

UNIVERSITATEA POLITEHNICA BUCUREȘTI
FACULTATEA DE AUTOMATICĂ ȘI CALCULATOARE
DEPARTAMENTUL CALCULATOARE



PROIECT DE DIPLOMĂ

Generare de Povești din Fluxuri Vizuale

Dragoș-Gabriel Unguru

Coordonator științific:
Conf. dr. ing. Traian Rebedea

BUCUREȘTI

2021

CUPRINS

1	INTRODUCERE	3
1.1	Context	3
1.2	Problema	3
1.3	Obiective	4
1.4	Structura lucrării	4
2	MOTIVAȚIE	5
3	ABORDĂRI EXISTENTE	6
4	SOLUȚIA PROPUȘĂ	10
4.1	Structura de bază a modelului - Seq2Seq	10
4.2	Unitatea recurentă de bază	11
4.3	Codificatorul de imagini	14
4.4	Codificatorul text	19
4.5	Decodificatorul și modelul complet	22
5	DETALII DE IMPLEMENTARE	24
5.1	Tehnologii	24
5.2	Seturile de date	25
5.2.1	Prezentarea seturilor de date	25
5.2.2	Procesarea setului de date	26
5.3	Detaliile algoritmului de antrenare	27
5.4	Detaliile algoritmului de inferență	29
6	EVALUAREA REZULTATELOR	31
6.1	Metricile folosite	31
6.1.1	Scorul <i>METEOR</i>	31
6.1.2	Scorul <i>ROUGE</i>	32
6.2	Rezultate	32
7	CONCLUZII	39
8	BIBLIOGRAFIE	40
9	ANEXE	42

1. SINOPSIS

De-a lungul ultimilor ani, utilizarea algoritmilor de învățare automată în rezolvarea problemelor vizual-lingvistice a căpătat o atenție considerabilă. Deși s-au obținut rezultate excelente în probleme de descriere punctuală a imaginilor în izolare, generarea și detecția lucrurilor ieșite din domeniul concretului conturează o problemă mai puțin abordată și o provocare în domeniul rețelelor neuronale profunde. Diferite de simplele descrieri ale imaginilor, poveștile sunt compuse într-un stil lingvistic mult mai personal, mai dinamic. Mai mult decât atât, poveștile au adițional în scop și emoțiile și relațiile create de întreg ansamblul de imagini. Pentru abordarea unei astfel de probleme, propunem o rețea neuronală profundă ce are la bază o arhitectură de tip codificator-decodificator și ce îmbină un sistem de rețele recurente și convoluționale ce pleacă de la conceptul de învățare prin transfer de cunoștințe. Astfel, obținem o rețea de bază peste care experimentăm diverse arhitecturi de codificare în vederea comparării performanțelor acestora atât dintr-un punct de vedere subiectiv, critic, cât și din punctul de vedere al metricilor de evaluare automată. Mai mult decât atât, punem sub îndoială relevanța și aptitudinea acestor metrici pe baza rezultatelor obținute.

2. ABSTRACT

Over the past few years, the use of machine learning algorithms to solve numerous visual-linguistic problems has gained a lot of traction. Even though excellent results were achieved in image-in-isolation captioning, detecting and stating the entities that are not that obvious still represents an unexplored territory and a challenge for deep neural networks. Storytelling, different from the simple task of image captioning, is related in a much more personal and dynamic linguistic style. Moreover, stories take into account both emotions and the various relationships between images. In order to tackle such a problem, we put forward a deep neural network based on the encoder-decoder architecture, combining both recurrent and convolutional neural networks that start off the principle of transfer learning. Thus we obtain a base network over which we experiment with various encoding heuristics in order to compare not only the performance in both a subjective point of view and based on scoring metrics, but also question the relevance and suitability of these metrics based on the said results.

1 INTRODUCERE

În acest capitol ne dorim să detaliem motivația din spatele obiectivului propus alături de provocările impuse de aceasta, cât și țelul pe care îl avem în vizor. Capitolul se încheie cu o scurtă prezentare a capitolelor lucrării.

1.1 Context

De-a lungul ultimilor ani, învățarea automată a adus mari progrese în procesarea imaginilor și în generarea de text în limbaj natural cu ajutorul rețelelor neuronale profunde. În plus, domeniul ce le îmbină pe cele două a căpătat o atenție sporită odată cu stăpânirea și rafinarea rețelelor neuronale profunde.

Au fost obținute rezultate excelente în nenumărate probleme de acest tip precum: subtitrarea de imagini [1], traducerea automată [2], agenții conversaționali și răspunderea la întrebări [3] și multe altele. Deși obținând rezultate impresionante de-a lungul timpului pe astfel de probleme, capacitatea rețelelor neuronale de a capta o înțelegere asemănătoare omului este încă limitată.

1.2 Problema

În urmărirea obiectivului de a obține un algoritm de inteligență artificială capabil de a raționa cât mai asemănător cu un om, capacitatea de a asocia și relata concepte care nu sunt explicit vizibile rămâne o problemă deschisă.

Niciuna dintre problemele mai sus menționate nu abordează problema extragerii de caracteristici contextuale și legarea acestora într-o poveste coerentă. Din problemele mai sus menționate, subtitrarea de imagini se apropie de problema generării de povești cu o excepție majoră. Modelul prezentat își propune trecerea de bariera staticului și a descrierilor punctuale, lipsite de context, la propoziții coerente ce împreună formează o narațiune a evenimentelor petrecute, cu un puternic accent pe emoțiile din spatele acestora, a cronologiei și a tuturor aspectelor ce constituie o povestire.

Obținerea unui algoritm capabil de a extrage context semnificativ dintr-un flux de imagini, de a rezuma ideile, întâmplările și chiar emoțiile reprezintă una dintre cele mai interesante provocări vizual-lingvistice.

1.3 Obiective

Ne dorim obținerea unui model capabil de a genera propoziții descriptive cât mai lungi ce vor constitui povești cât mai bogate și de menținerea coerenței pe întreg corpusului generat.

Propunem o arhitectură ce îmbină mai multe rețele neuronale profunde ce vor aborda separat fiecare paradigmă ce stă la baza problemei.

Mai mult decât atât, după atingerea obiectivului funcțional al problemei dintr-un punct de vedere subiectiv, explorăm performanțele mai multor abordări derivate din această arhitectură. Avem astfel în vizor dobândirea unor rezultate remarcabile având la bază o implementare lipsită de euristici ce vizează pură creștere a valorilor metricilor de evaluare automată.

1.4 Structura lucrării

În capitolul 2 explorăm atât amănuntele ce desemnează un rezultat corect cât și o implementare corectă a problemei. Mai departe, în capitolul 3 facem cunoscute lucrările de specialitate actuale ce vizează aceeași problemă și au la bază același set de date. Tot în acest capitol ne familiarizăm cu lucrările semnificative din domeniu și aflăm poziționarea lucrării noastre în raport cu literatura de specialitate. Capitolul 4 abordează noțiunile teoretice din spatele arhitecturii prezentate și creează o imagine amănunțită a ansamblului acestuia efectuând o construcție pe plan vertical al modelului pe baza acestora. În capitolul 5 explorăm atât modalitățile prin care am implementat noțiunile prezentate în capitolul anterior cât și provocările apărute pe parcurs și rezolvările acestora. Capitolul 6 are rol de evaluare a implementării executate. Acest capitol are în vizor corectitudinea descrisă în capitolul 2 și efectuează o comparație atât cu alte abordări experimentate în această lucrare cât și cu lucrările prezentate în capitolul 3. Lucrarea se încheie cu capitolul 7 unde tragem concluzii pe baza a ceea ce am obținut și descoperit cu un capitol în urmă și cu o discuție finală legată de viitoare îmbunătățiri posibile.

2 MOTIVAȚIE

În decursul ultimilor ani, aplicații precum subtitrarea de imagini [1] sau detecția claselor de obiecte din imagini [4] reprezintă doar o mică parte a algoritmilor care ne influențează viața de zi cu zi. Obținând rezultate uluitoare în astfel de probleme, nenumărate industrii au accelerat.

Cu toate acestea, problema propusă reprezintă în continuare un subiect deschis și conturează primii pași către obținerea aceluiași algoritm de învățare automată capabil de înțelegerea substratului, de detectare și exprimare a lucrurilor care nu sunt vizibile, capabil de extragerea și chiar construirea de context.






	1	2	3	4	5
					
Desc-in-Isolation	A black frisbee is sitting on top of a roof.	A man playing soccer outside of a white house with a red door.	The boy is throwing a soccer ball by the red door.	A soccer ball is over a roof by a frisbee in a rain gutter.	Two balls and a frisbee are on top of a roof.
Desc-in-Sequence	A roof top with a black frisbee laying on the top of the edge of it.	A man is standing in the grass in front of the house kicking a soccer ball.	A man is in the front of the house throwing a soccer ball up.	A blue and white soccer ball and black Frisbee are on the edge of the roof top.	Two soccer balls and a Frisbee are sitting on top of the roof top.
Story-in-Sequence	A discus got stuck up on the roof.	Why not try getting it down with a soccer ball?	Up the soccer ball goes.	It didn't work so we tried a volley ball.	Now the discus, soccer ball, and volleyball are all stuck on the roof.

