Raport: Analiza Sentimentului pe Text în Limba Română

1. Introducere și Descriere Generală

Scopul Proiectului

Acest proiect are ca scop dezvoltarea și evaluarea unui subsistem de analiză a sentimentului (Sentiment Analysis) pentru limba română, utilizând rețele neurale recurente (RNN, LSTM). Obiectivul principal este clasificarea automată a recenziilor în două categorii: **pozitiv** și **negativ**.

Setul de Date

Pentru antrenarea și evaluarea modelelor, s-a utilizat setul de date `ro_sent`, disponibil public. Acesta este constituit din recenzii de produse și filme, fiind un benchmark relevant pentru prelucrarea limbajului natural (NLP) în limba română.

- **Dimensiune**:
  - Set de antrenare: 17.941 exemple.
  - Set de testare: 11.005 exemple.

Problema este abordată ca o sarcină de clasificare binară supervizată.

---

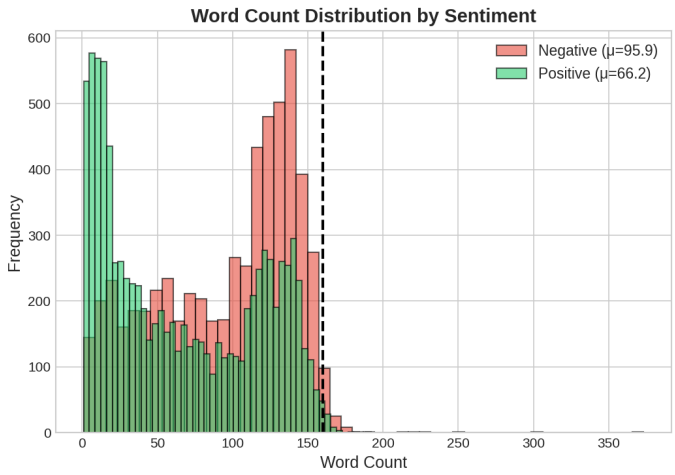
2. Explorarea Datelor (EDA)

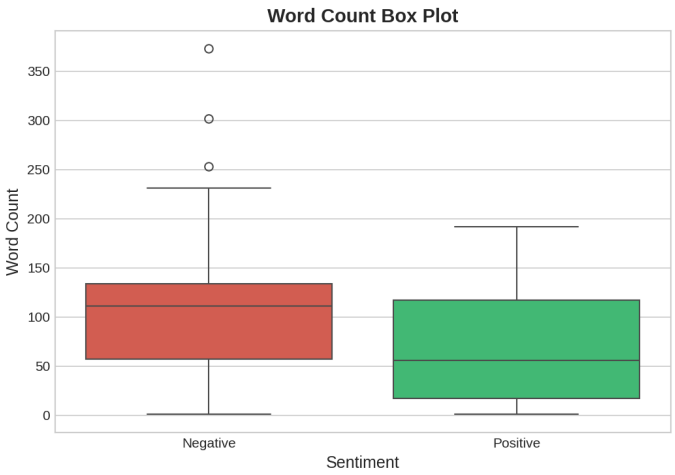
În această etapă, am analizat distribuția datelor și caracteristicile textuale pentru a înțelege mai bine natura task-ului și eventualele provocări.

2.1 Analiza Echilibrului de Clase

Dat fiind dezechilibru semnificativ între clasile pozitive și negative (aproximativ 2:1), modelele ar putea tinde să favorizeze clasa majoritară. Pentru a mitiga acest risc, în antrenarea fara augmentare am folosit`WeightedRandomSampler`, iar ulterior, dupa ce am aplicat augmentare offline pentru clasa minoritară si am obținut o distribuție echilibrată, am renuntat la aceasta tehnica`.

