurte, similare cu cele întâlnite pe platformele de social media; (ii) Tematicile sunt destul de generale, iar numărul acestora este suficient de mare; (iii) Categoria fiecărui articol a fost adnotată manual; (iv) Setul de date este suficient de mare pentru a permite antrenarea eficientă a modelelor de învățare automată; (v) A fost colectat relativ recent.

Pentru problema noastră de clasificare, ne vom concentra doar pe atributele "titlu" și "descriere scurtă" ale setului de date, ignorând autorii și data publicării. Prin urmare, am fuzionat atributele "titlu" și "descriere scurtă" pentru a crea un atribut nou numit *text_merged*. Marea majoritate a textelor fuzionate conțin între 94 și 254 de caractere, media fiind de ≈174 și deviația standard de aproape 80 de caractere. Acest lucru dovedește că textele generate au caracteristicile textelor scurte similare cu cele prezente pe platformele de socializare (de exemplu, un tweet de pe Twitter este limitat la 280 de caractere, un comentariu de pe TikTok este limitat la 150 de caractere).

În continuare, am realizat o investigație preliminară a datelor și am identificat unele probleme legate de distribuția și granularitatea celor 41 de etichete de clasă originale, așa cum este ilustrat în Figura 4.1. Primele trei cele mai populare clase sunt: "POLITICS" cu aproximativ 16% din înregistrări, "WELLNESS" cu aproximativ 9%, și "ENTER-

[1] `https://www.kaggle.com/datasets/rmisra/news-category-dataset`

[2] `https://www.huffpost.com/`

TAINMENT" cu aproximativ 8% din total. Cele mai puțin frecvente patru clase sunt: "COLLEGE", "LATINO VOICES", "CULTURE & ARTS" și "EDUCATION", fiecare reprezentând doar 0.5% din înregistrări, ceea ce indică un dezechilibru semnificativ între clase în cadrul datelor.

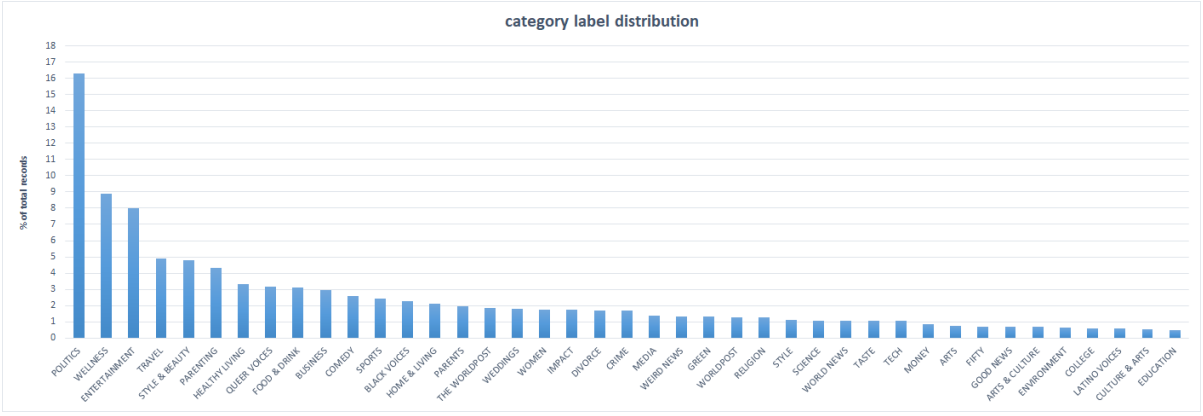

Figure 4.1: Distribuția originală a categoriilor tematice, 41 de clase

Am observat că există încă două probleme legate de tematicile existente: un subset dintre ele se suprapun, iar altele sunt mult prea granulare. De exemplu, categoriile "SCIENCE" și "TECH" sunt prea specifice, dar pot fi grupate în mod natural într-o clasă comună, precum "SCIENCE & TECH", în timp ce alte clase au etichete diferite, dar denotă același lucru, cum ar fi "ARTS & CULTURE" și "CULTURE & ARTS".

Așadar, pentru a îmbunătăți calitatea datelor, am decis să grupăm categoriile excesiv de granulare și cele sinonime. Prin urmare am transformat următoarele clase după cum urmează: "HEALTHY LIVING" a fost reetichetată ca făcând parte din clasa existentă "WELLNESS"; "PARENTS" a fost reetichetată ca "PARENTING"; "STYLE" a fost reetichetată ca "STYLE & BEAUTY"; "GREEN" a fost reetichetată ca "ENVIRONMENT"; "TASTE" a fost reetichetată ca "FOOD & DRINK"; "COLLEGE" a fost reetichetată ca "EDUCATION"; "THE WORLDPOST" și "WORDPOST" au fost reetichetate ca "WORLD NEWS"; "ARTS" și "CULTURE & ARTS" au fost reetichetate ca "ARTS & CULTURE"; "BUSINESS" și "MONEY" au fost reetichetate într-o nouă clasă denumită "BUSINESS & FINANCES"; "SCIENCE" și "TECH" au fost reetichetate într-o nouă clasă denumită "SCIENCE & TECH"; "QUEER VOICES", "BLACK VOICES" și "LATINO VOICES" au fost reetichetate într-o nouă clasă denumită "GROUPS VOICES"; "FIFTY" și "GOOD NEWS" au fost reetichetate într-o nouă clasă denumită "MISCELLANEOUS".

La finalul acestui proces, setul de date ajustat conține 26 de teme care sunt cu adevărat distincte, iar nicio clasă nu are mai puțin de 1% din etichetele înregistrărilor, ceea ce înseamnă că cea mai puțin populară clasă are peste 2,000 de instanțe. Acest lucru ar trebui să contribuie la creșterea performanței modelelor care vor fi antrenate ulterior și, în același timp, să asigure consistență și coerență în sarcina de clasificare tematică. Noul atribut de clasă a fost denumit *category_merged*, iar distribuția completă a acestuia este ilustrată în Figura 4.2.

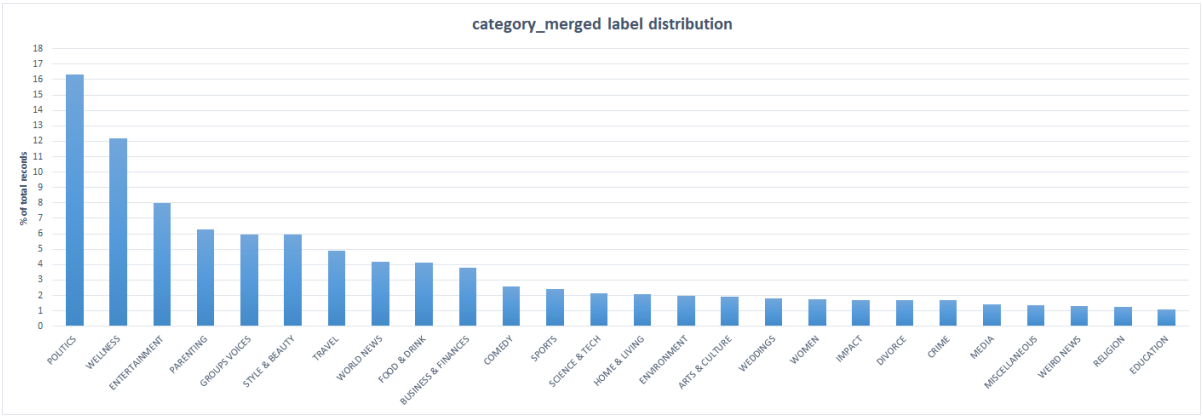

Figure 4.2: Distribuția categoriilor tematice comasate, 26 de clase

Pentru experimentele noastre, setul de date a fost împărțit în seturi de antrenare și de testare folosind o împărțire standard de 75–25%: ≈151,500 de instanțe pentru antrenare și ≈50,500 pentru testare, cu distribuții similare ale claselor în ambele seturi. Mai mult, datele de antrenare și testare în limba engleză și în limba română sunt identice în sensul că ele conțin același set de instanțe.

Pentru a elimina zgomotul natural existent în setul nostru de date, am utilizat următorii 5 pași de preprocesare, aplicați în această ordine specifică:

1. Eliminarea spațiilor albe suplimentare (independent de limbă).
2. Lematizarea și tokenizarea cuvintelor (dependent de limbă).
3. Eliminarea cuvintelor de legătură (dependent de limbă).
4. Transformarea tuturor literelor în minuscule (independent de limbă).
5. Eliminarea semnelor de punctuație (independent de limbă).

Dacă predicțiile sunt realizate pe texte reale din social media, atunci este necesar un număr suplimentar de pași de preprocesare pentru a le aduce într-un format mai apropiat de cel existent în setul de antrenament. Deoarece textele din setul de date News Category nu prezintă aceste caracteristici, nu vom intra în detalii suplimentare, însă informații adiționale despre acești pași de procesare pot fi găsite în Capitolul 3.

Pentru acest studiu, am selectat două dintre cele mai populare abordări, așa cum sunt sugerate de literatura de specialitate în NLP [25, 39]: TF-IDF și Word2Vec.

Nu vom include Doc2Vec ca metodă de extragere a caracteristicilor din două motive. În primul rând, clasificatorii care au utilizat acest tip de reprezentare au avut, de departe, cele mai slabe performanțe predictive în experimentele de analiză a sentimentului, așa cum este prezentat în Capitolul 3. În al doilea rând, setul de date pentru clasificarea tematică este considerabil mai mare decât cel utilizat pentru analiza sentimentului, conținând aproximativ 202,000 de instanțe comparativ cu 15,000. Aceasta echivalează cu o creștere de 13.5 ori a volumului de date, ceea ce ar conduce la timpi de execuție mult mai mari pentru optimizarea hiperparametrilor, antrenarea și testarea modelelor.

Am antrenat un vectorizator TF-IDF pe setul de antrenament original în limba engleză și un altul pe traducerea în limba română. TF-IDF a fost aplicat pe listele de tokeni preprocesați, iar vocabularul a fost configurat să conțină doar tokenii care apar de cel puțin 5 ori. Acest lucru a fost făcut pentru a elimina un număr mare de tokeni care

sunt foarte rar utilizați sau care ar fi putut fi generați eronat în etapa de preprocesare. În continuare, modelele TF-IDF nou create au fost aplicate pe seturile de testare, iar ulterior am observat că vocabularul în limba engleză conține aproximativ 25,000 de tokeni, în timp ce vocabularul în limba română conține 27,500 de tokeni. Ne așteptam la această mică diferență, deoarece limba română este mai detaliată decât limba engleză.

Datorită naturii dispersate a TF-IDF, a numărului mare de instanțe de antrenament și a dimensiunii vocabularului, vectorii antrenați sunt stocați și utilizați în formatul Compressed Sparse Row (CSR). Formatul CSR este avantajos pentru gestionarea matricelor dispersate deoarece comprimă eficient stocarea elementelor nenule, reducând semnificativ utilizarea memoriei și îmbunătățind performanța în timpul operațiilor de înmulțire a matricilor/vectorilor [24].

Pentru a învăța reprezentările Word2Vec, am utilizat biblioteca Gensim [64] și, la fel ca în cazul TF-IDF, a fost necesar să creăm două modele dedicate, câte unul pentru fiecare limbă. Am selectat arhitectura Continuous Bag-Of-Word (CBOW), deoarece oferă rezultate mai bune pentru texte scurte, iar vocabularul a fost setat să includă doar tokenii care apar de cel puțin 5 ori. Dimensiunea vectorului de încorporare a tokenilor a fost stabilită la 300, pentru a echilibra înțelegerea contextuală și eficiența în execuție. Modelele Word2Vec au fost antrenate cu următorii parametri: rată de învățare (α) de 0.025, o fereastră de context de 5, pe parcursul a 5 epoci.

Am decis să nu reducem dimensiunile vectorilor TF-IDF deoarece, în urma experimentelor realizate în cadrul SA, am constatat că algoritmi precum LSA, NMF sau PCA pot într-adevăr să îmbunătățească viteza de execuție, însă reduc și capacitatea predictivă a modelelor. Având în vedere importanța menținerii unei acurateți ridicate pentru clasificarea tematică, acest compromis de performanță nu este justificabil, mai ales în contextul în care avem un număr mult mai mare de clase.