Figura 1 Exemplu de rezultat așteptat în comparație cu clasicele probleme de subtitrare a imaginilor¹

În figura 1 este ilustrată caracteristica ce deosebește problema generării de povești de simplele probleme de recunoaștere punctuală a obiectelor. Descrierile imaginilor în izolare (DII) vizează descrierea individuală a obiectelor și a acțiunilor prezentate în fiecare imagine. Cu cât ne apropiem de descrierea poveștilor în secvență, putem observa apariția mult mai multor cuvinte cu caracter personal ("we"), o frecvență crescută a cuvintelor ce descriu un peisaj temporal mai concret ("now"), cât și o formulare la un timp specific istorisirii. Toate aceste caracteristici sunt definite de conceptul de narațiune, textele descriptive transformându-se în povești cu dependențe foarte strânse de propozițiile și fotografiile anterioare.

Reușita antrenării unei rețele neuronale capabile de a simula un comportament uman ne-ar putea croi calea către acel algoritm de învățare automată capabil de a învăța lucruri pe care le considerăm posibile doar pentru om, lucru care ar aduce doar beneficii într-o varietate largă de industrii.

¹ @<https://visionandlanguage.net/VIST/>, accesat la (21-06-2021)

3 ABORDĂRI EXISTENTE

Huang et al. [5] pun bazele problemei prin introducerea unui nou set de date ce constituie fundamentul temei. De la introducerea acestui set de date până în prezent, problema generării de povești din date vizuale a căpătat o atenție considerabilă, fapt ce a condus cercetarea în acest domeniu către o multitudine de abordări.

De asemenea este introdusă și o implementare ce va pune fundația viitoarelor abordări alături de o metodologie automată de evaluare a performanțelor.

Atât implementarea cât și modalitatea de evaluare automată propusă de cei menționați anterior este inspirată din succesul arhitecturilor ce stau la baza problemelor de traducere automată [1].

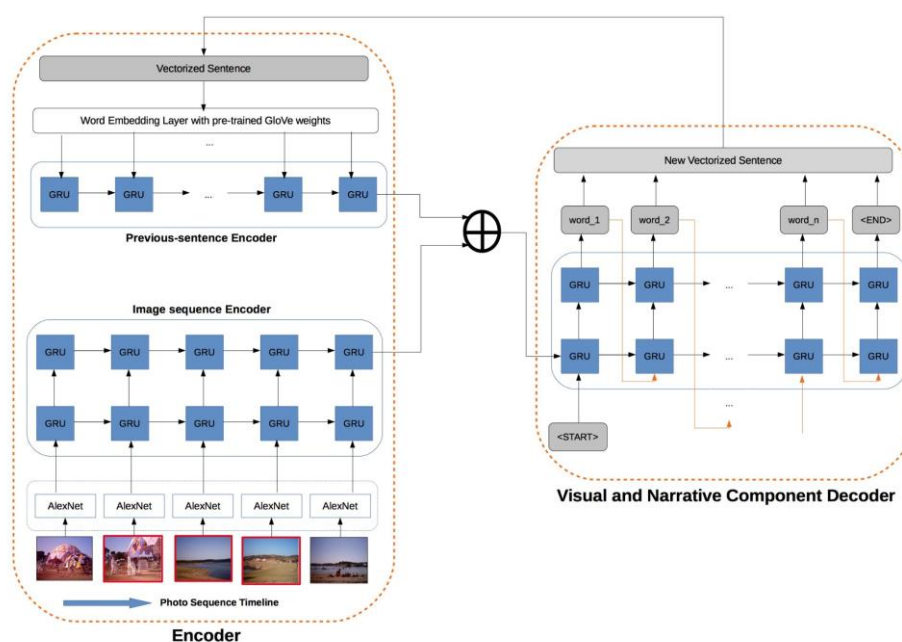


Figura 2 Modelul de bază propus pentru problema generării de povești din date vizuale de către cercetătorii din cadrul *Microsoft*²

Obținând rezultate remarcabile în nenumărate probleme de procesare de limbaj natural, Huang et al. [5] adaptează arhitectura *Model Secvențial către Model Secvențial* (cunoscută sub numele *Seq2Seq*) și o aplică cu succes în problema generării de povești pe baza datelor vizuale. Deși obțin rezultate admirabile folosind o astfel de arhitectură, lucrarea pune în centrul atenției performanțele obținute cu ajutorul mai multor euristici, îndreptând efortul către optimizarea performanțelor descrise de metricile de evaluare automată.

² @<https://github.com/Pendulibrium/ai-visual-storytelling-seq2seq>, accesat la (21-06-2021)

Drept experiment, în lucrarea ce a introdus setul de date [5] se introduce modelul *Greedy* ce are la bază exact aceeași arhitectură descrisă în figura 2. Singura adăugare este reprezentată de simpla euristică de eliminare a propozițiilor consecutive ce se repetă. O euristică atât de simplă a adus un câștig de performanță de aproximativ 8 procente din punctul de vedere al metricilor.

Acest focus a împins dezvoltarea și axarea multor lucrări spre optimizarea acestor euristici, alocând mai puțin timp pe rafinarea modelului ce stă la baza generării de povești. Un astfel de exemplu este prezentat mai departe de către Huang et al. [6] unde propun antrenarea unui model asemănător dar folosind punctual caracteristicile claselor de obiecte detectate în imagini.

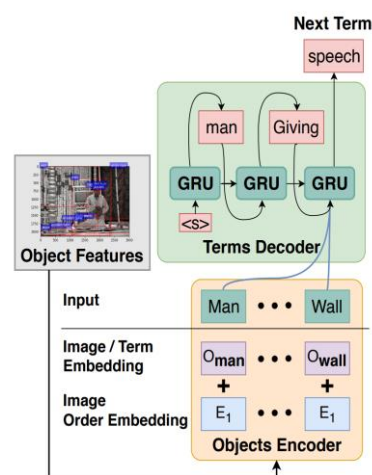


Figura 3 Mecanismul de bază ce descrie euristica folosită, descris sub forma *Faza de Distilare* [6]

Acest lucru este obținut prin folosirea unei rețele neuronale convoluționale axată pe regiuni, rețeaua *Faster R-CNN* [7] pre-antrenată pentru predicția obiectelor din imagini. Adicional, Huang et al. [6] dezvoltă un mecanism de grupare și corelare a cuvintelor la categorii specifice de obiecte sau acțiuni pentru fiecare imagine individuală din album. Acest pas este descris sub denumirea de *Faza de Distilare*. Un astfel de mecanism este obținut prin antrenarea unui model separat de restul, pe un set de date corespunzător problemei.

Astfel de euristici cu siguranță au succes în maximizarea metricilor de evaluare automată. Acest lucru se datorează faptului că aceste metrice au la bază analiza pur sintactică a rezultatelor iar astfel de euristici maximizează probabilitatea punctării exacte a obiectelor vizibile în imagini, fapt ce totuși nu garantează o poveste bogată în context, ci chiar apropierea abordării spre problemele de subtitrare de imagini. Caracteristica aceasta vine natural fiind dată natura complexă a narațiunii și multitudinea trăsăturilor acestora.

AREL [8] adresează direct problema prezentată de axarea pe astfel de metrici și critică riguros capacitatea acestora de a reflecta performanța. Mai mult decât atât, Xin Wang et al. [8] propun probabil cea mai avansată interpretare a problemei, mutând totodată focusul înapoi către dezvoltarea unui model capabil de a învăța să genereze povești bogate în context fără ajutorul euristicilor. În mod impresionant, în lipsa interesului pur de maximizare a valorilor metricilor de evaluare automată, AREL [8] obține de asemenea și cel mai impresionant punctaj pe o mare varietate de astfel de metrici, lucru ce poate fi observat și în tabelul 1.

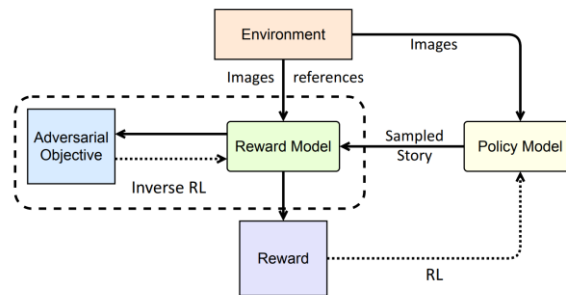


Figura 4 Arhitectura abstractă a modelului AREL [8]

Wang et al. [8] introduc un model bazat pe noțiuni sofisticate de rețele generative adversariale (GAN) și extind lucrarea cu un model de învățare automată pe bază de recompense. Cu toate acestea, la baza acestui model se află sub numele de *Policy Model* o rețea *Seq2Seq* la care ne putem raporta.

Deși prezintă un model complex și dificil de antrenat, acest cost este mai mult decât justificat. Însuși modelul de bază al AREL [8], modelul descris de simpla arhitectură *Seq2Seq*, obține cele mai bune rezultate pe o varietate largă de metrici.