2.2 Statistici despre Text

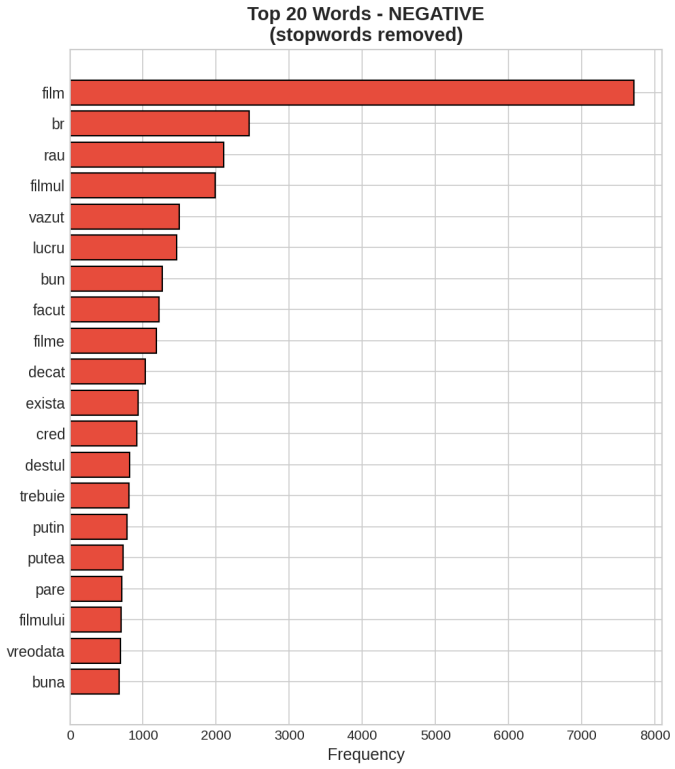


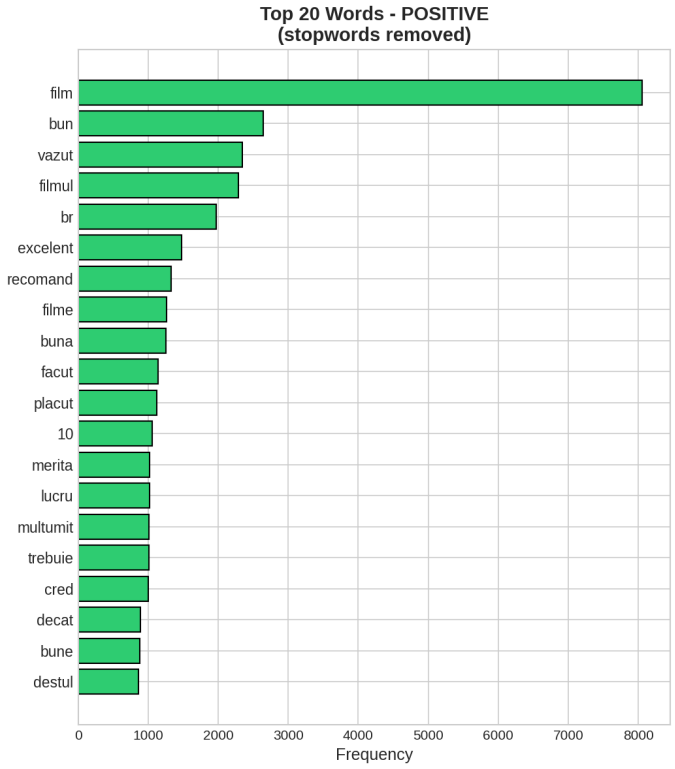


Majoritatea recenziilor au o lungime moderată, ceea ce justifică utilizarea unei lungimi maxime a secvenței (`max_seq_len`) de 160 de tokeni pentru a acoperi 95% din recenziile din setul de date.

2.3 Analiza Lexicală

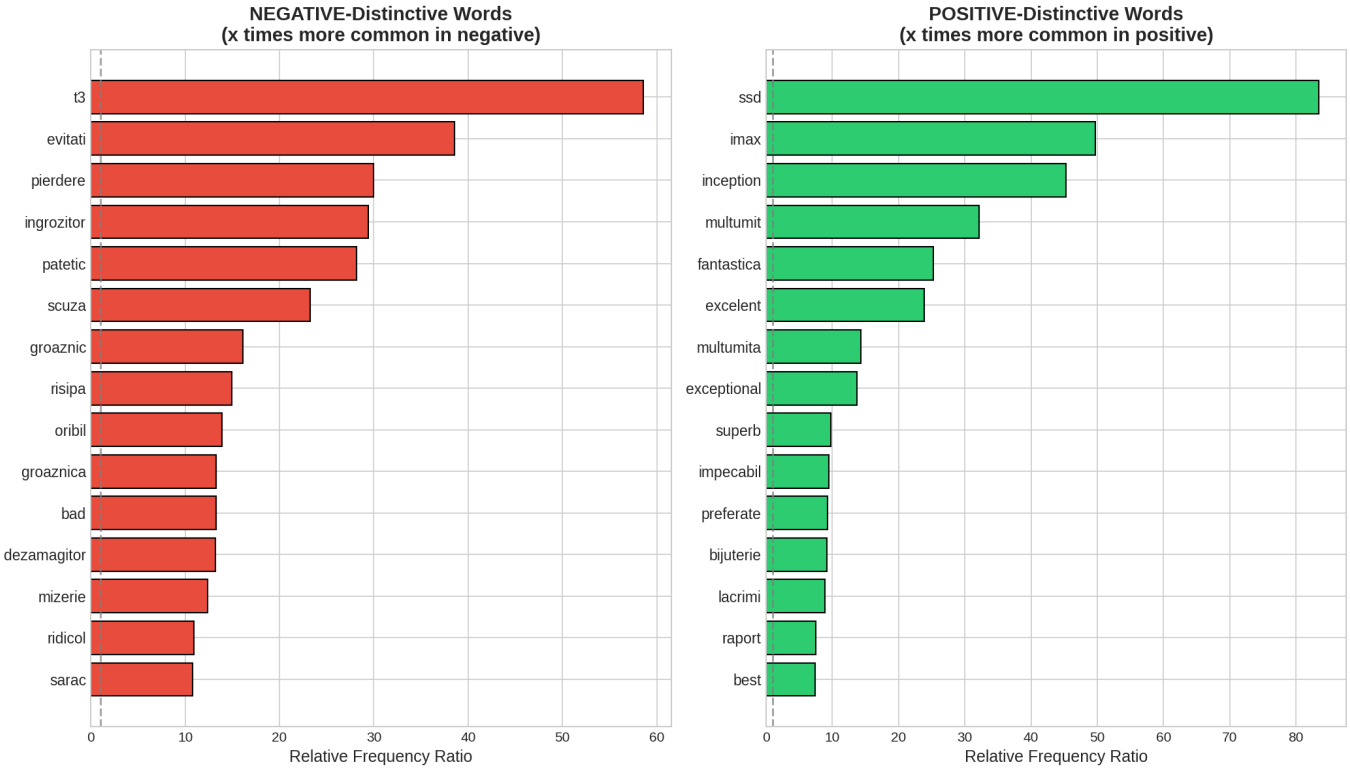
Pentru a intui ce cuvinte sunt discriminative pentru fiecare clasă, am vizualizat cele mai frecvente cuvinte (excluzând stop-words banale, dar păstrând cuvintele relevante pentru sentimente).