Pentru experimentele noastre am selectat următoarele metode de învățare:

- ML clasic:
  - Bernoulli Naive Bayes (Bernoulli NB)
  - Support Vector Machine with a linear kernel (Linear SVM)
  - Random Forest (RF)
- Deep Learning:
  - Long Short-Term Memory (LSTM)
  - Convolutional Neural Network (CNN)
- Model Lingvistic Avansat:
  - Multilingual BERT

Am decis să nu includem LR și DNN aici deoarece aceste modele au fost utilizate în studiul SA și nu au obținut rezultate remarcabile. LSTM și CNN vor fi aplicate pe caracteristicile Word2Vec datorită capacității lor de a procesa date secvențiale, în timp ce algoritmii clasici de învățare vor fi aplicați pe TFIDF. Pentru a rula LSTM și CNN pe date secvențiate, a trebuit să introducem un strat suplimentar de embedding imediat după stratul de intrare în rețea. Acest strat de embedding mapează fiecare token dintr-o instanță la reprezentarea Word2Vec corespunzătoare, fiind echipat cu embedding-urile de cuvinte generate în timpul extragerii caracteristicilor.

Pentru BERT, am utilizat varianta de bază multilingvă "uncased" (fără majuscule) disponibilă în biblioteca Hugging Face transformers[3]. Peste M-BERT am adăugat un

[3] `https://huggingface.co/docs/transformers/en/model_doc/bert`

strat dens ascuns cu 128 de neuroni și funcția de activare ReLU, urmat de un strat de clasificare standard cu 26 de neuroni care produce clasa tematică. Optimizatorul ales a fost Adam, cu o rată de învățare de 2×10^{-5} și $\epsilon = 10^{-8}$. Funcția de pierdere a fost setată la Categorical CrossEntropy.

Algoritmul nostru de optimizare evolutivă pentru TC a fost proiectat după cum urmează. Având un număr N de hiperparametri de optimizat, un individ/cromozom este notat ca un vector $(p_1, p_2, \ldots, p_i, \ldots, p_N)$. În acest vector, fiecare p_i reprezintă valoarea selectată pentru hiperparametrul corespunzător N_i. O populație de 10 indivizi, fiecare reprezentând o posibilă combinație de hiperparametri, este evoluată pe parcursul a 20 de generații, cu o probabilitate de încrucișare de 80% pentru a combina caracteristicile indivizilor și o probabilitate de mutație de 10% pentru a introduce variație în populație. Indivizii sunt selectați pentru următoarea generație folosind un turneu elitist standard de dimensiune 3. Fiecare individ este evaluat utilizând acuratețea ca funcție de fitness, calculată prin validare încrucișată cu 3 fold-uri. În general, convergența se poate observa după 10–15 generații, astfel că evoluarea populației de-a lungul a 20 de generații este suficientă pentru a garanta o selecție bună a hiperparametrilor.

În cazul învățării automate clasice, toți parametrii descriși în documentația oficială Sklearn au fost optimizați. Pentru LSTM și CNN, am luat în considerare printre parametri următorii: capacitatea rețelei (numărul și dimensiunea straturilor ascunse), funcția de activare, funcția de regularizare și rata de drop-out. Pentru M-BERT, deoarece antrenarea unui singur model este foarte lentă, am omis efectuarea procedurii de optimizare a hiperparametrilor. În loc de validarea încrucișată, am rezervat 10% din setul de antrenament pentru validare și am lăsat modelul să optimizeze funcția de pierdere pe parcursul mai multor epoci. Am observat că, pentru ambele limbi, acuratețea maximă pe setul de validare este atinsă după 5 epoci, astfel că am oprit procesul de antrenament în acest punct.

În continuare, prezentăm arhitectura la nivel înalt a sistemului nostru de Clasificare Tematică. Figura 4.3 sintetizează toate procedurile și pașii implicați.

În partea de sus a diagramei este prezentată traducerea automată a dataset-ului News Category din limba engleză în limba română, ca primul pas esențial pentru a permite TC în diferite limbi, utilizând un singur set de date uni-lingual ca sursă de informații.

Pasul de preprocesare constă în diverse proceduri, grupate în două fluxuri abstracte diferite. Cel din stânga va genera texte potrivite pentru tehnicile TF-IDF și Word2Vec. În cel din dreapta se aplică doar o tokenizare la nivel de propoziție pentru a genera texte conform așteptărilor codificatorului BERT pre-antrenat.

Pasul de extragere a caracteristicilor evidențiază transformarea textului preprocesat în caracteristici numerice care pot fi folosite pentru antrenarea modelelor de învățare automată. În cazul TF-IDF, un text preprocesat este transformat într-un vector de lungime N, în timp ce Word2Vec convertește același text preprocesat într-o matrice $N \times M$, unde numărul de rânduri este egal cu numărul de cuvinte din text, iar numărul de coloane este egal cu dimensiunea embedding-ului pentru cuvinte. Codificarea contextuală BERT va reprezenta textele folosind mai mulți vectori, cu ajutorul tokenizer-ului multilingv.

Bernoulli NB, Linear SVM și RF sunt grupate împreună pentru a evidenția relația de tip mulți-la-unu dintre acest grup și caracteristicile TF-IDF. LSTM și CNN sunt grupate împreună pentru a sublinia faptul că ambele utilizează caracteristicile Word2Vec, în timp ce clasificatorul BERT folosește codificările specifice generate de propriul său Tokenizer Multilingv.

În partea de jos a diagramei este evidențiat scopul principal al studiului nostru, și anume clasificarea textelor de intrare în 26 de subiecte tematice de discuție. Lista com-

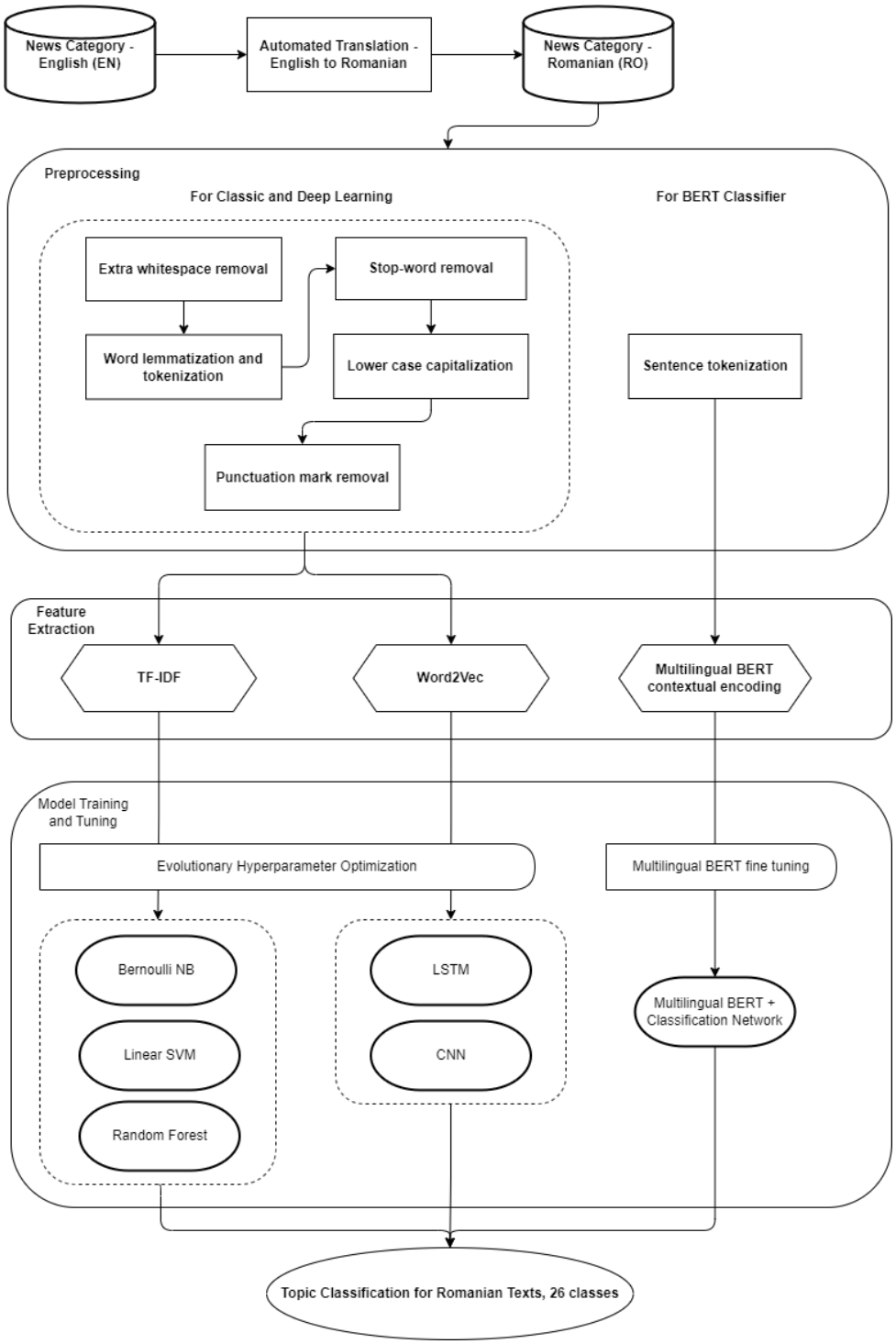

Figure 4.3: Arhitectura sistemului de Clasificare Tematică

pletă a tuturor subiectelor, ordonată de la cel mai frecvent la cel mai puțin frecvent, este: ”politics”, ”wellness”, ”entertainment”, ”parenting”, ”groups voices”, ”style & beauty”, ”travel”, ”world news”, ”food & drinks”, ”business & finances”, ”comedy”, ”sports”, ”science & tech”, ”home & living”, ”environment”, ”arts & culture”, ”weddings”, ”women”, ”impact”, ”divorce”, ”crime”, ”media”, ”miscellaneous”, ”weird news”, ”religion”, ”education”.

În continuare, prezentăm experimentele noastre și discutăm implicațiile acestora. Modelele vor fi evaluate strict pe seturile de testare utilizând măsura standard a acurateței

(Top-1), precum și Top-2 și Top-3. Măsura acurateței Top-K este foarte utilă având în vedere numărul mare de clase tematice și potențialul de suprapunere a subiectelor în cadrul textelor [26, 52]. În plus, vom prezenta și analiza timpii de execuție pentru fiecare model de învățare automată. Toate experimentele au fost realizate pe același hardware descris în Capitolul 3.

Am aplicat fluxul de procesare descris în Figura 4.3 pe setul de date News Category, folosind exact aceeași metodologie pentru ambele limbi, atât pentru consistență, dar mai ales pentru a izola impactul traducerii automate de alte variabile.

Unul dintre cele mai importante observații din Tabelul 4.1 este că performanța predictivă a Multilingual BERT, Bernoulli NB și Linear SVM între cele două limbi este surprinzător de consistentă, variațiile fiind de doar aproximativ 2–3% pentru toate valorile Top-K considerate. CNN a înregistrat o scădere mai vizibilă pe setul de date românesc, cu diminuări de aproximativ 4% pentru Top-1 și Top-2 și 5% pentru Top-3, ceea ce denotă o sensibilitate moderată la traducerea lingvistică. Pe de altă parte, LSTM a avut cea mai mare scădere, cu diminuări de 14% pentru Top-1, 15% pentru Top-2 și 13% pentru Top-3, evidențiind faptul că eficacitatea sa a fost serios afectată de procesul de traducere. RF a înregistrat o scădere similară de aproximativ 13–14% pentru toate metricile, indicând o adaptabilitate limitată pe datele traduse.

Encoding	Classifier	English original dataset			Romanian translated dataset		
		Top-1	Top-2	Top-3	Top-1	Top-2	Top-3
TFIDF	Bernoulli NB	64.17	78.93	85.27	62.80	77.70	84.11
	Linear SVM	67.97	81.75	87.10	66.73	80.05	85.30
	RF	30.0	40.74	50.21	16.56	28.89	37.0
Word2Vec	CNN	66.15	80.07	85.24	61.66	74.05	79.28
	LSTM	67.64	80.41	85.55	53.50	65.59	72.39
Multilingual BERT Tokenizer	Multilingual BERT	74.85	87.29	91.73	72.63	85.56	90.25

Table 4.1: Performanța predictivă a clasificatorilor

Multilingual BERT obține rezultate de ultimă generație, cu o acuratețe de 74.85% pentru Top-1, 87.29% pentru Top-2 și 91.73% pentru Top-3 pe setul de date în limba engleză, respectiv 72.63% pentru Top-1, 85.56% pentru Top-2 și 90.25% pentru Top-3 pe setul de date în limba română.