Tabelul 1 Evaluarea performanțelor modelelor prezentate din punct de vedere al metricilor de evaluare automată

	Modelul de bază propus de Huang et al. [1]		Knowledge- Enriched Visual Storytelling [2]	AREL [3]		
Model/E uristică	Fără euristică	Greedy	Visual Genome	Policy Model	GAN	AREL
METEOR	0.277	0.301	0.296	0.348	0.350	0.350
ROUGE	-	-	0.241	0.297	0.295	0.295
BLEU	-	-	0.451	0.623	0.628	0.638

În tabelul 1 vizualizăm superioritatea modelelor propuse în lucrarea AREL [8] și ni se justifică utilizarea unor modele mai complexe decât cele propuse de autorii lucrării originale.

Însă mai important decât atât, putem observa performanțele remarcabile obținute chiar de modelul *Seq2Seq* al acestei lucrări, model sub numele *Policy Model*. Acest model nu doar

obține cel mai bun rezultat pe una dintre metrice, dar prezintă și o performanță aproximativ egală în comparație cu restul arhitecturilor mai complexe.

Aceste rezultate ne demonstrează faptul că există posibilitatea obținerii de modele mai mult decât capabile folosind o rețea neuronală de tip codificator-decodificator.

Drept urmare, în prezenta lucrare ne propunem abordarea problemei prin intermediul unei astfel de arhitecturi. Plecând de la o abordare similară modelului de bază propus de Huang et al. [5], ne propunem construirea unei rețele neuronale profunde de tipul *Seq2Seq*.

Avem în vizor obținerea de povești cât mai bogate în context și cât mai lungi, maximizarea metricilor de evaluare automată reprezentând obiectivul secundar. Drept urmare, explorăm rafinarea unui astfel de model făcând abstracție de orice formă de euristică ce are ca scop pură creștere a acestor metrice.

În fază incipientă ne dorim o slabă menținere a similarității cu modelul propus de Huang et al. [5], similaritate menținută strict din punct de vedere arhitectural. Inițial propunem o schimbare a unităților recurente de bază, a rețelelor ce stau la baza codificării și a metodei de îmbinare a acestora. Mai departe explorăm metode diferite de codificare prin inspirarea din caracteristicile definitorii ale rețelei de bază AREL [8].

Un obiectiv final important îl reprezintă examinarea capacității acestor metrice folosite în evaluarea performanțelor. Comparăm astfel performanța mai multor modele obținute din punct de vedere subiectiv, în paralel cu concluziile aduse de valorile metricilor.

Așadar, în prezenta lucrare descriem o abordare a problemei folosind o nouă rețea neuronală profundă recurentă pentru generarea de povești coerente, o rețea *Seq2Seq* care demonstrează faptul că putem atinge obiectivul principal fără utilizarea rețelelor complexe de învățare prin recompensă și fără crearea și folosirea diverselor euristici ce forțează un anumit comportament constrâns al rețelei în timpul inferenței.

4 SOLUȚIA PROPUȘĂ

Ne propunem extinderea studiului problemei și continuarea lucrării originale [5] prin introducerea unei noi rețele neuronale recurente, o nouă abordare folosind un model de tipul Sequence-to-Sequence (Seq2Seq).

În subcapitolele următoare ne propunem detalierea construcției modelului ales pe plan vertical, urmărirea provocărilor întâlnite pe parcurs și analiza avantajelor și dezavantajelor diverselor abordări, plecând de la unitățile care stau la baza construcției și ajungând la descrierea modelului ca ansamblu.

4.1 Structura de bază a modelului - Seq2Seq

Introdusă de Sutskever et al. [9], această arhitectură a obținut rezultate excelente în rezolvarea problemelor ce presupun procesarea datelor secvențiale. Popularitatea acestei arhitecturi se datorează obținerii multiplelor rezultate remarcabile într-o varietate largă de probleme asemănătoare precum subtitrare de imagini [1] și traducere automată [2].

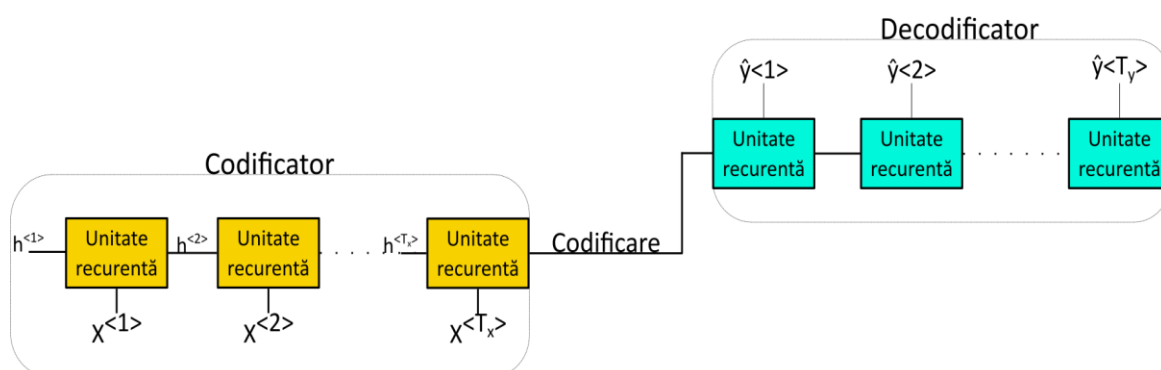


Figura 5 Model tipic Seq2Seq bazat pe codificator și decodificator

Probabil cel mai mare avantaj al unui astfel de model este reprezentat de faptul că această arhitectură oferă libertatea de a manipula dimensionalitatea datelor. Spre deosebire de o singură rețea neuronală recurentă, care leagă o intrare la o ieșire, arhitectura Seq2Seq permite corelarea unei secvențe de intrări de lungime T_x la o secvență de ieșire de o dimensiune oarecare dorită T_y .

Un model tipic Seq2Seq constă în două părți:

- codificatorul, un model care are ca scop învățarea unei funcții de codificare a unei secvențe de date și aducerea acestei secvențe într-o dimensionalitate redusă.
- decodificatorul, un model care primește valorile de ieșire a codificatorului și are ca scop învățarea generării unei secvențe care poate reprezenta obiectivul dorit.

Modelul tipic descris nu satisface în întregime cerințele problemei generării de povești dintr-un flux de imagini. Având de-a face cu necesitatea de a lua decizii atât pe baza imaginii curente cât și a textului descriptiv ale imaginilor precedente pentru a genera noua predicție, avem nevoie de o arhitectură care poate combina codificările atât ale imaginilor cât și ale textelor precedente.

4.2 Unitatea recurentă de bază

O primă majoră decizie în implementarea unui astfel de model este reprezentată de alegerea unei unități recurente care să stea la baza arhitecturii acesteia.

Unitățile recurente tradiționale sunt folosite cu mare succes în construirea modelelor ce procesează date secvențiale. Acestea au însă un dezavantaj atunci când este nevoie de o rază lungă de acțiune. Acestea eșuează să mențină dependențe și să conecteze informația, având în final doar influențe locale. În vederea generării de povești cu relații strânse între contextele definite de imagini, această dependență este crucială deoarece avem ca scop nu doar vizarea cuvintelor apropiate următorului cuvânt care trebuie generat, ci tot restul propozițiilor prezise până în acel punct.

Aceste unități pot fi privite ca niște simple straturi de activare, pentru date de intrare de tip secvențial, aplicate pe activările specifice fiecărei unități de timp.

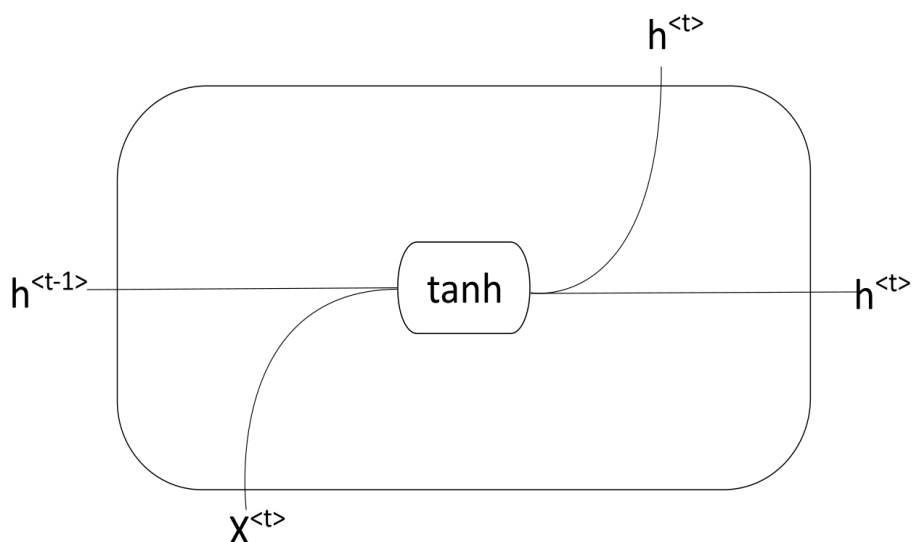


Figura 6 Celulă recurentă fără porți

Unitățile recurente fără porți mențin doar activările celulei. Acestea sunt reprezentate de o sumă ponderată peste care se aplică o funcție non-lineară.

$$h_t = f(W_h h_{t-1} + W_x x_t) \quad (1)$$

unde f este o funcție non-lineară oarecare, h_t reprezintă starea ascunsă a celulei la timpul t , x_t intrarea celulei la timpul t , iar W matricea ponderilor.