De asemenea, norii de cuvinte (Word Clouds) oferă o perspectivă vizuală asupra vocabularului specific fiecărui sentiment:

De asemenea, graficul de mai jos evidențiază cuvintele cele mai distinctive (cu cea mai mare diferență de frecvență relativă) pentru fiecare clasă:



Această analiză confirmă că anumite cuvinte (adjective precum "rau" vs "bun"/"bine"/"bună", "recomand", "multumit" vs. "slab", "evitati", "pierdere", "ingrozitor") sunt indicatori puternici ai sentimentului, justificând abordarea bazată pe embeddings și rețele recurente care pot capta contextul acestor cuvinte.

3. Preprocesare și Embeddings

Procesarea textului este esențială pentru a transforma recenziile brute într-un format compatibil cu rețelele neurale. Pipeline-ul de preprocesare (`src/preprocessing/text.py`) a inclus următorii pași:

1. **Curățarea Textului**:
    - Normalizare Unicode.
    - Standardizarea diacriticelor.
    - Păstrarea punctuației relevante și eliminarea caracterelor speciale(ex. taguri html) care introduc zgomot. **Note**:
    - Am ales sa pastrez stop words fiindca am observat ca prin eliminarea acestora, chiar daca aveam grija sa le pastrez pe cele relevante pentru sentimet(negatii, conjunctii, etc), obtineam

rezultatea mai proaste. Asta se intampla probabil fiindca prin prin eliminarea cuvintelor de legatura sentimentul exprimat in textul se schimba
    - Am pastrat semnlee de punctuatie si emoji-urile, ele in special fiind relevante pentru recenziile negative(multe "!")

2. **Tokenizare**:
    - Utilizarea **spaCy** (`ro_core_news_sm`) pentru o segmentare corectă a cuvintelor.
3. **Gestionarea Vocabularului și Embeddings**:
    - **Vocabular**: Construit pe baza setului de antrenare, păstrând cuvintele cu o frecvență minimă de 2 apariții pentru a reduce zgomotul.
    - **Embeddings**: Am utilizat vectori pre-antrenați **FastText** (`cc.ro.300.bin`), cu o dimensiune de 300. Aceștia captează relații semantice între cuvinte și permit modelului să generalizeze mai bine, chiar și pentru cuvinte care apar rar în setul de antrenare.
    - Cuvintele din afara vocabularului (OOV) au primit reprezentări generate din sub-cuvinte (folosind FastText), iar acestea nu au fost ajustate în timpul antrenării.

---

4. Arhitecturi de Modele

Am experimentat cu trei arhitecturi principale bazate pe rețele recurente. Toate modelele au împărțit un pipeline-ul descris anterior si au folosit o configurație de antrenament similară.

4.1 Configurație Comună și Hiperparametri

In urma mai multor experimenete, am ales sa folosesc urmatorul det de hiperparametrii pentru toate arhitecturile:

- **Embedding**: FastText pre-antrenat (dimensiune 300), înghețat (`freeze_embeddings=True`).
- **Encoder**: Bidirecțional (`bidirectional=True`), 2 straturi (`num_layers=2`).
- **Dimensiune Ascunsă (Hidden Dim)**: 128 (per direcție, deci 256 total la ieșirea encoder-ului). Cand am folosit o dimentine mai mare modelul intre rapid in overfit
- **Clasificator (Head)**: Strat complet conectat (FC) cu dimensiunea 128, activare ReLU și Dropout.
- **Optimizator**: AdamW (`lr=0.0005`, `weight_decay=0.001`).
- **Regularizare**: Dropout (`p=0.5`) aplicat după Embedding, între straturile recurente și în clasificator. Am incercat si p=0.5,dar modelul intra in overfit
- **Batch Size**: 64.
- **Lungime Secvență**: Max 160 tokeni(95% recenziile sunt mai scurte decat aceasta).

4.2 Simple RNN (Recurrent Neural Network)

Acesta este modelul de referință (baseline), implementat în clasa `SimpleRNN`. Deși rapid, este vulnerabil la problema gradientului evanescent.

Flux de Date (Pipeline):

1. **Input**: Indici tokeni $[B, 160]$.
2. **Embedding**: $[B, 160, 300]$ + Dropout(0.5).

3. **Encoder (Elman RNN)**:
    - 2 straturi, bidirecțional, non-linearitate `tanh`.
    - Output: $[B, 160, 256]$ (concatenare forward+backward).
4. **Pooling**: Max Pooling peste axa timpului $\rightarrow [B, 256]$.
5. **Clasificator**:
    - Linear(256 $\rightarrow$ 128) + ReLU + Dropout(0.5).
    - Linear(128 $\rightarrow$ 2).

4.3 LSTM (Long Short-Term Memory)

Acest model înlocuiește celula RNN simplă cu celule LSTM, capabile să mențină informația pe termen lung prin porți logice (input, forget, output gates).

Arhitectură:

- **Encoder**: LSTM Bidirecțional, 2 straturi.
- **Funcționare**: La fiecare pas, LSTM-ul decide ce informație să păstreze și ce să uite, fiind ideal pentru recenzii lungi unde sentimentul poate fi determinat de cuvinte aflate la distanță mare.
- **Pooling**: S-a utilizat tot **Max Pooling** pentru a extrage cele mai proeminente trăsături (features) din întreaga secvență.

4.4 BiLSTM cu Atenție (Attention Mechanism)

Acest model rafinează modul în care sunt agregate stările ascunse ale LSTM-ului. În loc de un simplu Max/Mean Pooling, mecanismul de atenție învață să "pondereze" fiecare cuvânt în funcție de importanța sa pentru decizia finală.