În cazul învățării automate clasice, Linear SVM a obținut cele mai bune rezultate și ocupă locul al doilea la nivel general, după Multilingual BERT. Acesta a menținut o consistență ridicată a performanței predictive între engleză și română, atingând aproximativ 68%, 82% și 87% pe setul de date în limba engleză și 67%, 80% și 85% pe cel în limba română. Bernoulli NB este al doilea cel mai bun model din această categorie și al treilea la nivel general, cu scoruri Top-K cu 2–3% mai mici decât Linear SVM pe setul de date în limba engleză. Pe setul de date în limba română, aceeași diferență de 2–3% se păstrează pentru Top-2 și Top-3, însă pentru Top-1 se observă o scădere mai mare, de aproximativ 4%, ceea ce indică că Linear SVM este de preferat atunci când tema principală contează cel mai mult în analiză.

RF a avut de departe cea mai slabă performanță, cu o acuratețe de doar 30%, 40% și 50% în limba engleză și 17%, 29% și 37% în română. Pentru a pune lucrurile în perspectivă, acest lucru înseamnă că chiar și acuratețea Top-3 este considerabil mai mică decât

acuratețea Top-1 a lui Bernoulli NB, ceea ce este surprinzător, deoarece RF-urile au fost utilizate cu succes în NLP pentru multe sarcini de clasificare. Chiar și în experimentele noastre de analiză a sentimentului, RF a funcționat bine pentru ambele limbi, fiind comparabil cu cei mai buni algoritmi clasici, însă rezultatele obținute aici arată clar că ceva nu a funcționat cum trebuie. Cea mai mare diferență față de SA este că, în acest caz, avem un număr mult mai mare de clase țintă (26 comparativ cu 3). RF împarte datele folosind arbori de decizie pentru a maximiza câștigul informațional, însă un număr mare de clase poate reduce "puritatea" diviziunilor, deoarece este mai dificil să se găsească suficiente caracteristici lingvistice care să distingă clar între subiecte abstracte de discuție.

CNN a avut inițial performanțe superioare față de Bernoulli NB în limba engleză, cu acurateță de 66% Top-1, 80% Top-2 și 85% Top-3, însă a înregistrat o scădere mai semnificativă în limba română, performând cu 2–4% mai slab decât Bernoulli NB, reducându-i scorurile la 62%, 74% și 79%, respectiv. LSTM a avut rezultate similare cu CNN pe textele în engleză, chiar depășind CNN cu aproape 2% la predicția Top-1, dar, așa cum s-a menționat anterior, a înregistrat cea mai mare scădere pe textele în română, atingând doar 54% Top-1, 66% Top-2 și 72% Top-3.

Cu această analiză observam că, deși traducerea din engleză în română implică anumite variații și modificări, performanțele majorității clasificatorilor supuși acestui tip de experiment rămân stabile. Acest lucru confirmă faptul că o abordare bazată pe traducere automată poate fi utilizată pentru a crea resursele necesare pentru TC în limba română.

În Tabelul 4.2 prezentăm timpii de execuție pentru optimizarea hiperparametrilor, antrenarea modelelor finale și testarea acestora. Cu ajutorul optimizării evolutive am reușit să creștem acuratețea Top-1 a modelelor cu 2–5%.

Encoding	Classifier	English original dataset			Romanian translated dataset		
		Opt. (s)	Train (s)	Test (s)	Opt. (s)	Train (s)	Test (s)
TFIDF	Bernoulli NB	443	0.567	0.035	389	0.59	0.04
	Linear SVM	8005	25.798	0.034	11803	45.91	0.042
	RF	2992	14.593	0.1	845	0.6	0.183
Word2Vec	CNN	37317	46.493	1.58	36797	56.98	1.65
	LSTM	286062	130.19	9.5	63605	119.1	6.16
Multilingual BERT Tokenizer	Multilingual BERT	N/A	7420	157	N/A	7498	157

Table 4.2: Timpii de optimizare a hiperparametrilor, antrenare și evaluare ale clasificatorilor

În primul rând, putem observa că timpii pentru optimizarea hiperparametrilor variază mult în funcție de limbă și clasificator. Comparativ cu experimentele SA, unde majoritatea modelelor au avut timpi de antrenare și optimizare mai lenți pentru limba română, aici putem vedea că situația este mai echilibrată. Motivul pentru care 4 din cele 5 modele au timpi de optimizare mai buni pe textele în limba română este, cel mai probabil, datorat unei convergențe mai rapide a algoritmilor în etapa de optimizare a hiperparametrilor. Acest lucru este în concordanță cu timpii de antrenare, deoarece în acest caz doar 2 modele, și anume LSTM și RF, au fost mai rapide pe datele în limba română.

Optimizarea lui Linear SVM a fost substanțial mai lentă decât în cazul celorlalte metode clasice de ML: pe setul de date în limba engleză a durat ≈2.2 ore, iar pe cel în limba română aproximativ ≈3.3 ore. Aceste perioade extinse pot fi atribuite numărului mare de parametri din familia de modele SVM, ceea ce înseamnă că au fost necesare

mai multe iterații evolutive pentru a atinge performanțe optime. Un alt factor care ar fi putut crește și mai mult timpii pentru limba română este numărul mai mare de tokeni din vocabular.

RF a avut un timp de optimizare mult mai scurt pentru limba română, durând doar ≈14 minute, comparativ cu ≈50 de minute pentru engleză. Acești timpi scăzuți, împreună cu performanța predictivă slabă în ambele limbi, sugerează că procesul de optimizare a hiperparametrilor nu a reușit să găsească un set bun de parametri, ratând atât optimurile globale cât și pe cele locale. În schimb, Bernoulli NB a fost de departe cel mai rapid de optimizat: ≈7.4 minute pentru engleză și ≈6.5 minute pentru română.

Similar optimizării hiperparametrilor, timpii de antrenare diferă de la model la model, Bernoulli NB fiind extrem de rapid (0.567 secunde pentru engleză și 0.59 secunde pentru română) datorită implementării sale simple. Linear SVM este considerabil mai solicitant din punct de vedere computațional, dar a reușit să finalizeze antrenarea într-un interval de timp rezonabil, necesitând 25.79 secunde pentru engleză și 45.91 secunde pentru română. RF prezintă din nou performanțe mixte, cu timpi de antrenare de 14.593 secunde în engleză și doar 0.7 secunde în română.

Așa cum era de așteptat, modelele Deep Learning au avut timpi de execuție mult mai mari în comparație cu algoritmii clasici, atât în cazul optimizării hiperparametrilor, cât și al antrenării. CNN a necesitat aproximativ ≈10.4 ore pentru optimizare pe setul de date în engleză și ≈10.2 ore pe cel în română, cu timpi de antrenare de 46.49 secunde, respectiv 56.98 secunde. LSTM a fost chiar mai lent, optimizarea sa durând incredibil de mult, aproximativ ≈79.5 ore pentru engleză și un timp mai realist de ≈17.7 ore pentru română. Timpul semnificativ mai scurt pentru setul de date în română, asociat cu o scădere considerabilă a performanței predictive pentru această limbă, ne sugerează că procesul de optimizare a hiperparametrilor ar fi găsit un optim local, și nu unul global. În schimb, timpii de antrenare de 130.19 secunde pentru engleză și 119.1 secunde pentru română pot fi considerați acceptabili.

Deși mai ridicate decât ale celorlalte modele, timpii de execuție ai Multilingual BERT sunt foarte similari între cele două limbi, antrenarea durând aproximativ ≈2.1 ore, iar testarea 157 secunde pentru fiecare limbă. Această consistență este în concordanță cu arhitectura BERT, care este concepută să gestioneze date multilingve, însă cu un cost computațional mai ridicat. Absența optimizării hiperparametrilor este compensată de preantrenarea lui BERT pe volume mari de date diverse, permițându-ne să obținem rezultate predictive superioare cu parametri standard.

Timpii de testare sunt relativ scăzuți pentru toate perechile de modele și codificări, având în vedere dimensiunea generoasă a setului de test, care conține aproximativ 51,000 de instanțe. Acest lucru indică faptul că, odată antrenați, clasificatorii pot furniza rapid un număr semnificativ de predicții, indiferent de limbă. Multilingual BERT, în ciuda antrenării lente, are timpi de evaluare mult mai acceptabili. Al doilea cel mai lent a fost LSTM, cu 9.5 secunde pentru engleză și 6.16 secunde pentru română. CNN a obținut timpi mai buni, sub 1.7 secunde pentru ambele seturi de date, în timp ce modelele clasice de ML au fost cele mai rapide, toate terminând în mai puțin de 0.2 secunde.

Per ansamblu, acești timpi de execuție oferă informații utile despre cerințele algoritmilor selectați. Un model complex precum BERT are capacități predictive superioare, dar rulează mai lent, în timp ce modele mai simple precum Linear SVM sau Bernoulli NB oferă un compromis rezonabil între viteză și acuratețe, făcându-le ideale în medii cu resurse computaționale limitate.

Capitolul 5

BERTweetRO: Modele lingvistice pentru texte românești din rețele sociale

În acest capitol prezentăm eforturile noastre de pre-antrenare de la zero a 8 modele BERTweetRO, bazate pe arhitectura RoBERTa, cu ajutorul unui corpus în limba română ce conține tweet-uri publice. Pentru a evalua modelele noastre, le ajustăm (fine-tune) pentru Analiza Sentimentelor (cu 3 clase de polaritate) și Clasificare Tematica (cu 26 de clase), și le comparăm cu Multilingual BERT, precum și cu o serie de alte modele ML clasice și deep learning [47].

Twitter Stream[1], colectat de către Archive Team, este un corpus public valoros care oferă un volum uriaș de texte extrase de pe Twitter și stocate în format JSON. Acesta acoperă toți anii începând cu 2012 până la mijlocul anului 2021, fiind împărțit în 2,900 de fișiere ce însumează ≈6.8 TB de date. Numărul exact de tweet-uri din acest set de date nu este specificat, însă luând în considerare intervalul lung de timp acoperit, precum și dimensiunea documentelor, putem afirma, cu un grad ridicat de certitudine, că Twitter Stream ar trebui să satisfacă o gamă largă de obiective. Cercetătorii, instituțiile private sau publice pot folosi aceste date pentru a analiza subiectele de actualitate, sentimentele publice, evenimentele culturale sau sociale, și multe altele, fie în timp real, fie retrospectiv, pentru a răspunde întrebărilor legate de dinamica societăților moderne.

Spre deosebire de alte resurse, aceasta nu conține doar tweet-uri în limbi internaționale populare, deoarece, în procesul de web-scraping, majoritatea postărilor publice au fost colectate indiferent de limba în care au fost scrise. Astfel, vom folosi Twitter Stream pentru preantrenarea modelelor BERTweetRO, întrucât acesta surprinde evoluția limbii române și modul în care ea este utilizată într-un context de microblogging. Având în vedere dimensiunea întregii arhive de tweet-uri, hardware-ul nostru limitat impune necesitatea unei selecții de date pentru a putea antrena mai multe versiuni de modele RoBERTa într-un interval de timp rezonabil. În acest sens, am decis să folosim doar un subset din Twitter Stream, care cuprinde aproximativ 800 GB de date corespunzătoare unei perioade de un an: iulie 2020 – iunie 2021.

Un alt factor care a condus la această decizie are legătură cu procesele de fine-tuning ce urmează a fi realizate pe noile modele RoBERTa, și anume Analiza Sentimentelor (SA) și Clasificarea Subiectelor (TC). Prin recunoașterea acestei constrângeri, ne propunem să stabilim un Proof of Concept (POC) care demonstrează fezabilitatea antrenării modelelor

[1] `https://archive.org/details/twitterstream`

bazate pe BERT pe texte din social media în limba română, utilizând un set de date relativ redus. Acest lucru va duce, cel mai probabil, la performanțe mai scăzute comparativ cu utilizarea întregii arhive, însă obiectivul nostru principal este de a arăta că este posibil pentru cercetători să creeze modele decente chiar și în situațiile în care există limitări semnificative de hardware sau timp. În iterațiile viitoare, dacă vom avea acces la mai multe resurse computaționale, sperăm să includem datele rămase în procesul de pre-antrenare pentru a dezvolta modele și mai performante.

Ca prim pas, am descărcat datele corespunzătoare perioadei menționate anterior, după care am efectuat o inspecție manuală pentru a ne familiariza cu structura și natura acestora. Documentele JSON conțin două tipuri diferite de instanțe: un tip indică ștergerea conținutului de pe platformă și include ID-ul ștergerii, alături de alte metadate, dar fără informații textuale utile. Celălalt tip, denumit "post", conține o cantitate mare de informații, însă pentru studiul nostru sunt relevante câmpul "text", care reprezintă mesajul tweet-ului, și câmpul "lang", care indică limba în care este scris mesajul.