Având secvențe lungi de generat, aceste stări ajung foarte rapid să convergă la o valoare infimă, apropiată de 0, la aplicarea repetată a *Backpropagation* a algoritmilor de optimizare bazați pe algoritmul Gradient Descent.

$$h_t = f(W_x x_t + W_h h_{t-1}) \quad (2)$$

Acest lucru va conduce nu doar la dependențe exponențial mai mici odată cu creșterea lungimii secvenței dar și la problema *Vanishing Gradients*, unde ponderile rețelei vor ajunge la 0 în încercarea minimizării erorii. Cum ne dorim să prezicem probabilități, avem nevoie de funcții de activare care să ne ofere o distribuție a acestora, iar având

$$\lim_{n \rightarrow \infty} W^n = 0, \quad W \in [0, 1) \quad (3)$$

, acest lucru va permite rețelei să învețe toate ponderile ca fiind 0.

Pentru a rezolva această problemă, de-a lungul ultimelor două decenii, s-au rafinat și introdus unitățile recurente cu porți [10]. Mai departe ne vom referi la LSTM ca implementarea care folosește conexiuni cu vizer pentru accesul la celulele de memorie a acestora, abordare propusă de Gers, Schmidhuber și Cummins în 2000 [11].

În marea majoritate a studiilor efectuate în problema generării de povești, atât în modelul de bază al AREL [8] cât și în modelul inițial propus de către Huang et al. [5], observăm că *Gated Recurrent Unit* (GRU) este alegerea populară ca unitate recurentă dominantă. Așadar, având ca scop explorarea și testarea diverselor abordări față de cele existente, o primă diferență majoră pe care o propunem este bazarea rețelei recurente pe unități de tip *Long Short Term Memory* (LSTM) [11].

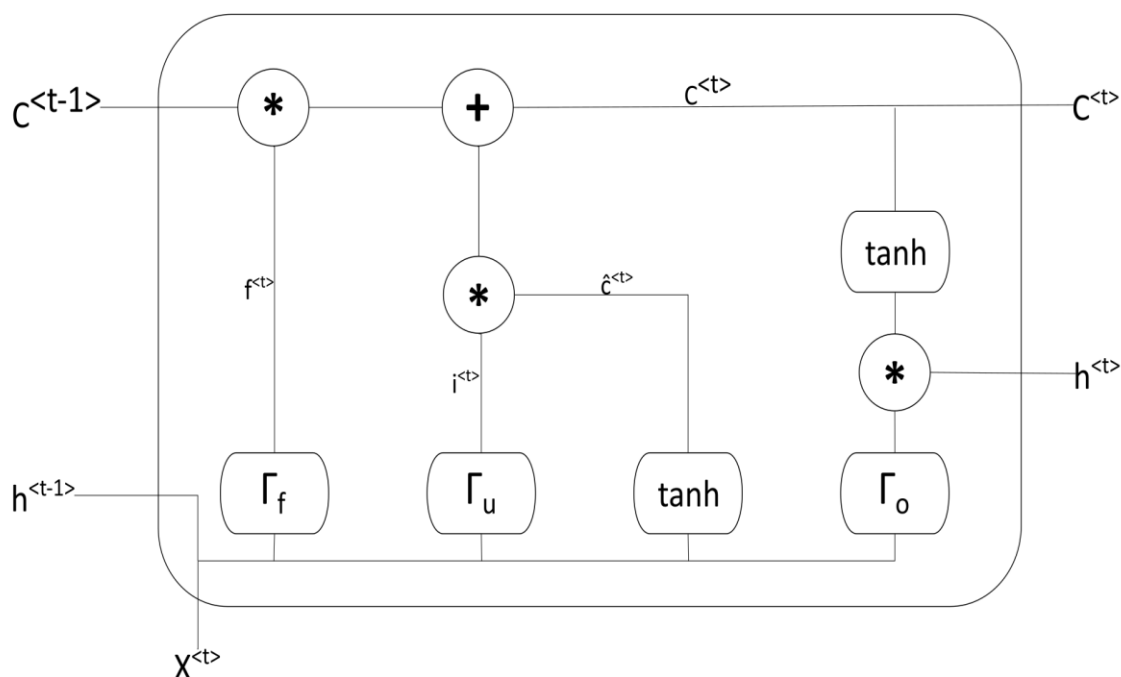


Figura 7 Celulă recurentă de tip LSTM

Fiecare celulă LSTM menține o memorie c_t pentru fiecare unitate de timp t a secvenței. Această memorie este folosită la fiecare moment t pentru a modela contribuția secvențelor anterioare pentru momentul curent. Așadar, valoarea activării se calculează ca

$$h_t = \Gamma_o \cdot \tanh(c_t) \quad (4)$$

de Γ_o reprezintă *poarta de ieșire*. Această poartă controlează cantitatea de expunere a celulei la conținutul lui c_t . Aceasta se calculează ca

$$\Gamma_o = \sigma(W_o[h_{t-1}, x_t] + c_t) \quad (5)$$

Celula de memorie c_t este actualizată folosindu-se de *poarta de actualizare* și de *poarta de omitere*.

$$c_t = \Gamma_u \hat{c}_t + \Gamma_f c_t \quad (6)$$

Poarta de actualizare (sau factorul de actualizare) este folosită pentru a modula importanța noii celule de memorie candidat \hat{c}_t la fel cum *poarta de omitere* (sau factorul de omitere) modulează importanța memorii existente c_t . Noua celulă de memorie candidat este calculată ca

$$\hat{c}_t = \tanh(W_c[h_{t-1}, x_t]) \quad (7)$$

Cei doi factori care modulează importanța celulei de memorie și celulei candidat se calculează similar cu *poarta de ieșire*

$$\Gamma_u = \sigma(W_u[h_{t-1}, x_t] + c_{t-1}) \quad (8)$$

$$\Gamma_f = \sigma(W_f[h_{t-1}, x_t] + c_{t-1}) \quad (9)$$

Deși similare, unitățile recurente Long Short Term Memory [11] vin cu o complexitate ridicată față de Gated Recurrent Units, LSTM prezentând acest mecanism de actualizare a memoriei la fiecare unitate de timp t . GRU actualizează *valorile ascunse* folosind o interpolare liniară a valorii precedente și a valorii candidat. Ambele abordări reușesc să rezolve problema *Vanishing Gradients* printr-un mecanism de a uita și memora dependențele necesare.

Odată decisă unitatea de bază a arhitecturii modelului, mai departe putem începe să detaliem construcția rețelei.

4.3 Codificatorul de imagini

În vederea creării modelului nostru, ne folosim de o rețea neuronală convoluțională pentru a putea extrage caracteristici din imagini, pe care ulterior le vom codifica. Așadar, la baza codificatorului nostru de imagini va sta o rețea neuronală convoluțională pre-antrenată.

Rețelele neuronale convoluționale au primit o atenție considerabilă odată cu modelele în stil LeNet [12], iar astăzi stau la baza rezolvărilor problemelor de interpretare a imaginilor în literatura de specialitate a învățării automate. LeNet [12] propune utilizarea a trei operații primordiale care să stea la baza unei rețele convoluționale:

- straturi de convoluție, folosite cu filtre de valori antrenabile. Acest lucru face posibilă extragerea și detecția de diferite caracteristici având complexitate progresivă.
- straturi de *pooling*, care au un efect de concentrare și extragere a caracteristicilor importante găsite în urma unei operații de convoluție.
- straturi complet conectate, stratul de bază al unei rețele neuronale cu un anumit număr dat de neuroni.

După o îndelungată rafinare a acestei familii de rețele, ia naștere GoogLeNet (cunoscută și sub denumirea de Inception V1) care pune bazele arhitecturii moderne Xception [4].

Xception [4] este fără îndoială una dintre cele mai populare rețele convoluționale actuale³. Cunoscută și sub denumirea *Extreme Inception*, această arhitectură obține rezultate excelente în detecția multiplelor clase de obiecte în imagini.

Așadar, această rețea se prezintă ca un candidat excelent pentru aplicația noastră, combinând atât acuratețe remarcabilă cât și un număr redus de caracteristici față de alte rețele.