Mecanismul de Atenție:

1. **Calcul Scoruri**: Stările ascunse H $[B, 160, 256]$ trec printr-un MLP mic (Linear $\rightarrow$ Tanh $\rightarrow$ Linear) pentru a obține un scor de importanță pentru fiecare pas de timp.
2. **Ponderi (Softmax)**: Scorurile sunt normalizate pentru a obține ponderi α care sumează la 1.
3. **Context Vector**: Se calculează media ponderată a stărilor ascunse: $C = \sum \alpha_t \cdot h_t$.
4. **Clasificare**: Vectorul context C este trimis către clasificatorul final.

Această abordare permite modelului să ignore cuvintele de umplutură și să se concentreze exclusiv pe adjectivele și verbele cheie (ex: "excelent", "dezamăgit").

---

5. Augmentarea Datelor

Pentru a îmbunătăți generalizarea și a combate overfitting-ul, am aplicat tehnici de augmentare a datelor, atât offline (înainte de antrenare), cât și online (dinamic, în timpul antrenării):

5.1 Tehnici Utilizate

1. **Fără Augmentare (No Aug)**: Setul de date original.
2. **Augmentare Offline (Balanced)**:
    - Am aplicat **Back-Translation** (Română -> Engleză -> Română) **doar pentru clasa minoritară (Negativ)**.

- Scopul a fost atingerea echilibrului perfect între clase înainte de antrenare.

3. **Full Augmentation (EDA+)**:
   - Pe lângă setul echilibrat offline, am aplicat augmentări **online** dinamice (Random Swap, Random Delete, Synonym Replacement, contextual insert, contextual replace) în timpul antrenării, cu o probabilitate de 10%.

---

6. Evaluare și Rezultate

6.1 Evoluția Modelelor (Training Curves)

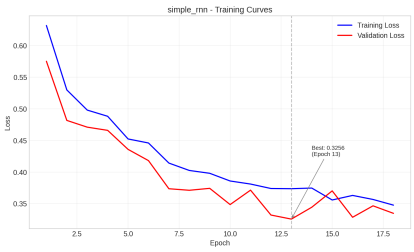
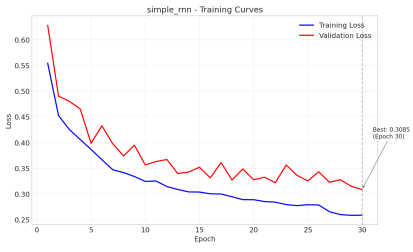
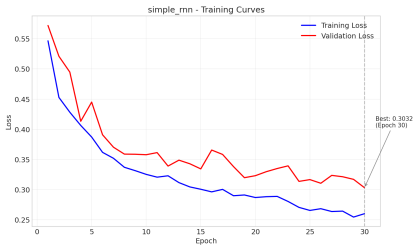
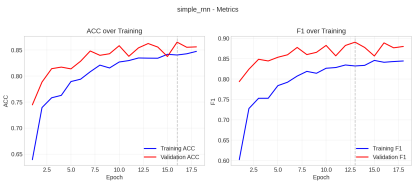
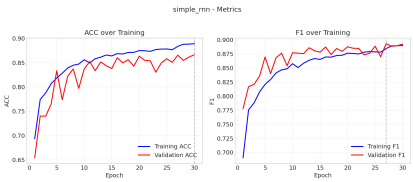
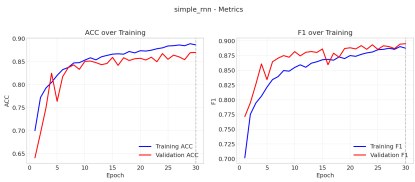
Pentru a analiza stabilitatea și viteza de convergență, prezentăm curbele de Loss, Acc și F1 pentru fiecare arhitectură în cele trei scenarii de antrenare: **Fără Augmentare**, **Balanced (Offline)** și **EDA+ (Offline + Online)**.

Analiza vizuală a acestor grafice relevă câteva tendințe clare:

- **Fără Augmentare (No Aug)**: Modelele tind să suprapună rapid (overfitting). Se observă o divergență timpurie între curba de antrenare (care scade continuu) și cea de validare (care stagnează sau crește).
- **Balanced (Offline)**: Echilibrarea claselor stabilizează semnificativ antrenarea. Curbele de validare urmăresc mai îndeaproape curbele de antrenare, indicând o generalizare mai bună.
- **EDA+ (Offline + Online)**: Introducerea zgomotului prin augmentare online produce curbe ușor mai "zgomotoase" local, dar previne overfitting-ul pe termen lung, forțând modelul să învețe trăsături mai robuste.