În continuare, am selectat și analizat în detaliu 200 de postări aleatorii dintr-o perioadă de 2 luni și am descoperit o problemă majoră legată de câmpul "lang": un număr semnificativ de tweet-uri erau etichetate ca fiind în limba română, deși în realitate nu erau. Multe dintre acestea erau pur și simplu clasificate greșit, în unele cazuri extreme fiind etichetate ca aparținând unei limbi foarte diferite, precum malaieza, iar altele erau pur și simplu "zgomot", adică postări care conțin doar o combinație de mențiuni pe Twitter, hashtag-uri, URL-uri și emoji-uri. Acest lucru evidențiază problemele care pot apărea atunci când se lucrează cu conținut generat de utilizatori în mediul online, unde tonul informal al comunicării, greșelile gramaticale și alte neregularități afectează degradează acuratețea instrumentelor automate de identificare a limbii.

În urma acestei investigații inițiale, am decis să folosim Python[2] împreună cu langid[3] pentru a identifica corect limba postărilor. Am ales langid deoarece a fost antrenat pe un număr mare de limbi (în prezent suportând un total de 97), ceea ce îl face o alegere potrivită pentru setul nostru de date multilingv, și oferă timpi de procesare foarte rapizi împreuna cu rezultate de ultimă generație. Un alt avantaj al langid este faptul că oferă un scor de "nivel de încredere" pentru fiecare predicție, care acționează ca o măsură a fiabilității. Am rulat langid pe același subset de 200 de tweet-uri folosind o abordare cu prag ridicat, în care considerăm textele ca fiind în limba română doar dacă nivelul de încredere depășește 95%, pentru a evita includerea de rezultate fals pozitive. Am făcut o nouă revizuire a clasificării limbii și am constatat că majoritatea textelor au fost etichetate corect de această dată.

Performanța pe textele brute a fost satisfăcătoare, însă pentru a îmbunătăți rezultatele am implementat un modul de preprocesare care include identificarea și eliminarea automată a mențiunilor de pe Twitter, hashtag-urilor, link-urilor URL și emoticoanelor. Prin acest mecanism vrem să oferim lui langid texte mai "curate" și mai standardizate, în speranța creșterii acurateței. Am rulat din nou langid, de această dată pe datele curățate, și am realizat o nouă rundă de investigații. Rezultatele au fost vizibil mai bune, ceea ce înseamnă că proporția de tweet-uri etichetate corect ca fiind în limba română a crescut, validând astfel modulul nostru personalizat de identificare a limbii.

Tabelul 5.1 arată că, pe o perioadă de 12 luni, am identificat și extras aproximativ 51,000 de tweet-uri postate în limba română, ceea ce înseamnă că avem în medie ≈4,250 de tweet-uri lunar. Deși acest set de date poate părea mic la o primă vedere, susținem că

[2] `https://www.python.org/`

[3] `https://pypi.org/project/langid/`

este suficient pentru a oferi o imagine relevantă asupra activității vorbitorilor de română pe Twitter. Datorită preprocesării și identificării limbii, timpul total de execuție pentru acest proces de extragere a fost foarte mare, totalizând peste 72 de ore.

Year-Month	Number of texts labeled as Romanian in Twitter Stream	Number of texts labeled as Romanian with our approach	Percent Romanian	Execution Time (Hours)
2020-07	48415	4256	8.8	6.4
2020-08	56292	5100	9.06	7.7
2020-09	59346	4729	7.97	7.3
2020-10	57778	4788	8.27	7.5
2020-11	48867	4406	9.02	5.5
2020-12	52896	4935	9.33	6.1
2021-01	22621	1771	7.83	2.5
2021-02	56163	4621	8.23	6.21
2021-03	57993	5210	8.98	6.7
2021-04	24149	2095	8.68	2.7
2021-05	58576	4702	8.03	7.2
2021-06	52475	4330	8.25	6.3

Table 5.1: Compararea tweet-urilor etichetate în limba română

Acest număr relativ modest de tweet-uri extrase se aliniază și cu numărul redus de utilizatori români de Twitter. Conform Statista[4], numărul utilizatorilor de Twitter din România era de aproximativ 600,000 în perioada vizată de noi. Este important de menționat și faptul că nu toți utilizatorii își fac postările publice, iar unele conturi pot avea setări de confidențialitate active. Având în vedere aceste aspecte, împreună cu restricțiile geografice sau de altă natură, este posibil ca o parte din conținutul generat de utilizatori să fi fost omis în procesul de extragere a datelor din Twitter Stream.

Pentru limba română, mai multe studii au abordat crearea de modele lingvistice folosind arhitectura transformer împreună cu seturi de date de mari dimensiuni, pentru a crește nivelul de înțelegere și generare automată a limbajului. Printre acestea, menționăm lucrările lui Dumitrescu et al.[21] care au introdus primul model lingvistic bazat exclusiv pe transformers pentru limba română, depășind Multilingual BERT în sarcina NER, și pe cele ale lui Masala et al.[41] care au creat RoBERT folosind texte alese aleatoriu de pe internet și texte formale din pagini Wikipedia în limba română.

Dorim să dezvoltăm în total 8 variante distincte ale modelului RoBERTa, motivația fiind diversitatea și complexitatea lingvistică a limbii române, precum și necesitatea aplicării unor pași diferiți de preprocesare în anumite aplicații de tip NLP.

Variantele modelului BERTweetRO:

- Raw Cased
- Raw Uncased
- PreProcessed (PP) Cased
- PreProcessed (PP) Uncased
- Min Tokens Raw Cased
- Min Tokens Raw Uncased
- Min Tokens PP Cased
- Min Tokens PP Uncased

Primele 4 variante (Raw Cased, Raw Uncased, PP Cased și PP Uncased) diferă prin pașii de preprocesare și modul de tratare a majusculelor. Raw Cased păstrează scrierea

[4] `https://www.statista.com/forecasts/1143811/twitter-users-in-romania`

originală, Raw Uncased convertește toate caracterele în minuscule, în timp ce variantele PP Cased și PP Uncased transformă datele prin eliminarea URL-urilor, mențiunilor și hashtag-urilor Twitter, emoticonurilor și cuvintelor cheie rezervate platformei. Aceștia sunt principalii candidați pentru experimentele noastre, care ne vor permite să vedem cum sunt impactate modelele de sensibilitatea la majuscule și preprocesarea textelor. Următoarele 4 variante (Min Tokens Raw Cased, Min Tokens Raw Uncased, Min Tokens PP Cased și Min Tokens PP Uncased) sunt similare cu primele, diferența fiind că în aceste cazuri excludem tweet-urile cu mai puțin de 5 tokeni din setul de date. Cu această filtrare dorim să eliminăm cât mai multe instanțe zgomotoase din setul de antrenament, în speranța de a crește puterea predictivă a modelelor.

Tokenizarea poate fi văzută ca puntea care conectează reprezentarea naturală a textelor utilizate ca intrări și valorile numerice care codifică informațiile astfel încât acestea să poată fi utilizate de algoritmii de ML, existând numeroase strategii diferite pentru realizarea acestui proces.

Algoritmul Byte Pair Encoding (BPE) transformă textele într-o formă tabelară și este utilizat frecvent în diverse sarcini de modelare. O modificare adusă algoritmului original îi permite să combine tokeni care codifică atât caractere individuale, cât și cuvinte întregi [14]. În acest caz, toate caracterele unice sunt considerate un set inițial de n-grame de lungime 1. Apoi, cele mai frecvente perechi adiacente de caractere sunt îmbinate pentru a genera noi n-grame de lungime 2, iar toate aparițiile perechilor anterioare sunt înlocuite cu acest nou token. Procesul se repetă până când se atinge un vocabular de dimensiune prestabilită. Această versiune a BPE este adesea utilizată ca metodă de codificare pentru LLM-uri și transformere. În schimb, BPE standard nu combină cea mai frecventă pereche de octeți de date, ci o înlocuiește cu un nou octet care nu a fost văzut în setul de date inițial [57].

Datorită popularității și eficienței BPE, am decis să îl aplicăm în lucrarea noastră cu ajutorul implementării ByteLevelBPETokenizer din biblioteca Hugging Face[5]. Pentru antrenarea fiecărei variante a tokenizer-ului BERTweetRO am selectat următoarea configurație a parametrilor: (i) Vocabular cu 16,000 de tokeni (ii) Prag minim de frecvență de 2 (iii) Un set de tokeni speciali care conține $< s >$, $< pad >$, $< /s >$, $< unk >$, și $< mask >$.

Crearea tokenizatorilor constă în antrenarea acestora pentru a transforma corpusul de tweet-uri într-o serie de moduri care corespund variantelor noastre țintă de date. În timpul acestui proces, algoritmul BPE descoperă și învață modele statistice pe baza textelor de intrare și își actualizează iterativ vocabularul pentru a captura cât mai multe informații pentru fiecare unitate subcuvânt. Cele 8 modele de tokenizer rezultate au fost apoi salvate pentru utilizări viitoare.

Pentru a antrena cu succes modelele BERTweetRO pentru NLP în limba română, am selectat o configurație internă care poate oferi performanțe bune în raport cu timpii de antrenament și am integrat tokenizer-ele create anterior cu fiecare variantă BERTweetRO într-un mod consistent, pentru a ne asigura că hiperparametrii și sistemul end-to-end permit o comparație echitabilă a performanțelor în sarcinile NLP care urmează.

Am decis să folosim abordarea denumită Masked Language Modeling (MLM), implementată cu ajutorul RobertaForMaskedLM din biblioteca Hugging Face, o tehnică de pre-antrenament care permite modelelor de tip transformers să prezică tokenii mascați din secvențele de intrare. Acest lucru se realizează fără a fi nevoie de date etichetate, ceea ce o face o metodă de învățare nesupravegheată și, spre deosebire de alți algoritmi tradiționali,

[5] `https://huggingface.co/`

care pot prezice doar următorul token dintr-o anumită secvență, MLM poate utiliza atât tokenii anteriori, cât și pe cei următori pentru a prezice unul mascat. Specificațiile arhitecturale ale modelelor noastre BERTweetRO sunt următoarele: (i) Număr de neuroni per strat ascuns de 768 (ii) 12 capete de atenție, (iii) 12 straturi ascunse (iv) Probabilitate MLM de 15%.

Variantele BERTweetRO au fost antrenate pe durata a 5 epoci, deoarece am constatat că acest număr este suficient pentru a atinge un nivel acceptabil de convergență, fără a genera un cost prea mare în ceea ce privește timpul de execuție. Alegerea unei probabilități MLM de 15% se aliniază cu recomandările din literatura de specialitate [38, 30], bazate pe raționamentul că modelele nu pot învăța reprezentări bune atunci când este mascat prea mult sau prea puțin text. Pre-antrenamentul a fost efectuat pe GPU-ul nostru cu o dimensiune a lotului de 16, iar timpul total de execuție pentru toate cele 8 variante a fost puțin sub 4 ore, ceea ce este rezonabil având în vedere sarcina computațională ridicată în crearea de transformere de la zero.

Pentru a realiza ajustarea fină în analiza de sentiment se adaugă straturi noi peste un model BERT sau RoBERTa preantrenat, iar arhitectura rezultată este antrenată supervizat pe un set de date adnotat. Acest proces permite modelului să identifice și să învețe caracteristici relevante pentru sentiment din date, astfel încât să poată face predicții pe texte nemaivăzute. Calitatea modelului depinde, printre altele, de volumul datelor, de procesul de optimizare și de numărul de iterații.

Când vorbim despre limbile sub-reprezentate, precum limba română, menționăm lucrarea lui Ciobotaru et al. [17], în care autorii au antrenat un model fastText și au ajustat fin un model standard BERT, comparând ulterior performanțele acestora. Ei au selectat un set de date public care conține postări de pe Twitter legate de COVID-19, împărțite în 2 categorii (negative și pozitive). Ulterior, au extins acest set de date prin adăugarea clasei de sentiment "neutru" și prin includerea unui număr mai mare de exemple textuale pentru toate categoriile. Acest nou set de date a fost utilizat ca punct de referință în experimentele lor, iar rezultatele raportate pe setul de test au arătat că BERT a obținut un scor F1 macro de 0.84 în timp ce fastText a avut un scor mai slab de doar 0.73.

Așa cum a fost menționat în Capitolul 3, nu am identificat un set de date în limba română potrivit pentru analiza de sentiment (SA) a conținutului de tip microblogging, așa că am tradus setul de date Twitter US Airline Sentiment Tweets în română. Ulterior, am făcut o serie de experimente folosind noul set de date tradus în limba română împreună cu toate cele 8 variante MLM BERTweetRO dezvoltate de către noi. Utilizând BertClassifier din Hugging Face, am realizat ajustarea fiecărei variante pentru sarcina de SA, folosind tokenizer-ul corespunzător. În funcție de fiecare variantă de model, pipeline-ul de preprocesare aferent a fost aplicat în același mod în care a fost folosit în timpul pre-antrenării, pentru a ne asigura că vom putea face comparații valide între modele în etapele care urmează.