³ <https://towardsdatascience.com/convolutional-neural-networks-most-common-architectures-6a2b5d22479d>

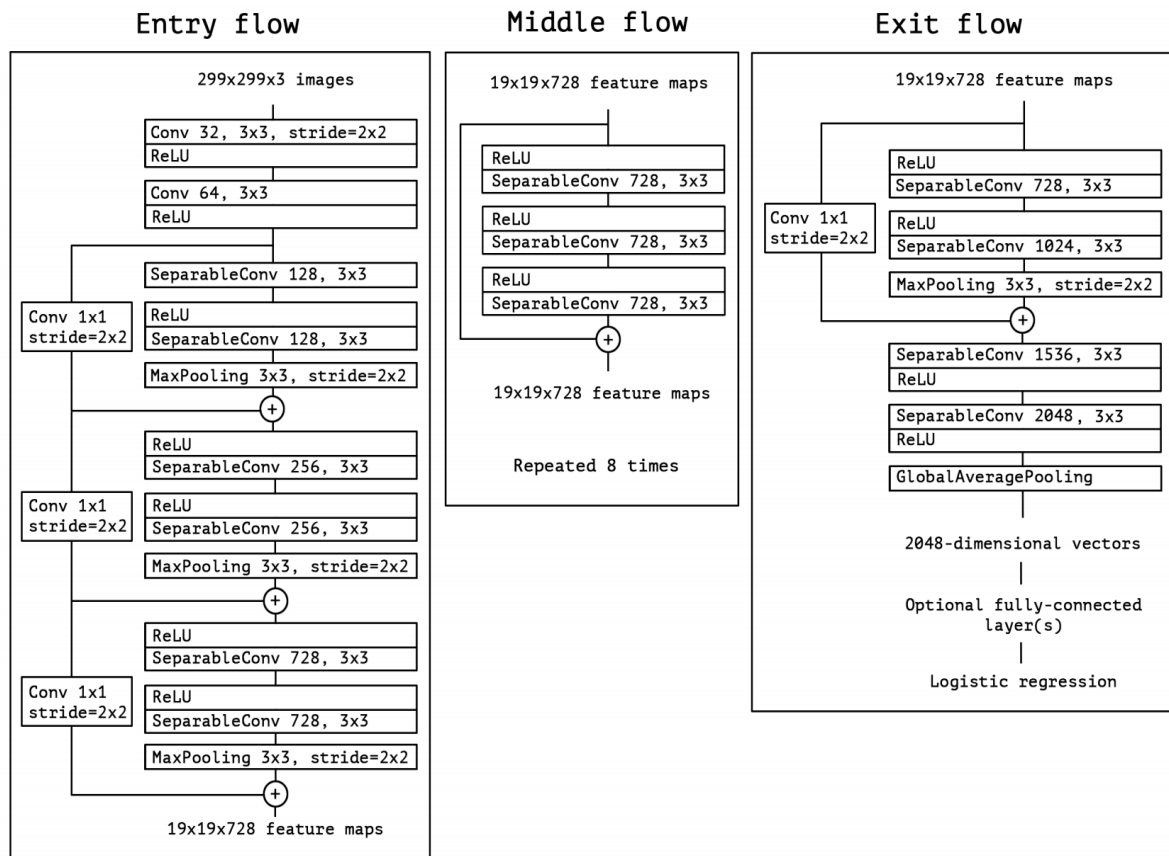


Figura 8 Arhitectura modelului Xception [4]

Acest model urmărește cunoscutul tipar de repetare a aceleiași operații convoluționale, crescând treptat numărul de canale. Acest tipar nu doar a obținut rezultate excelente comparativ cu celelalte modele propuse înaintea Xception [4], dar a obținut și reducerea considerabilă a adâncimii rețelei alături de numărul de hiper-parametri care necesită atenție specială.

În continuare, ne propunem să explorăm atât caracteristicile principale ale rețelei Xception [4] cât și motivațiile din spatele deciziilor arhitecturale ale acesteia.

Probabil cel mai important rol este jucat de stiva de straturi convoluționale descrisă în figura 8 ca *SeparableConv*, cu ajutorul căreia reușește să obțină atât o acuratețe competitivă cât și o eficiență notabilă împreună cu un număr redus de caracteristici și straturi.

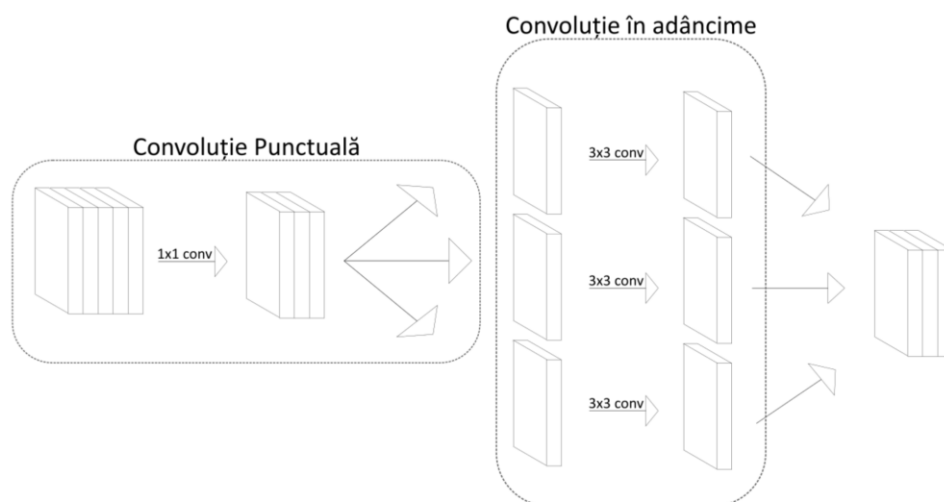


Figura 9 Stiva de straturi *SeparableConv*

Figura 9 detaliază construcția care stă la baza rețelei convoluționale. Se aplică operația de convoluție separabilă în profunzime, adică o convoluție spațială efectuată independent peste fiecare canal al datelor de intrare, precedată de o convoluție punctuală, adică o operație de convoluție cu un filtru de dimensiune 1×1 .

Această operație care stă la baza modelului Xception [4] asigură o lipsă totală a aplicării funcțiilor non-lineare, lucru care a fost dovedit experimental [4] să aducă un plus de acuratețe în problema clasificării claselor de obiecte din imagini⁴ pe o rețea mai puțin adâncă precum aceasta.

Având în vedere marile adâncimi ale rețelelor neuronale convoluționale, nu este neobișnuită apariția problemei *Vanishing Gradients* (sau, mai rar, chiar *Exploding Gradients*), iar modelul folosit nu este vreo excepție de la această problemă. Conexiunile reziduale sunt deseori introduse în rețelele neuronale convoluționale profunde ca soluție a acestei probleme. Observăm în figura 8 că Xception [4], o rețea relativ mai puțin adâncă față de restul rețelelor din aceeași clasă, nu le omite, implementând conexiuni reziduale aditive.

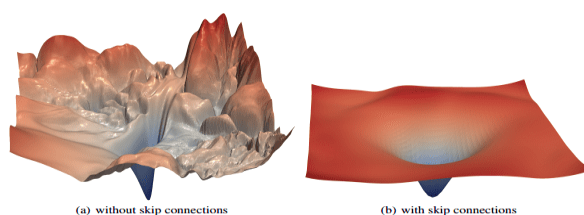


Figura 10 Vizualizare a suprafeței erorii în modelul ResNet-56 cu și fără conexiuni reziduale [13]

⁴ Peste setul de validare al concursului anual ILSVRC pentru un număr de 1000 de clase

Motivația din spatele conexiunilor reziduale este ca acestea să asigure o modalitate alternativă de actualizare a fluxului gradientilor în faza de *Backpropagation* a algoritmilor de optimizare bazați pe *Gradient Descent*.

În figura 10 putem vizualiza suprafața erorilor a aceluiași model cu și fără conexiuni reziduale. Având o complexitate ridicată, rețelele neuronale convoluționale prezintă o suprafață de eroare mult mai complexă, introducând astfel riscul ca algoritmi de optimizare să eșueze să converge sau chiar să converge la un minimum local. Introducând conexiuni reziduale, putem observa cum suprafața erorii se transformă într-o suprafață convexă, eliminând posibilitatea de a converge la o valoare minimă locală.

Mai mult decât atât, aceste conexiuni permit reutilizarea caracteristicilor de aceeași dimensionalitate extrase în straturile anterioare ale modelului și pasarea acestora mai departe în model, fără să aibă vreun impact în puterea rețelei de a generaliza.

Toate aceste beneficii vin cu, virtual, niciun cost, așa cum putem demonstra mai departe cel mai defavorabil caz.

Considerăm o rețea neuronală oarecare cu o conexiune reziduală din stratul l către stratul $l + 2$. Fără această conexiune, activările stratului $l + 2$ ar avea forma

$$h^{[l+2]} = f(z^{[l+2]}) \quad (10)$$

unde f reprezintă o funcție non-lineară oarecare, iar

$$z^{[l+2]} = W^{[l+2]}h^{[l+1]} + b^{[l+2]} \quad (11)$$

unde $W^{[l+2]}$ reprezintă ponderile antrenabile ale conexiunii dintre stratul $l + 1$ și stratul $l + 2$, iar b reprezintă unitatea de *bias*.