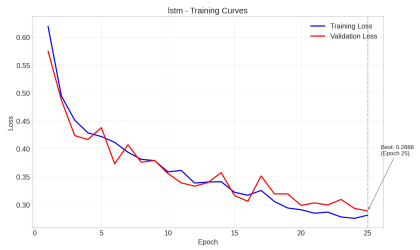
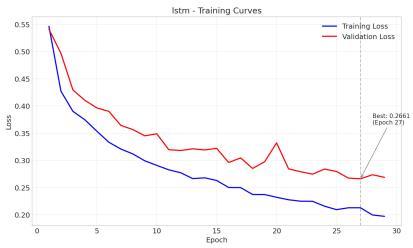
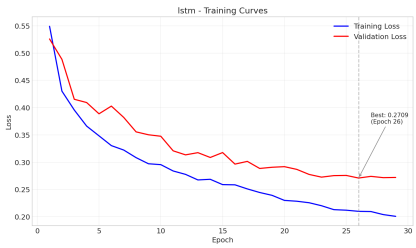
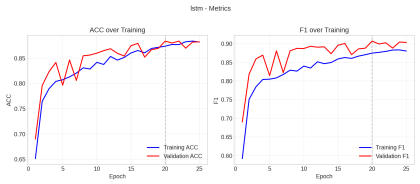
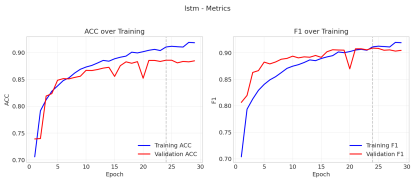
Simple RNN (Bidirecțional)

La modelul Simple RNN, impactul lipsei de date echilibrate este cel mai evident. Varianta **No Aug** prezintă valori ale losului in general mai mari decat versiunile cu augmentare. Un fenomen interesant la acest model fără augmentare este faptul că pierderea pe validare este mai mică decât cea pe antrenare, ceea ce se explică prin regularizarea puternică (Dropout 0.5) activă doar la antrenament. Variantele cu augmentari oferă o curbă de învățare mai lină și mai stabilă.

Fără Augmentare	Balanced (Offline)	EDA+ (Offline + Online)
		
		

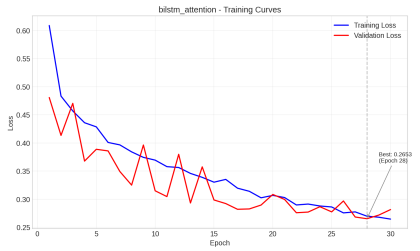
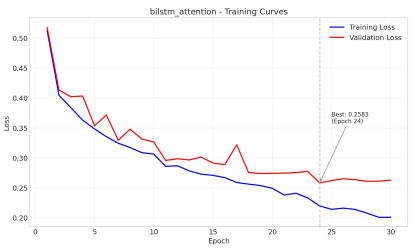
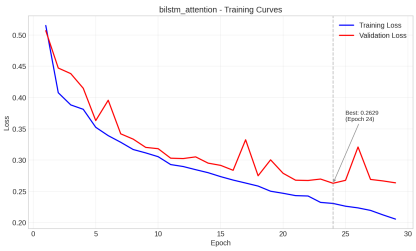
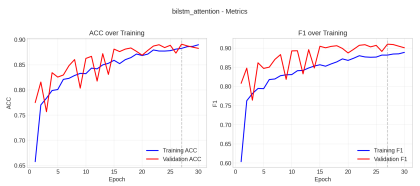
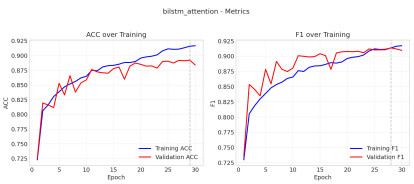
LSTM (Bidirecțional)

LSTM-ul gestionează mai bine datele neechilibrate decât RNN-ul simplu, dar echilibrarea a introdus niste spike-uri ciutate pe setul de validare in timpul antrenarii, care dispar cand sunt introduse si augmentarile online.

Fără Augmentare	Balanced (Offline)	EDA+ (Offline + Online)
		
		

BiLSTM cu Atenție

Acest model demonstrează cea mai bună performanță. Mecanismul de atenție, combinat cu datele echilibrate, permite atingerea unor valori mai mici ale loss-ului și menținerea lor stabilă.

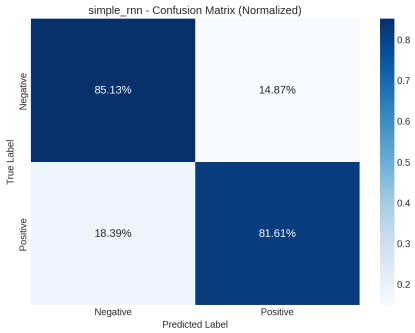
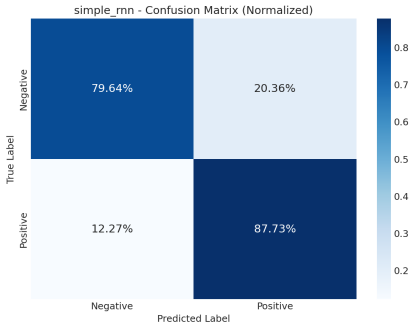
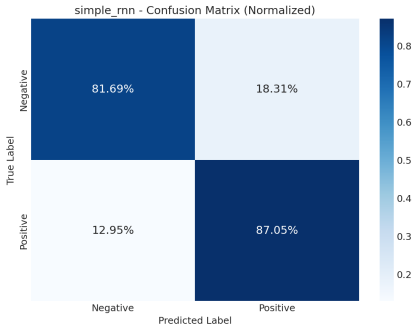
Fără Augmentare	Balanced (Offline)	EDA+ (Offline + Online)
		
		

6.2 Matricile de Confuzie: Impactul Augmentării

Analiza matricilor de confuzie (normalizate) evidențiază modul în care augmentarea corectează bias-ul modelului.