Vom evalua performanța celor 8 variante BERTweetRO ajustate pentru analiză de sentiment într-un studiu comparativ în care vom include cei mai buni clasificatori din Capitolul 3. Menționăm aici că toți clasificatorii tradiționali de SA incluși în studiul comparativ au trecut printr-un proces riguros de optimizare a hiperparametrilor. Prin urmare, este important de subliniat că modelele ajustate fin vor fi comparate cu clasificatori care au atins performanța maximă posibilă pentru setul de date selectat.

Pentru variantele noastre BERTweetRO am adăugat un strat secvențial suplimentar pentru clasificare, capabil să gestioneze ieșirea straturilor preantrenate și etichetele de sentiment așteptate. La fel ca în cazul Multilingual BERT, variantele noastre nu au fost

supuse unui proces de optimizare a hiperparametrilor din cauza limitărilor de timp. În schimb, parametrii pentru ajustarea fină au fost aleși pe baza recomandărilor standard din industrie, dar și ținând cont de parametrii inițiali folosiți în pre-antrenarea modelelor: dimensiunea lotului de 32, dimensiune ascunsă BERT de 768, dimensiune ascunsă pentru clasificare de 75, lungime maximă a secvenței de 80 tokeni, ReLU ca funcție de activare și categorical cross-entropy ca funcție de pierdere. Numărul de epoci, cuprins între 2 și 10, care asigură un nivel decent de acuratețe, a fost investigat și identificat individual pentru fiecare variantă BERTweetRO, precum și pentru Multilingual BERT.

Rezultatele analizei noastre comparative sunt prezentate în Tabelul 5.2, în care modelele sunt evaluate exclusiv pe setul de test. Având în vedere natura dezechilibrată a datelor, am setat Macro F1 ca principala măsură a performanței predictive. Astfel ne asigurăm că evaluarea tratează fiecare clasă în mod egal pentru a oferi o interpretare corectă a eficienței modelelor pentru toate categoriile de sentiment. Prin urmare, am ordonat tabelul în funcție de scorurile Macro F1 în ordine descrescătoare, ceea ce înseamnă că cele mai bune rezultate sunt prezentate în partea de sus a tabelului.

Classifier	Encoding	Macro F1	Weighted F1	Accuracy
Multilingual BERT	Multilingual Tokenizer	74.81	80.50	80.99
BERTweetRO Raw Cased	BERTweetRO Tokenizer Raw Cased	72.11	78.40	78.74
BERTweetRO Raw Uncased	BERTweetRO Tokenizer Raw Uncased	72.07	78.33	78.66
Bernoulli NB	TFIDF	71.91	78.20	78.20
BERTweetRO Raw Min Tokens Cased	BERTweetRO Tokenizer Raw Min Tokens Cased	71.67	78.14	78.61
BERTweetRO Raw Min Tokens Uncased	BERTweetRO Tokenizer Raw Min Tokens Uncased	71.58	78.00	78.47
LSTM	Word2Vec	71.39	77.98	78.17
Linear SVM	TFIDF	70.54	77.47	78.36
DNN	Word2Vec	69.19	76.23	77.20
Logistic Regression	TFIDF	69.04	76.45	77.81
CNN	Word2Vec	68.67	76.00	77.69
BERTweetRO PP Min Tokens Cased	BERTweetRO Tokenizer PP Min Tokens Cased	64.21	73.10	72.86
BERTweetRO PP Cased	BERTweetRO Tokenizer PP Cased	43.84	59.40	64.50
BERTweetRO PP Uncased	BERTweetRO Tokenizer PP Uncased	42.35	58.73	64.01
Random Forest	TFIDF	38.17	54.71	65.20
BERTweetRO PP Min Tokens Uncased	BERTweetRO Tokenizer PP Min Tokens Uncased	25.62	47.98	62.42

Table 5.2: Performanța în Analiza de Sentiment

Se poate observa că Multilingual BERT a depășit toți ceilalți clasificatori, obținând scoruri mai mari pentru toate metricile analizate, însă este important de menționat că cele mai bune variante ale noastre, și anume BERTweetRO Raw Cased și BERTweetRO Raw Uncased, au avut performanțe similare. Diferențele față de M-BERT sunt relativ mici, cu un scor Macro F1 cu aproximativ 3% mai mic și celelalte două metrici cu aproximativ 2% mai mici. Acest rezultat era într-o anumită măsură așteptat, având în vedere diferența de dimensiune a datelor folosite la pre-antrenare, M-BERT beneficiind de un volum mult mai mare și mai divers de date, în timp ce variantele noastre au fost antrenate pe un set

de date considerabil mai mic (≈51,000 de tweet-uri).

Surprinzător, variantele BERTweetRO care au fost antrenate pe texte cu o constrângere minimă de tokeni (BERTweetRO Raw Min Tokens Cased și Uncased) au obținut, de asemenea, rezultate competitive. Acest lucru ne indică că prin limitarea datelor folosite la antrenarea modelelor, i.e. păstrând doar textele cu mai mult de 5 tokeni, performanța predictivă obținută nu este afectată într-un mod semnificativ, iar în plus, acest lucru poate contribui la îmbunătățirea vitezei de execuție. Bernoulli NB, în ciuda simplității sale, a obținut rezultate bune situându-se între aceste 4 variante. Cu această excepție, modelele BERTweetRO care nu au utilizat pipeline-ul de preprocesare a textului au avut performanțe superioare tuturor modelelor clasice și celor de tip deep learning.

Un alt aspect care merită subliniat este că, indiferent de varianta BERTweetRO, cele antrenate pe texte originale (variantele Cased) au obținut rezultate ușor mai bune decât omologii lor antrenați pe texte convertite in minuscule (variantele Uncased). În mijlocul clasamentului se află LSTM, Linear SVM, DNN, Logistic Regression și CNN, cu performanțe decente care le fac adecvate pentru utilizare în scenarii reale.

În partea inferioră a tabelului, unde se află modelele cu cele mai slabe rezultate, regăsim toate variantele BERTweetRO care au fost combinate cu modulul nostru personalizat de preprocesare a textului, ceea ce indică clar că acest proces le-a afectat negativ performanțele predictive. Asta înseamnă că se pot obține modele BERT mai bune doar prin pre-antrenarea și ajustarea fină a acestora pe date brute din rețelele sociale, fără a fi necesare procese suplimentare de curățare a textului sau de inginerie a caracteristicilor. Aceste modele, alături de Random Forest, au înregistrat cele mai slabe performanțe predictive, ceea ce înseamnă că nu pot fi considerate adecvate pentru SA.

Ajustarea fină pentru clasificarea tematică (TC) implică utilizarea unui model preantrenat existent și adaptarea acestuia pentru a clasifica textele în funcție de subiectele de discuție regăsite in ele. Acest lucru se realizează prin adăugarea unor straturi specifice sarcinii în cadrul modelului BERT sau RoBERTa, după care întreaga arhitectură este antrenată într-un mod supervizat pe un set de date adnotat. Astfel, modelul poate identifica și învăța caracteristici tematice din date și poate realiza predicții pe texte nevăzute anterior cu ajutorul acestor reprezentări.

Ulterior, similar ajustării fine pentru SA, am desfășurat o serie de experimente folosind setul de date tradus News Category împreună cu toate cele 8 variante ale modelelor noastre pre-antrenate BERTweetRO MLM. Vom evalua performanța celor 8 variante BERTweetRO ajustate pentru clasificarea tematică într-un alt studiu comparativ, în care vom include cei mai buni clasificatori din Capitolul 4.

Pentru variantele BERTweetRO am adăugat un strat secvențial suplimentar pentru clasificare peste arhitectura existentă, capabil să gestioneze ieșirea straturilor preantrenate și etichetele de topic așteptate. La fel ca în cazul Multilingual BERT, variantele noastre nu au fost supuse procesului de optimizare a hiperparametrilor din constrângeri de timp. În schimb, parametrii folosiți pentru ajustarea fină au fost aleși pe baza recomandărilor din industrie dar și ținând cont de parametrii folosiți la pre-antrenarea modelelor: dimensiunea lotului de 32, dimensiunea ascunsă BERT de 768, dimensiunea ascunsă pentru clasificare de 128, lungimea maximă a secvenței de 120 tokeni, ReLU ca funcție de activare și categorical cross-entropy ca funcție de pierdere. Numărul de epoci, cuprins între 2 și 10, care duce la un nivel acceptabil de acuratețe a fost investigat și identificat individual pentru fiecare variantă BERTweetRO, precum și pentru M-BERT.

Spre deosebire de SA, unde scopul este de a detecta polaritatea globala a unui text, dificultatea lui TC constă și în numărul mare de clase țintă, care adesea se suprapun [25,

39]. Pentru a depăși această problemă, unii autori [26, 52] utilizează acuratețea Top-K în locul celei standard. În loc să clasifice un text într-o singură clasă și să o compare cu eticheta a-priori, modelul va prezice cele mai probabile K clase, iar dacă eticheta corectă se află printre acestea, textul este considerat corect clasificat. Ținem cont de acest aspect și raportăm acuratețea standard (adică Top-1), precum și Top-2 și Top-3, măsurate strict pe setul de test.

În Tabelul 5.3 prezentăm rezultatele studiului nostru comparativ, filtrate după acuratețea Top-1 în ordine descrescătoare, ceea ce înseamnă că cele mai bune modele apar la începutul tabelului. Aici observăm Multilingual BERT pe primul loc, cu acurateți impresionante Top-1, Top-2 și Top-3 de 72.63%, 85.56% și 90.25%. Acest rezultat era de așteptat dacă luăm în considerare volumul uriaș de date pe care acest model de tip transformer a fost preantrenat, ceea ce îi permite să genereze reprezentări inițiale robuste, chiar și pentru limba română, ce pot fi ulterior ajustate cu ușurință pentru TC cu ajutorul setului nostru de date tradus.

Classifier	Encoding	Top-1 Acc.	Top-2 Acc.	Top-3 Acc.	opt (s)	train (s)	test (s)
Multilingual BERT	Multilingual Tokenizer	72.63	85.56	90.25	N/A	7498	157
Linear SVM	TFIDF	66.73	80.05	85.30	11803	45.91	0.042
BERTweetRO Raw Uncased	BERTweetRO Tokenizer Raw Uncased	66.14	79.21	84.93	N/A	8436	135
BERTweetRO Raw Min Tokens Uncased	BERTweetRO Tokenizer Raw Min Tokens Uncased	66.07	79.10	84.80	N/A	6899	157
BERTweetRO Raw Min Tokens Cased	BERTweetRO Tokenizer Raw Min Tokens Cased	65.78	79.02	84.79	N/A	6923	157
BERTweetRO Raw Cased	BERTweetRO Tokenizer Raw Cased	65.63	78.75	84.48		8442	132
Bernoulli NB	TFIDF	62.80	77.70	84.11	398	0.59	0.04
CNN	Word2Vec	61.66	74.05	79.28	36797	56.98	1.65
BERTweetRO PP Min Tokens Cased	BERTweetRO Tokenizer PP Min Tokens Cased	54.55	66.60	72.94	N/A	6046	137
LSTM	Word2Vec	53.50	65.59	72.39	63605	119.1	6.16
Random Forest	TFIDF	16.56	28.89	37	845	0.6	0.183
BERTweetRO PP Min Tokens Uncased	BERTweetRO Tokenizer PP Min Tokens Uncased	16.56	28.89	37	N/A	6019	135
BERTweetRO PP Cased	BERTweetRO Tokenizer PP Cased	16.56	28.89	37	N/A	6027	135
BERTweetRO PP Uncased	BERTweetRO Tokenizer PP Uncased	16.56	28.89	37	N/A	6022	135

Table 5.3: Performanța în Clasificarea Tematică

În cursa pentru poziția a doua avem mai multe modele cu scoruri similare pentru toate cele 3 metrici de evaluare, și anume Linear SVM și cele 4 variante Raw BERTweetRO. Acești clasificatori au obținut acurateți Top-1 între $\approx 65\%$ și $\approx 66\%$, performanțe suficiente pentru aplicații reale. La fel ca în cazul ajustării fine pentru SA, variantele BERTweetRO care nu au folosit pipeline-ul personalizat de preprocesare a textului au obținut rezultate mai bune decât cele care l-au utilizat. Diferența dintre cele mai bune

variante ale noastre și Multilingual BERT este de aproximativ 6%, dar poate fi considerată acceptabilă ținând cont de dimensiunea relativ mică a setului de date folosit pentru pre-antrenare. Prin utilizarea unui set de date mai bogat și prin aplicarea optimizării hiperparametrilor am putea îmbunătăți performanța variantelor BERTweetRO în iterațiile viitoare.