Introducând conexiunea reziduală între stratul l și stratul $l + 2$, activarea devine

$$h^{[l+2]} = f(z^{[l+2]} + h^{[l]}) \quad (12)$$

Ne vom referi la $h^{[l]}$ ca *blocul rezidual*. Așadar, activarea stratului $h^{[l+2]}$ poate fi scrisă ca

$$h^{[l+2]} = f(W^{[l+2]}h^{[l+1]} + b^{[l+2]} + a^{[l]}) \quad (13)$$

Vom asuma cel mai rău caz unde rețeaua neuronală suferă de *Vanishing Gradients* iar aceasta tinde spre a învăța toate ponderile 0. Așadar, vom avea

$$W^{[l+2]} = 0 \quad (14)$$

$$b^{[l+2]} = 0 \quad (15)$$

În cazul acesta, activarea va deveni

$$h^{[l+2]} = f(a^{[l]}) \quad (16)$$

Obținând în final nimic altceva decât o variantă mai simplistă a aceleiași rețele.

Așa cum am precizat anterior, Xception [4] va sta la baza codicatorului nostru de imagini, ajutându-ne să extragem caracteristici esențiale din acestea. Cum vom folosi o variantă pre-antrenată, este necesar să eliminăm straturile responsabile de clasificarea în cele 1000 de clase, modelul oferindu-ne astfel o codificare de 2048 de caracteristici pe imagine.

Tot ce mai rămâne de făcut pentru a construi codicatorul nostru de imagini este să introducem un strat de unități recurente peste modelul mai sus descris. Facem acest lucru în vederea obținerii unei codificări reprezentate de valorile ascunse h ale acestora și care vor fi mai departe transmise către decodicator. Acest strat va reprezenta partea antrenabilă a decodicatorului de imagini.

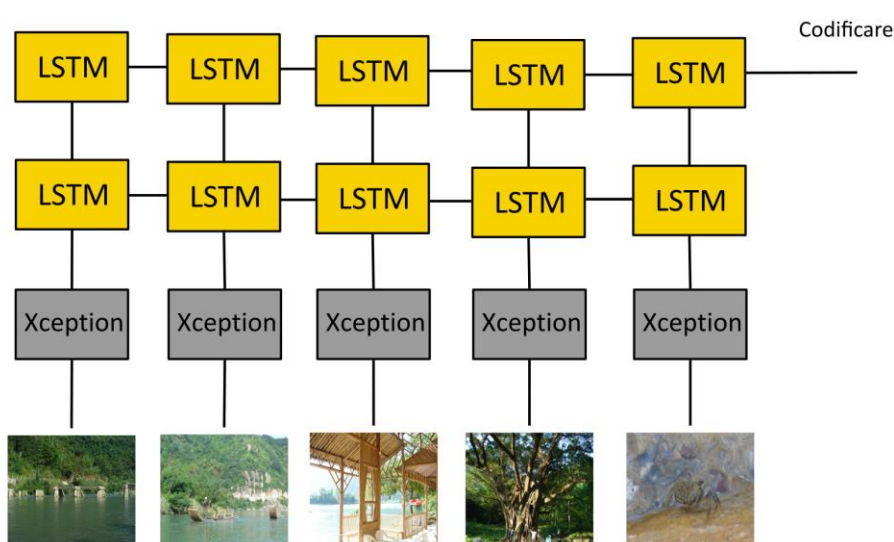


Figura 11 Codicatorul de imagini

În figura 11 putem vizualiza modelul care va fi folosit pentru codificarea secvențelor de imagini, înainte de a fi transmise, împreună cu codificările textelor anterioare, către decodicator.

Modelul prezentat este responsabil atât de extragerea caracteristicilor din imagini cât și de învățarea unei funcții de codificare f capabile să genereze un limbaj intermediar între codicator și decodicator.

În modelul prezentat vom păstra modelul Xception [4] ca fiind neantrenabil, având astfel o codificare deterministă ca intrare în submodelul construit peste acesta, partea antrenabilă fiind formată din celule LSTM.

Conectând doar un singur codicator care acționează pe imagini la decodicator ar putea rezulta un model capabil de a învăța să clasifice clase de obiecte. Avem ca obiectiv folosirea acestor clase de obiecte ce apar de-a lungul unei secvențe de imagini în vederea construirii

unei secvențe de propoziții care să nareze o poveste descriptivă, creând un context și un scenariu pe baza acestora.

Așadar, un codificator de imagini nu este de ajuns pentru problema noastră. Decodificatorul trebuie de asemenea legat la un codificator text, un codificator capabil să conecteze dependențe din propoziții anterioare și să secvențializeze textul la nivel de propoziție.

4.4 Codificatorul text

Pentru ca rețeaua neuronală să poată învăța dependențe la nivel semantic între cuvinte, avem nevoie de descrierea acestora prin prisma unui set de caracteristici. Ca rezolvare a acestei probleme, mai departe vom introduce noțiunea de *word embeddings*.

Word embeddings se referă atât la reprezentarea textului prin intermediul unor valori învățate, cât și la o clasă de tehnici care au în vizor asocierea cuvintelor la un vector de valori reprezentative. Ne vom referi la vectorul acestor valori reprezentative ca e_w , iar la matricea ce cuprinde toți acești vectori a tuturor cuvintelor din vocabularul folosit E . O_w reprezintă valoarea *one-hot* pentru cuvântul reprezentat de indexul în vocabular w . Cunoscând aceste notații, putem descrie vectorul de *embeddings* pentru un cuvânt cu ecuația

$$e_w = EO_w \quad (17)$$

unde matricea E este de forma

$$E = [e_1 | e_2 | e_3 | \dots | e_n] \quad (18)$$

Algoritmii de învățare ai acestor vectori de valori au început cu o euristică simplă: ne ghidăm după cuvintele apropiate sintactic, într-o fereastră de o lungime fixă.

Limitările acestei metode au fost atinse rapid. Complexitatea semantică a limbajului natural nu poate fi redusă la o euristică atât de simplă. Aceasta se bazează, în marea parte a timpului, pe dependențe semantice între cuvinte mult mai îndepărtate față de cât ar putea reprezenta o lungime a ferestrei care poate fi considerată rezonabilă din punct de vedere al eficienței. De asemenea trebuie menționat faptul că în această abordare ponderile cuvintelor uzuale (precum "și", "cu", "un") vor avea o pondere considerabil mai mare, eșuând astfel în obținerea unei distribuții uniforme a *embeddings*.

Algoritmul Vectorilor Globali (*GloVe*) [14] propune folosirea relației simetrice între două cuvinte, un context c și o țintă t , plecând de la ideea proximității în cadrul unei ferestre. Acest algoritm are la bază motivația că putem deriva relații semantice între cuvinte folosindu-ne de matricea de co-apariție.

Tabelul 2 - Exemplu de matrice de co-apariție cu lungimea ferestrei de 1 pentru propoziția "Beau un ceai de flori de tei"

	beau	un	ceai	de	flori	tei
beau	0	1	0	0	0	0
un	1	0	1	0	0	0
ceai	0	1	0	1	0	0
de	0	0	1	0	1	1
flori	0	0	0	1	0	0
tei	0	0	0	1	1	0

Matricea de co-apariție are ca scop reprezentarea frecvenței fiecărui cuvânt din vocabular în contextul celorlalte cuvinte. După cum putem observa în tabelul 2, matricea de co-apariție este simetrică. Exploatând această proprietate, putem extrage o metrică ce poate măsura similaritatea semantică dintre cuvinte.

Notăm astfel x_{tc} ca fiind numărul de apariții al unui cuvânt t în contextul lui c . Știind din tabelul 2 că această relație este simetrică, dacă cuvântul t este în proximitatea lui c , proximitatea fiind definită de lungimea ferestrei, putem spune că

$$x_{ij} = x_{ji} \quad (19)$$

Modelul GloVe [14] are ca obiectiv minimizarea după W a ecuației

$$L = \sum_{i=1}^{|voc|} \sum_{j=1}^{|voc|} f(x_{ij})(W_i^T e_j + b_i + b_j - \log(x_{ij}))^2 \quad (20)$$

unde b_i și b_j reprezintă unitățile de *bias* ale cuvintelor t respectiv c , iar $f(x_{ij})$ poate fi referită ca *termenul de ponderare* și este folosit pentru stabilitate numerică. Această funcție trebuie să prezinte trei caracteristici principale

- $f(x)$ tinde rapid spre 0 astfel încât să ofere stabilitate numerică pentru $x_{ij} = 0$, așadar

$$\lim_{x \rightarrow 0} f(x) = 0 \quad (21)$$

$$a. \hat{.} f(x) \log^2(x) = k \quad (22)$$

- $f(x)$ nu este descrescătoare pentru a nu supra-penaliza ponderile cuvintelor cu frecvență rară în favoarea cuvintelor comune.
- $f(x)$ ia valori relativ mici pentru valori foarte mari ale lui x pentru a nu crea ponderi foarte mari pentru cuvintele cu o frecvență mare a co-aparițiilor.