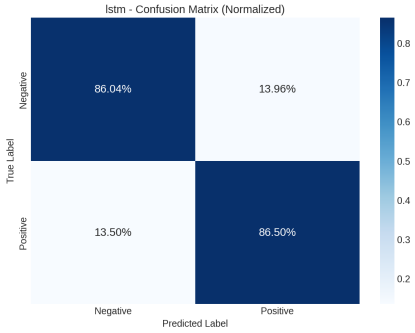
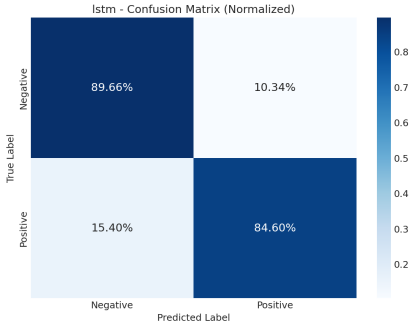
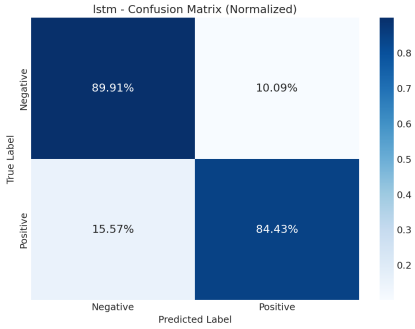
Simple RNN: Confuzia la Balansare

Fără Augmentare	Balanced (Offline)	EDA+ (Offline + Online)

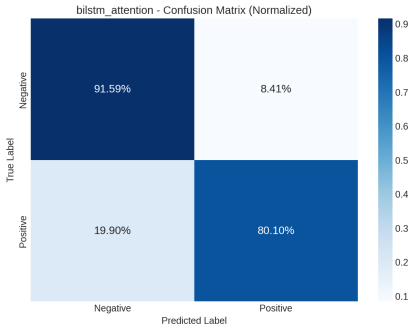
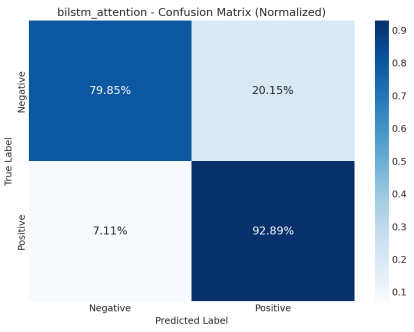
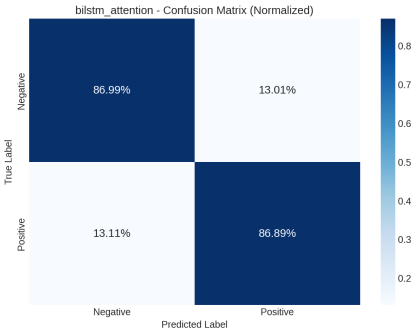
Fără Augmentare	Balanced (Offline)	EDA+ (Offline + Online)
		

Analiza erorilor pentru Simple RNN: Modelul standard suferă de o rată mare de False Negatives din cauza incapacității de a reține dependențe lungi. Surprinzător, echilibrarea prin back-translation a degradat performanța, introducând zgomot semantic pe care modelul nu l-a putut gestiona. În schimb, augmentarea online (EDA+) a stabilizat modelul, acționând ca un regularizator și forțându-l să se bazeze pe cuvinte cheie, reducând astfel erorile de tip False Positive.

BiLSTM Standard – "Cazul Ideal" (Succesul Balansării)

Fără Augmentare	Balanced (Offline)	EDA+ (Offline + Online)
		

BiLSTM + Attention – "Supra-Corecția și Stabilizatorul"

Fără Augmentare	Balanced (Offline)	EDA+ (Offline + Online)
		

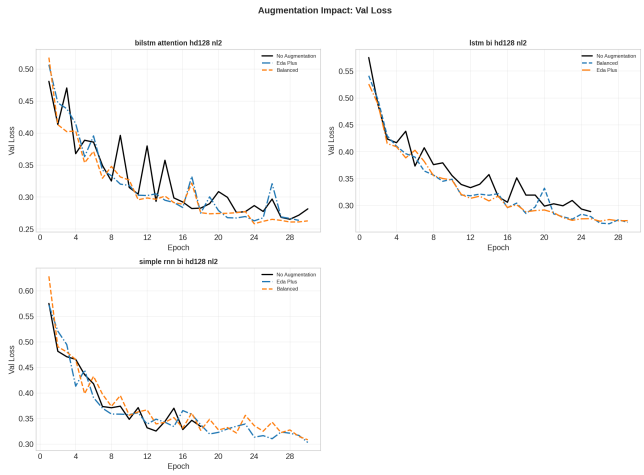
Observație: Analiza matricelor de confuzie relevă comportamente distincte între arhitecturi. În timp ce BiLSTM-ul standard a beneficiat liniar de pe urma balansării offline, îmbunătățindu-și detecția clasei minoritare (Negative), modelul BiLSTM cu Atenție a suferit o supra-corecție severă pe datele balansate, pierzând din specificitate.

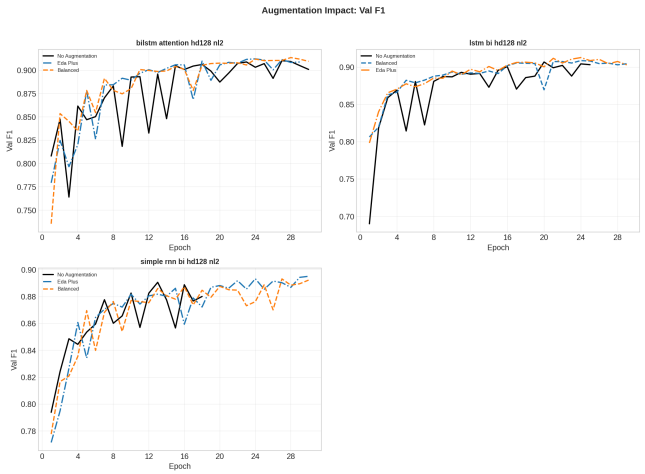
Totuși, introducerea augmentării online (EDA+) a remediat această instabilitate a mecanismului de atenție. Aceasta varianta oferind cel mai bun compromis, menținând o precizie ridicată pe negative (TN: 4200) fără a sacrifica masiv recall-ul pe pozitive, demonstrând că zgomotul controlat previne overfitting-ul pe artefactele generate de back-translation."