Variantele BERTweetRO Raw Min Tokens Uncased/Cased au performanțe competitive, așa cum s-a observat și în cazul SA, ceea ce reconfirmă faptul că restricționarea datelor folosite pentru ajustarea fină nu reduce semnificativ performanța predictivă, în timp ce timpii de antrenare sunt îmbunătățiți. Acest aspect este evident prin compararea duratelor de antrenament între variantele Raw și Min Tokens, unde variantele Min Tokens au fost aproximativ cu 22% mai rapide.

Bernoulli NB și CNN urmează, cu scoruri ușor mai mici, ambele având performanțe similare în ceea ce privește Top-1, însă pentru Top-2 și Top-3 CNN rămâne în urmă cu o diferență destul de semnificativă. Aceste modele ar trebui să se comporte mai mult sau mai puțin la fel în predicția topicului principal al unui text, însă Bernoulli NB ar trebui preferat dacă al doilea și al treilea topic sunt considerate importante în analiza efectuată.

BERTweetRO PP Min Tokens Cased și LSTM împart locul al patrulea în clasamentul nostru, cu rezultate modeste în toate metricile. În partea de jos a tabelului se află RF și cele 4 variante BERTweetRO care au încorporat modulul de preprocesare a textului, toate înregistrând de departe cele mai slabe performanțe predictive, cu o acuratețe Top-1 de doar 16.5%. Acest lucru ne arată încă o dată eficiența abordării care presupune doar pre-antrenarea și ajustarea fină a modelelor BERT pe texte brute din social media, fără a fi nevoie de curățarea textului sau inginerie de caracteristici în prealabil. Prin urmare, acest grup de modele nu este viabil pentru clasificarea tematică în aplicații reale.

Capitolul 6

Evaluarea performanței analizei sentimentelor pe cazuri reale

Având în vedere că scopul final al acestei lucrări este aplicarea modelelor ML pentru a deduce polaritatea oricărui tweet în limba română, am etichetat manual două seturi de testare de dimensiuni reduse, fiecare conținând 120 de tweet-uri distincte. Primul set include tweet-uri specifice industriei aeriene, comparabile cu cele folosite pentru antrenarea modelelor noastre, iar al doilea conține tweet-uri generale. Vom evalua pe aceste seturi de testare modelele cu cele mai bune performanțe, așa cum au fost raportate în Capitolul 5: Multilingual BERT, BERTweetRO Raw Uncased, Bernoulli NB, LSTM și DNN [47]. În plus, vom compara aceste modele cu un instrument public terț de analiză a sentimentelor pentru limba română, numit Sentimetric[1], pentru a vedea unde ne situăm în raport cu o soluție comercială.

Tweet-urile au fost etichetate manual de către 5 voluntari umani, care au fost instruiți în prealabil cu privire la modul în care acest proces trebuie desfășurat. Fiecare voluntar și-a exprimat opinia cu privire la polaritatea tweet-ului, iar sentimentul final a fost stabilit ca fiind cel ales de majoritate. Statisticile privind etichetarea și modul în care a fost evaluată polaritatea sunt prezentate în Tabelul 6.1. Menționăm că sarcina de etichetare s-a dovedit a fi una dificilă chiar și pentru voluntari, întrucât doar pentru 43 de tweet-uri (35.8%) în cazul setului de date specific industriei aeriene și pentru 47 de tweet-uri (39.2%) în cazul tweet-urilor generale, toți cei 5 contributori au ajuns la o decizie unanimă. Mai mult, distribuția pe clase a acestor tweet-uri este semnificativ diferită față de cea din setul de date Twitter US Airline Sentiment Tweets.

Dataset	Negative	Neutral	Positive	Unanimous Annotation
Airline industry-specific tweets	51 (46.5%)	36 (30%)	33 (27.5%)	43 (35.8%)
General tweets	45 (37.5%)	32 (26.66%)	43 (35.8%)	47 (39.2%)
Twitter US Airline Sentiment Tweets	63%	21%	16%	N/A

Table 6.1: Statistici privind etichetarea manuală a sentimentului (număr de tweet-uri și procentaj)

Așa cum am procedat și în cazul experimentelor de fine tuning, raportăm Macro F1, Weighted F1 și Acuratețea ca metrici de evaluare pentru fiecare clasificator, însă având

[1] `http://sentimetric.ro/`

în vedere distribuția dezechilibrată a etichetelor în ambele seturi de date, trebuie din nou să folosim Macro F1 ca principala măsură a performanței modelelor.

Tabelul 6.2 conține performanțele predictive ale modelelor analizate pe cele 120 de tweet-uri reale în limba română, legate de industria aviației. În acest caz, observăm că Bernoulli Naive Bayes (NB) a obținut cel mai mare scor Macro F1 de 61.18%, urmat îndeaproape de Multilingual BERT, cu un scor puțin mai mic. Acest rezultat este oarecum surprinzător, având în vedere că Multilingual BERT a avut performanțe mai bune pe setul de evaluare utilizat în experimentele de fine tuning din Capitolul 5 însă succesul lui NB poate fi atribuit procesului de optimizare a hiperparametrilor prin care a trecut.

Classifier	Encoding	Macro F1	Weighted F1	Accuracy
Bernoulli NB	TFIDF	61.18	63.11	65
Multilingual BERT	Multilingual Tokenizer	60.45	63.38	65.83
BERTweetRO Raw Uncased	BERTweetRO Tokenizer Raw Uncased	54.57	56.68	60
LSTM	Word2Vec	52.71	55.18	58.33
DNN	Word2Vec	52.22	54.9	59.17
Sentimetric	N/A	45.72	46.99	47.5

Table 6.2: Performanțele modelelor pe tweet-uri în limba română specifice industriei aviatice

Setul de date are un număr redus de exemple, însă, în ciuda acestui fapt, varianta noastră BERTweetRO Raw Uncased a reușit să obțină un onorabil loc trei. Deși nu a reușit să depășească pe Bernoulli NB și Multilingual BERT, performanța BERTweetRO este superioară modelelor de tip deep learning LSTM și DNN. Scorul Macro F1 de 54.5%, care este cu aproximativ 6% mai mic decât cel mai bun scor, poate fi considerat acceptabil având în vedere că și voluntarii umani au întâmpinat dificultăți la etichetarea a textelor.

Cel mai important lucru pe care dorim să îl subliniem aici este că toate modelele noastre au depășit performanța Sentimetric. Acest lucru evidențiază impactul pozitiv al unei metodologii personalizate pentru antrenarea și validarea modelelor de învățare automată, în comparație cu soluțiile standard disponibile. Cu asta confirmăm și valoarea cunoștințelor specifice domeniului în obținerea unor rezultate mai bune în astfel de contexte, întrucât modelele noastre au fost create folosind tweet-uri din același domeniu.

În Tabelul 6.3 prezentăm performanțele modelelor pe setul de date cu tweet-uri generale în limba română. În acest caz situația este puțin diferită deoarece Multilingual BERT a obținut cel mai bun rezultat, cu un scor Macro F1 de 55.22%, urmat îndeaproape de BERTweetRO Raw Uncased, cu o diferență neglijabilă de doar 1%. Atât în această evaluare, cât și în cea anterioară, modelele bazate pe arhitectura Transformer s-au clasat pe primele locuri, ceea ce indică faptul că sunt mai fiabile pentru sarcina de predicție a sentimentului în practică.

Pe de altă parte, Bernoulli NB și DNN au obținut rezultate mai modeste, care le plasează la mijlocul clasamentului, însă mai surprinzător este faptul că LSTM a avut o performanță semnificativ mai slabă în acest caz situându-se sub toate celelalte modele, inclusiv sub Sentimetric. Motivele pentru care acest lucru s-a întâmplat necesită investigații viitoare, însă este posibil ca complexitatea modelului LSTM, împreună cu sensibilitatea sa la forma datelor de intrare, să fi afectat capacitatea modelului de a recunoaște corect tiparele de sentiment din aceste exemple.

Un detaliu interesant pe care dorim să îl evidențiem este că performanța modelelor pe tweet-urile generale este mai scăzută decât pe tweet-urile din domeniul industriei aviatice.

Classifier	Encoding	Macro F1	Weighted F1	Accuracy
Multilingual BERT	Multilingual Tokenizer	52.22	54.17	55.85
BERTweetRO Raw Uncased	BERTweetRO Tokenizer Raw Uncased	51.35	52.39	54.17
Bernoulli NB	TFIDF	48.48	49.42	48.33
DNN	Word2Vec	48.16	49.29	50.83
Sentimetric		46.16	47.3	49.17
LSTM	Word2Vec	43.17	44.29	45.83

Table 6.3: Performanțele modelelor pe tweet-uri generale în limba română

Această scădere care este mai evidentă în cazul modelelor clasice și al celor de tip deep learning este un rezultat direct al diferențelor de domeniu dintre textele folosite pentru antrenare și cele utilizate pentru evaluare. În cazul modelelor Multilingual BERT și BERTweetRO, declinul este mai puțin accentuat, datorită faptului că acestea au fost preantrenate pe texte variate, reușind astfel să se adapteze mai bine în acest scenariu.

Cu toate acestea, la fel ca în prima evaluare, menționăm că toate modelele noastre (cu excepția LSTM) au depășit semnificativ soluția comercială selectată ca punct de referință pentru comparație. Acest lucru validează din nou importanța fine tuning-ului personalizat și a optimizării modelelor în obținerea unor rezultate superioare pentru SA în limba română.

Pentru ambele domenii, rezultatele modelelor noastre sunt mai scăzute decât cele obținute pe setul de testare tradus utilizat în Capitolul 5, deoarece acum tweet-urile sunt reale, nu traduse, iar caracteristicile lor intrinseci diferă; din punct de vedere statistic, seturile sunt extrase din populații statistice diferite.

Capitolul 7

Concluzii

Cu acest capitol încheiem teza noastră de doctorat, trecând în revistă și rezumând cele mai importante elemente prezentate în cadrul acestei lucrări de cercetare. La baza motivației noastre pentru realizarea acestui studiu se află platformele de social media. Acestea au evoluat din "rețelele sociale" apărute la sfârșitul anilor '90, devenind populare la mijlocul anilor 2000. Inițial, rețelele sociale nu erau altceva decât site-uri web de bază, cu funcționalități limitate, axate în principal pe distribuirea de fotografii și mesaje între utilizatori conectați în mod explicit. Chiar și așa, baza de utilizatori a crescut exponențial într-un timp foarte scurt, iar în timp au apărut platformele moderne de social media. Acestea oferă în prezent un număr impresionant de funcționalități complexe și interconectate, dintre care menționăm: publicarea de conținut pentru audiențe globale, distribuirea și consumul oricărui tip de conținut (texte, imagini, videoclipuri, reclame etc.) și accesul facil în orice moment de pe dispozitive mobile sau desktop.

Prin urmare, nu este o surpriză faptul că 68% din populația lumii este activă în prezent pe platformele de socializare, iar pentru România acest procent este chiar mai mare, 90% din populația totală a țării fiind activă pe astfel de platforme la începutul anului 2025. Aceste cifre au fost un alt factor motivant pentru noi, deoarece utilizatorii generează volume mari de date textuale, bogate în varietate, cu viteză mare (caracteristici specifice big data). Prin urmare, este evident că atât cercetătorii, cât și entitățile publice sau private pot beneficia semnificativ de informațiile ascunse în acest tip de date.

Cu toate acestea, extragerea de informații utile din textele de pe rețelele de socializare este problematică. Spre deosebire de texte literare, textele de microblogging sunt pline de elemente de "limbaj neadecvat", cum ar fi: greșeli gramaticale, erori de scriere/tastare, abrevieri ne-standard, utilizarea emoji-urilor. În plus, acestea sunt prin natura lor informale și scurte, din cauza limitărilor de dimensiune impuse de platformă. De exemplu, un tweet pe Twitter are o limită de 280 de caractere, iar un comentariu pe TikTok are o limită de 150 caractere. Din aceste motive, abordările NLP bazate pe reguli sau dicționare pentru procesarea limbajului natural eșuează de cele mai multe ori, iar literatura recomandă cu tărie utilizarea învățării automate pentru a depăși aceste probleme. Pentru limbi populare precum engleza, franceza sau spaniola există numeroase resurse și instrumente capabile să gestioneze conținutul din social media, însă pentru limba română (și alte limbi subreprezentate) situația este mai dificilă, deoarece astfel de resurse sunt mult mai greu de găsit și folosit în medii de producție. Astfel, contribuția la corpul existent de resurse NLP pentru limba română reprezintă un alt factor motivant pentru noi.