În efortul de a minimiza ecuația 20, GloVe [14] obține vectorii de valori descriptive pentru fiecare cuvânt din vocabular sub forma:

Tabelul 3 - Exemplu de vectori de valori ale *embeddings*

	Bărbat	Femeie	Rege	Regină
Gen	-1	1	-0.96	0.97
Roial	0.01	0.02	0.94	0.99
Epocă	0.03	0.02	0.71	0.68
Alimentar	0.09	0.01	0.02	0.01

În tabelul 3 putem vizualiza un exemplu simplificat al unor *embeddings* învățate de algoritmul GloVe [14]. Caracteristicile afișate în prima coloană reprezintă caracteristicile pe baza cărora algoritmul a învățat să reprezinte cuvintele vocabularului. În realitate, aceste caracteristici sunt mult mai abstracte, clasele prezentate în Tabelul 3 fiind exagerate. Așadar, având vectorii e învățați, acum putem deduce relații între cuvinte aplicând operații aritmetice peste vectorii din spațiul de *embeddings*. Avem ca exemplu, pe baza Tabelului 3, ecuația

$$e_{\text{bărbat}} - e_{\text{femeie}} \simeq e_{\text{rege}} - e_{\text{regină}} \simeq \begin{bmatrix} -2 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (23)$$

, valoare care poate fi tradusă sub forma "O regină este pentru un rege ce este o femeie pentru un bărbat." Vectorul e rezultat din ecuația 23 ne indică faptul că diferența majoră dintre cele două cuvinte din punct de vedere semantic este genul, neexistând alte diferențe notabile.

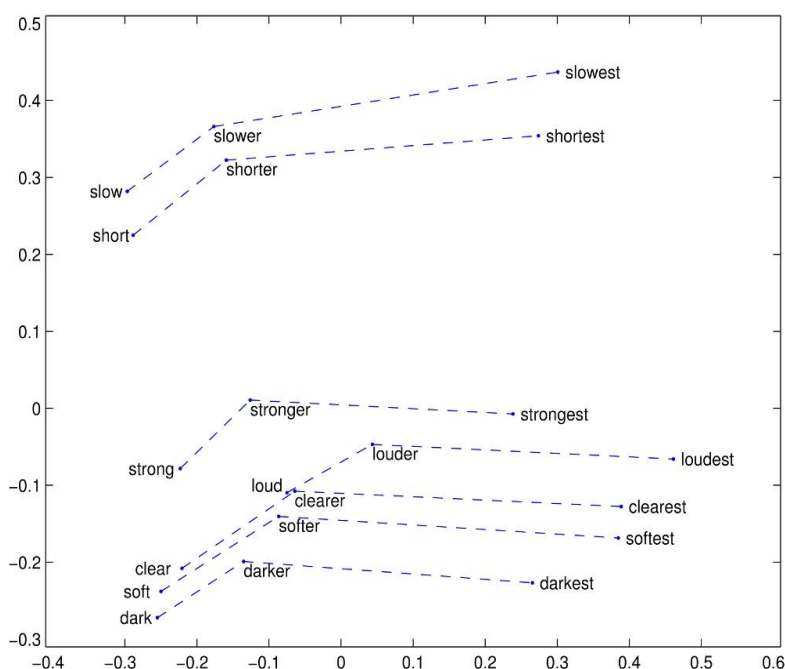


Figura 12 Reprezentare în dimensionalitate redusă a spațiului valorilor *embeddings* ale cuvintelor⁵

⁵ ©<https://nlp.stanford.edu/projects/glove/>, accesat la (23-06-2021)

În mod uzual, dimensiunea vectorilor de valori ce caracterizează cuvintele este de ordinul sutelor, lucru care face spațiul de *embeddings* greu de vizualizat. În figura 12 putem observa o reprezentare în plan 2D a acestui spațiu, reprezentare obținută cu ajutorul algoritmului t-SNE de reducere non-lineară a dimensionalității.

Având la dispoziție valorile caracteristice ale cuvintelor, putem extrage similaritatea semantică dintre oricare două cuvinte din vocabular folosind operații aritmetice, calculând distanțele sau unghiurile dintre două cuvinte oarecare.

Modelul GloVe [14] pre-antrenat va sta la baza codificatorului nostru de text. Proiectul este open-source și totodată oferă o colecție de ponderi obținută în urma unei sesiuni de antrenare pe un set vast de date.

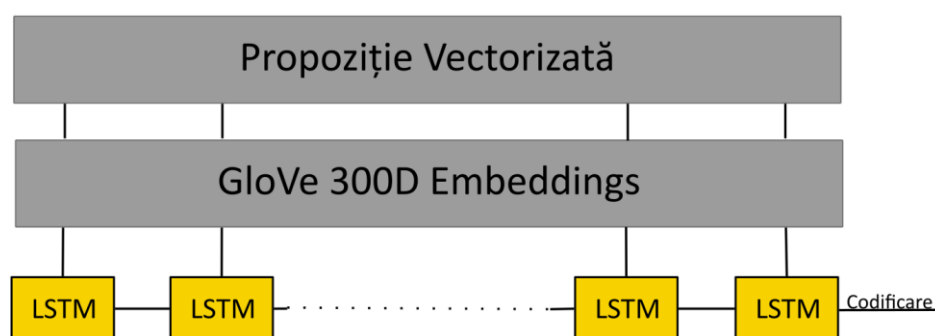


Figura 13 Codificatorul de propoziții

Peste acest model pre-antrenat, introducem un strat de unități recurente LSTM cu aceeași motivație ca la codificatorul de imagini. Acest strat va avea responsabilitatea de a învăța o funcție de codificare care va reprezenta un limbaj neutru între codificator și decodificator.

4.5 Decodificatorul și modelul complet

În vederea definitivării arhitecturii modelului folosit, definim decodificatorul ca două straturi de celule recurente LSTM [11] urmate de câte un strat ce aplică funcția non-lineară Softmax pentru a obține o distribuție de probabilități peste întreg vocabularul folosit.

Pentru a putea genera text descriptiv, aceste straturi sunt aplicate pe întreaga dimensiune a secvențelor de timp, fiecare dintre ele fiind responsabilă de generarea unui singur cuvânt din propoziție.

Pe baza rețelelor responsabile de generarea codificărilor descrise în secțiunile 4.3 și 4.4, mai departe putem vizualiza în figura 14 întregul model folosit ce stă la baza experimentelor noastre.

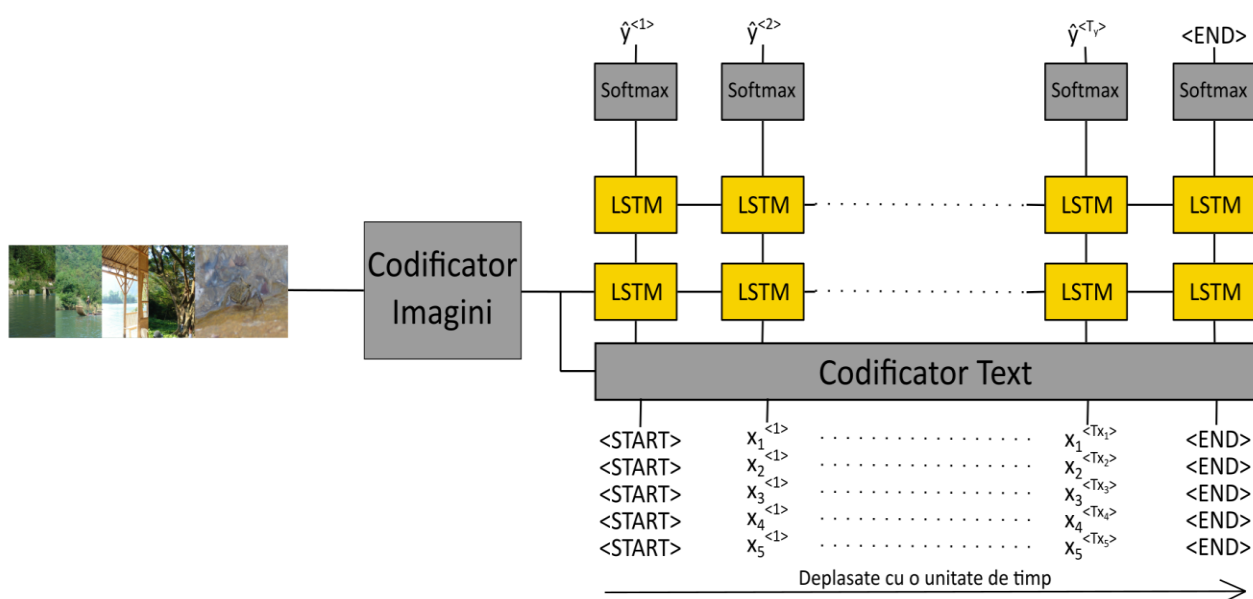


Figura 14 Modelul complet ce va fi antrenat

Așadar, pe baza componentelor descrise putem vizualiza rețeaua completă ce va fi antrenată.

În capitolul următor prezentăm setul de date folosit pentru a antrena modelul descris în figura 14, detalii de implementare și tehnicile folosite pentru a putea folosi modelul introdus. Totodată, detaliem funcționarea și modul de utilizare a arhitecturii propuse atât în modul de antrenare cât și în modul de inferență.