6.3 Impactul Augmentării

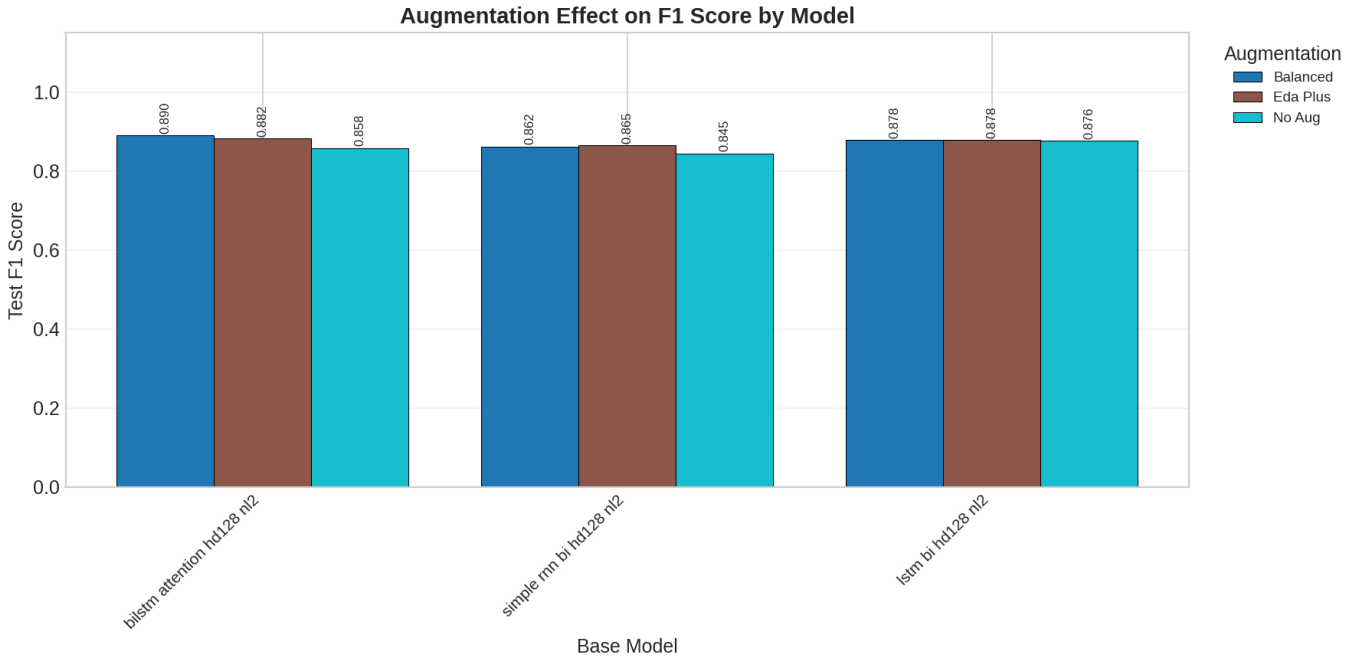
Comparăm direct efectul strategiilor de augmentare asupra pierderii (Loss) și scorului F1 pe setul de validare.

Pierdere (Val Loss)	F1-Score (Val F1)





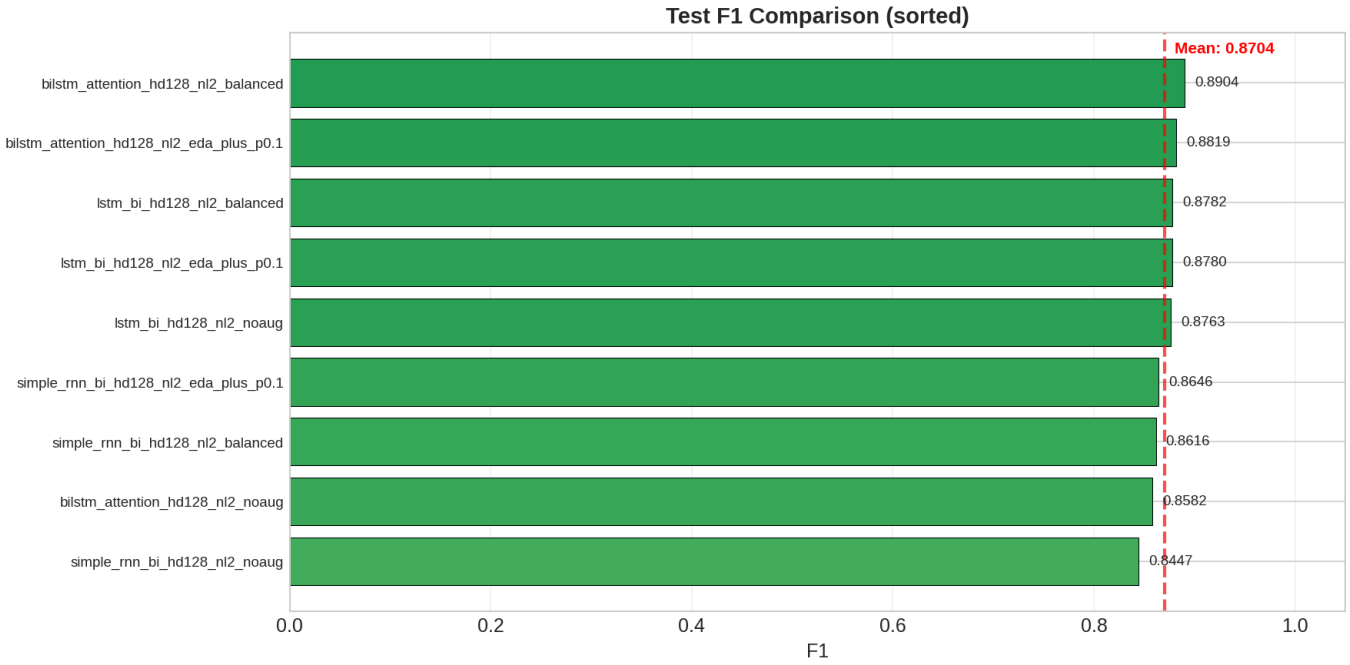
Efectul Global al Augmentării: Graficul de mai jos sumarizează câștigul de performanță adus de fiecare strategie.