La începutul acestei teze ne-am propus o serie de obiective, primul fiind studiul Analizei Sentimentelor pentru textele din social media în limba română. După o revizuire a

literaturii de specialitate, am constatat că majoritatea lucrărilor tratează SA ca o sarcină de clasificare binară (negativ vs. pozitiv). Această abordare este suficientă dacă știm dinainte că textele ce urmează a fi analizate sunt foarte polarizante, dar în majoritatea scenariilor acest lucru nu poate fi aplicat. În plus, multe studii se concentrează pe alte tipuri de texte, cum ar fi recenziile online de produse.

Am căutat în numeroase surse online un set de date etichetat care să satisfacă nevoile cercetării noastre, dar din păcate nu am reușit să găsit unul potrivit. Studiile pentru alte limbi sugerează că traducerea poate fi utilizată ca metodă pentru crearea unor resurse noi și utilizabile pentru învățare, astfel că am decis să testăm această ipoteză pentru limba română. Astfel, am selectat setul de date Twitter US Airline Sentiment Tweets pentru experimentele noastre, deoarece conține un număr mare de tweet-uri etichetate manual în trei clase de sentiment (negativ, neutru, pozitiv). Am tradus apoi acest set de date din engleză în română folosind un serviciu automat de traducere, pentru a crea setul nostru de antrenament. Folosind aceste date, am antrenat și evaluat o selecție largă de algoritmi de învățare automată din domeniul învățării clasice și deep learning, plus popularul Multilingual BERT, ca exemplu de model Transformer.

Rezultatele experimentelor de analiză a sentimentelor arată că modelele pentru limba română sunt comparabile cu omoloagele lor în limba engleză, variația în acurațețe fiind de doar ±2%. În acest context, am stabilit performanța de referință (state-of-the-art) pentru clasificarea a sentimentelor cu M-BERT la 83% pentru engleză și 81% pentru română. Doi dintre algoritmii clasici, Bernoulli NB și Linear SVM, au demonstrat performanțe competitive cu M-BERT, atingând o acuratețe de 78% în ambele limbi.

Al doilea nostru obiectiv este dedicat identificării subiectelor de discuție din texte cu format scurt. Pentru a atinge acest scop, am decis să utilizăm clasificarea supervizată tematică (Topic Classification) în locul modelării tematice (Topic Modeling), datorită dezavantajelor asociate naturii nesupervizate a metodelor TM: instabilitatea datelor, găsirea numărului optim de subiecte de extras nu este trivială, iar seturile de cuvinte cheie extrase necesită intervenție umană pentru a fi adnotate cu etichete de subiect relevante. Comparativ cu SA, eforturile actuale de cercetare în domeniul TC pentru conținutul microblogging în limba română sunt și mai limitate. Am putut găsi un singur studiu care a utilizat această abordare, dar setul de date folosit este mic și nu este disponibil public.

Văzând rezultatele încurajatoare ale experimentelor de SA, am decis din nou să găsim un set de date în limba engleză și să îl traducem în română. Am selectat News Category Dataset deoarece conține peste 200,000 de titluri și descrieri de știri de tip blog, grupate în 41 de clase tematice. Am constatat că unele dintre subiectele originale erau foarte specifice, în timp ce altele se suprapuneau, așa că, am îmbunătățit acest set de date prin gruparea categoriilor granulare și sinonime pentru a crea 26 de etichete tematice cu adevărat distincte. Apoi am antrenat și testat mai multe modele ML pe ambele seturi de date. Constatările noastre arată că modelele pentru limba română tind să aibă scoruri ușor mai mici decât cele pentru limba engleză, dar pentru cei mai buni clasificatori această diferență este neglijabilă (cu scăderi de 2–3%). Din cauza suprapunerii potențiale a temelor în instanțele textuale individuale, raportăm acuratețea Top-1, Top-2 și Top-3 ca metrici de evaluare. Astfel, M-BERT are din nou cele mai bune rezultate cu 74.85% Top-1, 87.29% Top-2 și 91.73% Top-3 pe datele în engleză, și 72.63% Top-1, 85.56% Top-2 și 90.25% Top-3 pe cele în română. Linear SVM este pe locul doi cu 68% Top-1, 82% Top-2 și 87% Top-3 pentru engleză, și 67% Top-1, 80% Top-2 și 85% Top-3 pentru română.

Toate modelele noastre de SA și TC, cu excepția BERT, au fost supuse unui proces de optimizare a hiperparametrilor, implementat cu algoritmi evolutivi, pentru a obține cea

mai bună performanță predictivă posibilă. Este de menționat că am comparat și timpii de execuție a modelelor în timpul optimizării, antrenării și inferenței. Având la îndemână descoperirile noastre experimentale, alții pot selecta mai ușor modelul potrivit nevoilor lor, pe baza performanței predictive așteptate în raport cu utilizarea hardware-ului.

Al treilea nostru obiectiv se referă la crearea unor modele Transformer destinate textelor din rețelele sociale românești. Folosind Twitter Stream Archive, care conține tweet-uri publice colectate pe o perioadă îndelungată, am selectat un interval de 12 luni pentru studiul nostru. După investigarea unui eșantion aleator de instanțe din acest interval, am constatat că majoritatea metadatelor din câmpul "limbă" pentru postările preetichetate ca fiind în limba română erau de fapt clasificări eronate. În acest sens, am implementat un modul de preprocesare pentru curățarea textelor, am rulat propriul proces de identificare a limbii și, în final, am reușit să extragem aproximativ 51,000 de postări cu text în limba română autentică. Folosind acest corpus, am pre-antrenat de la zero 8 variante de modele BERTweetRO, bazate pe arhitectura RoBERTa cu MLM. Aceste variante diferă între ele prin sensibilitatea la majuscule/minuscule (cased vs. uncased), formatul textului (brut vs. preprocesat) și numărul minim de tokeni în textele folosite pentru pre-antrenare (fără minim vs. minim 5 tokeni).

În continuare, am făcut fine-tuning variantelor noastre pe seturile de date traduse în română pentru SA și TC și am desfășurat o nouă serie de experimente. Am descoperit că variantele care folosesc textele brute sunt cele mai bune, însă cele cu constrângeri privind numărul minim de tokeni au o performanță predictivă doar puțin mai scăzută, dar rulează mai rapid. Preprocesarea textelor a avut un impact negativ major asupra capacității predictive și nu recomandăm asocierea acesteia cu transformerele. Pentru SA, cele mai bune variante BERTweetRO s-au clasat pe locul doi după M-BERT, cu un Macro F1 cu aproximativ 2.5% puncte mai mic decât scorul M-BERT de 74.8%. Pentru TC, cele mai bune variante BERTweetRO s-au situat pe locul trei după Linear SVM, însă acuratețea lor este aproape identică cu cea a Linear SVM. Aceste rezultate sunt remarcabile dacă luăm în considerare diferența extremă de dimensiune dintre datele folosite pentru a crea BERTweetRO față de datele extinse folosite la crearea lui Multilingual BERT.

Ultimul nostru obiectiv a constat în compararea celor mai performanți clasificatori ai noștri, printre care se numără BERTweetRO, M-BERT, Linear SVM, Bernoulli NB și DNN, cu un clasificator comercial de SA pentru limba română numit Sentimetric. Pentru a valida cu adevărat metodologia noastră de traducere, am colectat și etichetat manual două seturi de tweet-uri românești din viața reală: unul conținând tweet-uri legate de industria aeriană, iar celălalt conținând tweet-uri generale. În ambele domenii toți clasificatorii noștri au depășit Sentimetric, validând astfel cadrul nostru de modelare.

Prin scenariile și experimentele abordate în această teză am demonstrat, fără îndoială, că traducerea automată din engleză în română reprezintă o alternativă viabilă pentru crearea de resurse utile în procesarea limbajului natural. Mai mult, am arătat că este posibilă crearea de modele bazate pe BERT de la zero folosind un set de date de pre-antrenare relativ mic, compus din texte native în limba română, după care am ajustat aceste modele pentru alte sarcini NLP specifice folosind texte traduse în română, obținând rezultate bune în acest proces. Astfel, considerăm că munca noastră aduce contribuții valoroase la dezvoltarea resurselor lingvistice pentru procesarea textelor scurte în limba română. De asemenea, aceste concluzii ar trebui luate în considerare de alți cercetători care lucrează cu limbi subreprezentate și se confruntă cu dificultăți similare din cauza materialelor open-source limitate.

Referințe

[1] Charu C Aggarwal and Charu C Aggarwal. *Mining text data*. Springer, 2015.

[2] Amritanshu Agrawal, Wei Fu, and Tim Menzies. What is wrong with topic modeling? and how to fix it using search-based software engineering. *Information and Software Technology*, 98:74–88, 2018.

[3] Milam Aiken. An updated evaluation of google translate accuracy. *Studies in Linguistics and Literature*, 3:p253, 2019.

[4] Federico Albanese and Esteban Feuerstein. Improved topic modeling in twitter through community pooling. In *String Processing and Information Retrieval - 28th Intl. Symposium, SPIRE 2021*, volume 12944 of *LNCS*, pages 209–216. Springer, 2021.

[5] Andrei-Marius Avram, Darius Catrina, Dumitru-Clementin Cercel, Mihai Dascalu, Traian Rebedea, Vasile Florian Pais, and Dan Tufis. Distilling the knowledge of romanian berts using multiple teachers. *CoRR*, abs/2112.12650, 2021.

[6] Alexandra Balahur and José Manuel Perea Ortega. Sentiment analysis system adaptation for multilingual processing: The case of tweets. *Information Processing and Management*, 51(4):547–556, 2015.

[7] Alexandra Balahur and Marco Turchi. Comparative experiments using supervised learning and machine translation for multilingual sentiment analysis. *Computer Speech and Language*, 28(1):56–75, 2014.

[8] Carmen Banea, Rada Mihalcea, and Janyce Wiebe. Multilingual subjectivity: Are more languages better? In Chu-Ren Huang and Dan Jurafsky, editors, *COLING 2010, 23rd International Conference on Computational Linguistics, Proceedings*, pages 28–36. Tsinghua University Press, 2010.

[9] Francesco Barbieri, Luis Espinosa Anke, and José Camacho-Collados. XLM-T: A multilingual language model toolkit for twitter. *CoRR*, abs/2104.12250, 2021.

[10] Valentin Barrière and Alexandra Balahur. Improving sentiment analysis over non-english tweets using multilingual transformers and automatic translation for data-augmentation. In Donia Scott, Núria Bel, and Chengqing Zong, editors, *Proceedings of the 28th International Conference on Computational Linguistics, COLING 2020*, pages 266–271. International Committee on Computational Linguistics, 2020.

[11] David Blei and John Lafferty. Correlated topic models. *Advances in neural information processing systems*, 18:147, 2006.

[12] Jordan L. Boyd-Graber and David M. Blei. Syntactic topic models. In *Proc. of the 22nd Annual Conf. on Neural Information Processing Systems, 2008*, pages 185–192, 2008.

[13] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[14] Tom B Brown. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*, 2020.

[15] Xueqi Cheng, Xiaohui Yan, Yanyan Lan, and Jiafeng Guo. Btm: Topic modeling over short texts. *IEEE Transactions on Knowledge and Data Engineering*, 26(12):2928–2941, 2014.

[16] Alexandra Ciobotaru and Liviu P. Dinu. RED: A novel dataset for romanian emotion detection from tweets. In Galia Angelova, Maria Kunilovskaya, Ruslan Mitkov, and Ivelina Nikolova-Koleva, editors, *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021), Held Online, 1-3September, 2021*, pages 291–300. INCOMA Ltd., 2021.

[17] Alexandra Ciobotaru and Liviu P Dinu. Sart & covidsentiro: Datasets for sentiment analysis applied to analyzing covid-19 vaccination perception in romanian tweets. *Procedia Computer Science*, 225:1331–1339, 2023.

[18] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*, pages 4171–4186. Association for Computational Linguistics, 2019.

[19] Manuel Carlos Díaz-Galiano, Manuel García Vega, Edgar Casasola, Luis Chiruzzo, Miguel Ángel García Cumbreras, Eugenio Martínez Cámara, Daniela Moctezuma, Arturo Montejo-Ráez, Marco Antonio Sobrevilla Cabezudo, Eric Sadit Tellez, et al. Overview of tass 2019: One more further for the global spanish sentiment analysis corpus. In *IberLEF@ SEPLN*, pages 550–560, 2019.

[20] Susan T Dumais. Latent semantic analysis. *Annual review of information science and technology*, 38(1):188–230, 2004.

[21] Stefan Daniel Dumitrescu, Andrei-Marius Avram, and Sampo Pyysalo. The birth of romanian bert. *arXiv preprint arXiv:2009.08712*, 2020.

[22] Stefan Daniel Dumitrescu, Petru Rebeja, Beáta Lorincz, Mihaela Gaman, Andrei-Marius Avram, Mihai Ilie, Andrei Pruteanu, Adriana Stan, Lorena Rosia, Cristina Iacobescu, Luciana Morogan, George Dima, Gabriel Marchidan, Traian Rebedea, Madalina Chitez, Dani Yogatama, Sebastian Ruder, Radu Tudor Ionescu, Razvan Pascanu, and Viorica Patraucean. Liro: Benchmark and leaderboard for romanian language tasks. In Joaquin Vanschoren and Sai-Kit Yeung, editors, *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks 1, NeurIPS Datasets and Benchmarks 2021, December 2021, virtual*, 2021.

[23] Jacob Eisenstein. What to do about bad language on the internet. In Lucy Vanderwende, Hal Daumé III, and Katrin Kirchhoff, editors, *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, 2013*, pages 359–369. The Association for Computational Linguistics, 2013.

[24] Ahmed Elgohary, Matthias Boehm, Peter J Haas, Frederick R Reiss, and Berthold Reinwald. Compressed linear algebra for large-scale machine learning. *Proceedings of the VLDB Endowment*, 9(12):960–971, 2016.

[25] Matthew Gentzkow, Bryan Kelly, and Matt Taddy. Text as data. *Journal of Economic Literature*, 57(3):535–574, 2019.

[26] Maya R. Gupta, Samy Bengio, and Jason Weston. Training highly multiclass classifiers. *Journal of Machine Learning Research*, 15(1):1461–1492, 2014.

[27] Felix Hamborg, Karsten Donnay, Paola Merlo, et al. Newsmtsc: a dataset for (multi-) target-dependent sentiment classification in political news articles. Association for Computational Linguistics (ACL), 2021.

[28] Liangjie Hong and Brian D. Davison. Empirical study of topic modeling in Twitter. In *3rd Workshop on Social Network Mining and Analysis, SNAKDD 2009* , pages 80–88. ACM, 2010.

[29] Lucian Istrati and Alexandra Ciobotaru. Automatic monitoring and analysis of brands using data extracted from twitter in romanian. In Kohei Arai, editor, *Intelligent Systems and Applications - Proceedings of the 2021 Intelligent Systems Conference, IntelliSys 2021, Amsterdam, The Netherlands, 2-3 September, 2021, Volume 3*, volume 296 of *Lecture Notes in Networks and Systems*, pages 55–75. Springer, 2021.

[30] Peter Izsak, Moshe Berchansky, and Omer Levy. How to train bert with an academic budget. *arXiv preprint arXiv:2104.07705*, 2021.

[31] Max Jaderberg, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Reading text in the wild with convolutional neural networks. *International journal of computer vision*, 116(1):1–20, 2016.

[32] Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nédellec and Céline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, volume 1398 of *Lecture Notes in Computer Science*, pages 137–142. Springer Verlag, 1998.

[33] Ian Jolliffe and Jorge Cadima. Principal component analysis: A review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 374:20150202, 04 2016.

[34] Daniel Jurafsky and James H Martin. Vector semantics and embeddings. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*, pages 270–85, 2019.

[35] Kamran Kowsari, Kiana Jafari Meimandi, Mojtaba Heidarysafa, Sanjana Mendu, Laura Barnes, and Donald Brown. Text classification algorithms: A survey. *Information*, 10(4):150, 2019.

[36] Petra Kralj Novak, Jasmina Smailović, Borut Sluban, and Igor Mozetič. Sentiment of emojis. *Plos One*, 10(12):1–22, 12 2015.

[37] Quoc V. Le and Tomás Mikolov. Distributed representations of sentences and documents. In *Proceedings of the 31th International Conference on Machine Learning, ICML 2014, Beijing, China, 21-26 June 2014*, volume 32 of *JMLR Workshop and Conference Proceedings*, pages 1188–1196. JMLR.org, 2014.

[38] Yoav Levine, Barak Lenz, Opher Lieber, Omri Abend, Kevin Leyton-Brown, Moshe Tennenholtz, and Yoav Shoham. Pmi-masking: Principled masking of correlated spans. *arXiv preprint arXiv:2010.01825*, 2020.

[39] Bing Liu. *Sentiment analysis: Mining opinions, sentiments, and emotions*. Cambridge university press, 2020.

[40] Weijie Liu, Peng Zhou, Zhiruo Wang, Zhe Zhao, Haotang Deng, and Qi Ju. FastBERT: a self-distilling BERT with adaptive inference time. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel R. Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020*, pages 6035–6044. Association for Computational Linguistics, 2020.

[41] Mihai Masala, Stefan Ruseti, and Mihai Dascalu. Robert–a romanian bert model. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6626–6637, 2020.

[42] Andrew McCallum and Kamal Nigam. A comparison of event models for naive bayes text classification. In *Learning for Text Categorization: Papers from the 1998 AAAI Workshop*, pages 41–48, 1998.

[43] Walaa Medhat, Ahmed Hassan, and Hoda Korashy. Sentiment analysis algorithms and applications: A survey. *Ain Shams engineering journal*, 5(4):1093–1113, 2014.

[44] Tomás Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In Christopher J. C. Burges, Léon Bottou, Zoubin Ghahramani, and Kilian Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26: 27th Annual Conference on Neural Information Processing Systems 2013. Proceedings*, pages 3111–3119, 2013.

[45] Saif Mohammad, Cody Dunne, and Bonnie Dorr. Generating high-coverage semantic orientation lexicons from overtly marked words and a thesaurus. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 599–608, 2009.

[46] Naoki Mori, Masayuki Takeda, and Keinosuke Matsumoto. A comparison study between genetic algorithms and bayesian optimize algorithms by novel indices. In Hans-Georg Beyer and Una-May O'Reilly, editors, *Genetic and Evolutionary Computation Conference, GECCO 2005, Proceedings, Washington DC, USA, June 25-29, 2005*, pages 1485–1492. ACM, 2005.

[47] Dan Claudiu Neagu. Bertweetro: pre-trained language models for romanian social media content. *Studia Universitatis Babes-Bolyai Oeconomica*, 2025.

[48] Dan Claudiu Neagu, Andrei Bogdan Rus, Mihai Grec, Mihai Boroianu, and Gheorghe Cosmin Silaghi. *Topic Classification for Short Texts*, pages 207–222. Springer International Publishing, Cham, 2023.

[49] Dan Claudiu Neagu, Andrei Bogdan Rus, Mihai Grec, Mihai Augustin Boroianu, Nicolae Bogdan, and Attila Gal. Towards sentiment analysis for romanian twitter content. *Algorithms*, 15(10):357, 2022.

[50] Andrew Y. Ng and Michael I. Jordan. On discriminative vs. generative classifiers: A comparison of logistic regression and naive bayes. In Thomas G. Dietterich, Suzanna Becker, and Zoubin Ghahramani, editors, *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001]*, pages 841–848. MIT Press, 2001.

[51] Dat Quoc Nguyen, Thanh Vu, and Anh Tuan Nguyen. Bertweet: A pre-trained language model for english tweets. In Qun Liu and David Schlangen, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, EMNLP 2020 - Demos*, pages 9–14. Association for Computational Linguistics, 2020.

[52] Sechan Oh. Top-k hierarchical classification. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 2450–2456. AAAI Press, 2017.

[53] Varun Kumar Ojha, Ajith Abraham, and Václav Snásel. Metaheuristic design of feedforward neural networks: A review of two decades of research. *Engineering Applications of Artificial Intelligence*, 60:97–116, 2017.

[54] Najiba Ouled Omar, Azza Harbaoui, and Henda Ben Ghezala. Opinion mining and sentiment analysis on deft. *International Journal of Cognitive and Language Sciences*, 15(1):54 – 57, 2021.

[55] Andrew Ortony, Gerald L Clore, and Allan Collins. *The cognitive structure of emotions*. Cambridge university press, 2022.

[56] Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. fairseq: A fast, extensible toolkit for sequence modeling. In Waleed Ammar, Annie Louis, and Nasrin Mostafazadeh, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Demonstrations*, pages 48–53. Association for Computational Linguistics, 2019.

[57] Gerhard Paaß and Sven Giesselbach. *Pre-trained Language Models*, pages 19–78. Springer International Publishing, Cham, 2023.

[58] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.

[59] Razvan Pascanu, Tomás Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 1310–1318. JMLR.org, 2013.

[60] V. Paul Pauca, Farial Shahnaz, Michael W. Berry, and Robert J. Plemmons. Text mining using non-negative matrix factorizations. In Michael W. Berry, Umeshwar Dayal, Chandrika Kamath, and David B. Skillicorn, editors, *Proceedings of the Fourth SIAM International Conference on Data Mining, 2004*, pages 452–456. SIAM, 2004.

[61] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans, editors, *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014.

[62] Telmo Pires, Eva Schlinger, and Dan Garrette. How multilingual is multilingual bert? In Anna Korhonen, David R. Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019*, pages 4996–5001. Association for Computational Linguistics, 2019.

[63] Marco Pota, Mirko Ventura, Hamido Fujita, and Massimo Esposito. Multilingual evaluation of pre-processing for bert-based sentiment analysis of tweets. *Expert Systems with Applications*, 181:115119, 2021.

[64] Radim Rehurek and Petr Sojka. Gensim–python framework for vector space modelling. *NLP Centre, Faculty of Informatics, Masaryk University, Brno, Czech Republic*, 3(2), 2011.

[65] Sara Rosenthal, Noura Farra, and Preslav Nakov. Semeval-2017 task 4: Sentiment analysis in twitter. In Steven Bethard, Marine Carpuat, Marianna Apidianaki, Saif M. Mohammad, Daniel M. Cer, and David Jurgens, editors, *Proceedings of the 11th International Workshop on Semantic Evaluation, SemEval@ACL 2017, Vancouver, Canada, August 3-4, 2017*, pages 502–518. Association for Computational Linguistics, 2017.

[66] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing and management*, 24(5):513–523, 1988.

[67] Lucas Nunes Sequeira, Bruno Moreschi, Fábio Gagliardi Cozman, and Bernardo Fontes. An empirical accuracy law for sequential machine translation: the case of google translate. *CoRR*, abs/2003.02817, 2020.

[68] Ryan A Stevenson, Joseph A Mikels, and Thomas W James. Characterization of the affective norms for english words by discrete emotional categories. *Behavior research methods*, 39(4):1020–1024, 2007.

[69] Ayisha Tabassum and Rajendra R Patil. A survey on text pre-processing & feature extraction techniques in natural language processing. *International Research Journal of Engineering and Technology (IRJET)*, 7(06):4864–4867, 2020.

[70] Anca Maria Tache, Mihaela Gaman, and Radu Tudor Ionescu. Clustering word embeddings with self-organizing maps. application on laroseda - A large romanian sentiment data set. In Paola Merlo, Jörg Tiedemann, and Reut Tsarfaty, editors, *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021*, pages 949–956. Association for Computational Linguistics, 2021.

[71] Adrian Vasile, Roxana Rădulescu, and Ionel-Bujorel Păvăloiu. Topic classification in romanian blogosphere. In *12th Symposium on Neural Network Applications in Electrical Engineering (NEUREL)*, pages 131–134. IEEE, 2014.

[72] Ike Vayansky and Sathish AP Kumar. A review of topic modeling methods. *Information Systems*, 94:101582, 2020.

[73] Pradnya A Vikhar. Evolutionary algorithms: A critical review and its future prospects. In *2016 International conference on global trends in signal processing, information computing and communication (ICGTSPICC)*, pages 261–265. IEEE, 2016.

[74] Xuerui Wang and Andrew McCallum. Topics over time: a non-Markov continuous-time model of topical trends. In *12th ACM SIGKDD Intl. Conf. on Knowledge Discovery and Data Mining, 2006*, pages 424–433. ACM, 2006.

[75] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771, 2019.

[76] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing [review article]. *IEEE Comput. Intell. Mag.*, 13(3):55–75, 2018.

[77] Jichuan Zeng, Jing Li, Yan Song, Cuiyun Gao, Michael R. Lyu, and Irwin King. Topic memory networks for short text classification. In *Proc. of the 2018 Conf. on Empirical Methods in Natural Language Processing*, pages 3120–3131. ACL, 2018.