Se observă clar că **Echilibrarea prin backtranslation** reduce overfitting-ul și îmbunătățește convergența, în timp ce **EDA+** oferă o stabilitate suplimentară. Procesul de backtranslation a crescut dimensinea setului de date doar cu ~18%, beneficiul adus sugereaza ca aceata metoda are potențialul de a aduce cele mai mari imbunătăți. Un experiment in care am dubla setul de data prn backtranslation ar fi interesant de urmarit.

6.4 Comparație Finală și Clasament

Ierarhia finală a tuturor modelelor antrenate, ordonată după performanța pe setul de test (F1-Score).



Tabelul de mai jos centralizează performanța tuturor modelelor pe setul de testare. Se remarcă o corelație directă între complexitatea modelului și performanță, dar **augmentarea datelor joacă un rol egal sau chiar mai important**. De exemplu, un model simplu (Simple RNN) cu augmentare (F1 ~0.865) depășește un model complex (BiLSTM Attention) fără augmentare (F1 ~0.858), subliniind importanța calității datelor. Modelele antrenate pe setul echilibrat ("balanced") domină clasamentul, în special la capitolul Recall.

Experiment	Model	Augmentare	F1-Score	Acuratețe	Precizie	Recall
BiLSTM Attention (Balanced)	bilstm_attention	backtranslate(balanced)	**0.8904**	0.8717	0.8550	0.9289
BiLSTM Attention (EDA+)	bilstm_attention	eda_plus	0.8819	0.8693	0.8952	0.8689
LSTM (Balanced)	lstm	backtranslate(balanced)	0.8782	0.8682	0.9128	0.8460
LSTM (EDA+)	lstm	eda_plus	0.8780	0.8683	0.9146	0.8443
LSTM (No Aug)	lstm	none	0.8763	0.8630	0.8880	0.8650
Simple RNN (EDA+)	simple_rnn	eda_plus	0.8646	0.8470	0.8588	0.8705

Experiment	Model	Augmentare	F1-Score	Acuratețe	Precizie	Recall
Simple RNN (Balanced)	simple_rnn	backtranslate(balanced)	0.8616	0.8418	0.8465	0.8773
BiLSTM Attention (No Aug)	bilstm_attention	none	0.8582	0.8514	0.9242	0.8010
Simple RNN (No Aug)	simple_rnn	none	0.8447	0.8315	0.8753	0.8161

7. Concluzii

Studiul a demonstrat impactul semnificativ al calității datelor și al arhitecturii asupra analizei sentimentului în limba română. Principalele concluzii sunt:

1. **"Data Quality is King" - Rolul Critic al Augmentării**:
    - **Echilibrarea prin Back-Translation** a fost factorul determinant pentru performanță, fiind singura metodă care a corectat bias-ul inerent al setului de date (dezechilibru 2:1 pozitiv/negativ). A crescut spectaculos **Recall-ul** pe clasa negativă, permițând modelului să identifice corect recenziile critice, un aspect esențial în aplicații reale.
    - **Modelele fără augmentare (No Aug)** au suferit de overfitting rapid și au manifestat un bias puternic spre clasa majoritară, obținând o precizie artificială mare dar eșuând în detectarea sentimentelor negative.
2. **Arhitectura BiLSTM cu Atenție - State of the Art**:
    - Acest model a obținut constant cele mai bune rezultate (**F1 ~89%**). Mecanismul de atenție i-a permis să "filtreze" zgomotul și să se concentreze pe cuvintele purtătoare de sentiment (adjective, adverbe specifice), oferind atât performanță cât și interpretabilitate.
3. **Investiția în Date vs. Investiția în Model**:
    - Un fapt notabil este că un model elementar (**Simple RNN**) antrenat pe date augmentate (Balanced/EDA+) a reușit să depășească modele mult mai complexe (**BiLSTM Attention**) antrenate pe date brute. Aceasta subliniază că, în NLP-ul modern pe seturi limitate (low-resource languages), efortul depus în curățarea și augmentarea datelor aduce un randament mai bun decât simpla creștere a complexității modelului.
4. **Stabilitate prin EDA+**:
    - Deși Back-Translation a adus cele mai mari câștiguri brute, augmentarea online prin EDA+ a funcționat ca un excelent mecanism de regularizare, producând modele mai robuste la variații lingvistice și prevenind overfitting-ul pe termen lung.