5 DETALII DE IMPLEMENTARE

În subcapitolele următoare vom detalia tehnologiile folosite, vom intra în amănuntele datelor folosite pentru antrenarea modelului și vom prezenta atât modul de funcționare al acestuia cât și dificultățile tehnice întâlnite pe parcursul dezvoltării.

5.1 Tehnologii

Fără îndoială putem susține că *Python* este de departe cel mai popular limbaj de programare pentru dezvoltarea aplicațiilor de învățare automată⁶. Acest lucru se datorează numeroaselor biblioteci care reușesc să eficientizeze masiv procesul de dezvoltare cât și datorită bibliotecilor de procesare numerică eficiente și ușor de utilizat. Așadar, la baza proiectului se află limbajul de programare *Python 3.9* împreună cu următoarele biblioteci:

- **numpy** [15], folosit atât explicit cât și implicit prin intermediul altor biblioteci, pentru operații aritmetice folosind structuri multidimensionale
- **pickle** [16], folosit pentru a persista datele într-un mod eficient
- **Keras** [17], folosit pentru definirea și construirea rețelelor neuronale
- **Tensorflow 2.5.0** [18], bibliotecă care stă la baza backend-ului Keras, folosită atât implicit cât și explicit pentru gestionarea datelor
- **matplotlib** [19], folosit pentru crearea de grafice și vizualizarea datelor
- **PIL** [22], folosit pentru a vizualiza, filtra, manipula și adăuga text peste imagini

De asemenea, având de-a face cu un set mare de date și îmbinând rețele cu o complexitate considerabilă, pentru a antrena modelul ne folosim de suportul bibliotecilor de procesare pe unitatea grafică, profitând astfel de aptitudinile acestora de paralelizare masivă. Așadar, folosim:

- **CUDA Toolkit 11.2** [20]
- **cuDNN SDK 8.10** [21]

Aceste kit-uri de dezvoltare software ne oferă posibilitatea de a antrena mult mai rapid modelele noastre folosindu-ne de un sistem ce are la bază o unitate grafică Nvidia GeForce RTX 2080 SUPER împreună cu un procesor Ryzen 9 3900X. Avem astfel posibilitatea de a utiliza un set de date de o mărime considerabilă.

⁶ <https://towardsdatascience.com/what-is-the-best-programming-language-for-machine-learning-a745c156d6b7>

5.2 Seturile de date

În acest capitol urmărim familiarizarea cu seturile de date pe care lucrăm și cu seturile de date ale modelelor pre-antrenate folosite, de ce folosim acest set de date, iar la final clarificarea metodei abordate în procesarea acestuia.

5.2.1 Prezentarea seturilor de date

Huang et al. [5] introduc un nou set de date pe nume VIST (*Visual Storytelling*) care va sta la baza tuturor lucrărilor ce abordează problema generării de povestiri din date vizuale.

Setul de date este creat cu ajutorul platformei *Amazon Mechanical Turk*, platformă ce reprezintă o piață de *crowdsourcing* folosită pentru externalizarea unor procese către o forță de muncă distribuită. Așadar, poveștile descrise în setul de date este construit pe baza judecății oamenilor.

Așadar, ia naștere setul de date VIST, un set de date ce cuprinde 10.117 albume compuse dintr-un total de 210.819 imagini colectate de pe platforma *Flickr* și 50.000 de povești. Poveștile sunt obținute prin alegerea a 5 poze dintr-un album și descrierea lor, o propoziție pe imagine.

Apoi, Huang et al. [5] supun aceste povești la un proces de normalizare, proces ce presupune:

1. înlocuirea numelor persoanelor cu tokenii *[male]* sau *[female]*, păstrând genul persoanei din imagine
2. înlocuirea denumirilor locațiilor cu tokenul *location*
3. gruparea și reprezentarea semnelor de punctuație ca fiind tokeni de sine stătători separați prin spațiu
4. transformarea tuturor cuvintelor în cuvinte scrise cu minuscule

Huang et al. [5] introduc astfel trei subseturi de date care ajută mai departe la conturarea problemei:

1. Descrierea imaginilor în izolare (Description of Images-in-Isolation, DII)
2. Descrierea imaginilor în secvență (Description of Images-in-Sequence, DIS)
3. Povești descrise în secvență (Stories of Images-in-Sequence, SII)

Tabelul 4 - Cele mai frecvente cuvinte ale fiecărui corpus de text prezentat în setul de date VIST [5]

DII			DIS			SII		
man	sitting	black	chatting	amount	trunk	went	[female]	see
woman	white	large	gentleman	goers	facing	got	today	saw
standing	two	front	enjoys	sofa	bench	[male]	decided	came
holding	young	group	folks	egg	enjoying	took	really	started
wearing	image		shoreline	female		great	time	

În tabelul 4 observăm cele mai comune cuvinte ale fiecărui set de texte în parte. Setul de date SII prezintă o frecvență ridicată a cuvintelor ce posedă un caracter mai personal față de celelalte două. În comparație cu SII, DII reflectă un context lipsit de dinamism și de specificitate socială.

Aceste caracteristici sunt esențiale în descrierea unei povești. Așadar, setul de texte SIS va sta la baza antrenării modelului nostru.

Pentru cele două codificatoare prezentate în secțiunile 4.3 respectiv 4.4, am precizat utilizarea unor modele pre-antrenate.

Pentru modelul codicatorul de imagini, modelul Xception [4] vine cu ponderi open-source deja antrenate pe un set de 1.2 milioane de imagini pentru clasificarea a 1000 de clase de obiecte în cadrul concursului anual ILSVRC (*ImageNet Large Scale Visual Recognition Challenge*).

Pentru modelul codicatorului de texte, am folosit GloVe [14] 300D care presupune vectori de *embeddings* de 300 de elemente, pre-antrenat pe tot corpusul *Wikipedia 2014* și *Gigaword 5*, analizând astfel 42 de miliarde de cuvinte și construind un vocabular de 400.000 de cuvinte.

5.2.2 Procesarea setului de date

Lucrând cu un set de date considerabil de mare, simpla procesare a acestora reprezintă o provocare. Imaginile ce constituie setul de date pregătit pentru antrenare, excluzând setul de validare și setul de testare, au o amprentă pe disc de aproximativ 280GB.

Pentru o astfel de problemă, Tensorflow [18] pune la dispoziție o bibliotecă de prelucrare a datelor în format TFRecord.

TFRecord este un format simplu compus din "exemple" într-un format de tipul `{"cheie": valoare}` pentru stocarea și serializarea eficientă a datelor structurate. Acest format este proiectat pentru lucrul cu Tensorflow [18] și oferă un avantaj remarcabil în procesarea paralelă a acestora.

SIS leagă o frază la o imagine pentru povești constituite din câte 5 imagini. Setul de date ne oferă această relație text-imagine cu ajutorul unui JSON din care extragem textul trecut prin etapa de normalizare descris la punctul 5.2.1 și creăm legătura către imaginea la care se referă, obținând astfel structura exemplurilor care vor constitui fișierele noastre de format TFRecord care vor sta la baza antrenării algoritmului. O intrare din fișierul TFRecord păstrat va conține datele binare reprezentative a celor 5 imagini ale poveștii împreună cu cele 5 texte descriptive.

În procesul de creare a acestor fișiere, fiecare imagine este trecută printr-un proces de

filtrare unde sunt eliminate toate imaginile corupte, imagini în format JPEG care nu respectă structura imaginilor compresate cu acest standard. Pentru asta verificăm prezența și corectitudinea *marker bytes* de la începutul imaginii și *end-of-image marker*.

Având setul de date în format TFRecord și imaginile corect filtrate, putem avea certitudinea că nu ne vom lovi de erori în procesul de antrenare.

5.3 Detaliile algoritmului de antrenare

Vocabularul folosit este construit pe baza tuturor textelor atât din setul de texte pentru antrenare cât și din cele pentru validare și testare. După indexarea tuturor cuvintelor separate prin spațiu și eliminarea cuvintelor ce apar de 3 sau de mai puține ori, obținem un vocabular de 12.191 de cuvinte. Vocabularul include 3 tokeni speciali:

- [UNK], token care marchează un cuvânt necunoscut, ce nu apare în vocabular
- [START], token ce semnalează începutul unei noi propoziții
- [END], token ce indică finalul unei propoziții

Pentru implementarea algoritmului am folosit reprezentarea directă a cuvintelor ca indexul apariției lui în vocabular.

Având de-a face cu un model ce procesează date secvențiale de mărimi necunoscute, avem nevoie de un mecanism ce satisface nevoia și cererea unei dimensiuni fixe ale straturilor rețelei. Pentru asta, rezervăm indexul 0 al vocabularului ca fiind o valoare de *padding*, un token special ce va fi folosit pentru a satisface caracterul dinamic al generării de text și va duce toate propozițiile la o lungime fixă. Acest lucru ne va putea permite definirea rețelei de o lungime fixă.

Calculând frecvența lungimilor propozițiilor din întreg setul de date, incluzând și seturile de validare și test, obținem că pentru acoperirea a 90% din textele prezente avem nevoie de secvențe de lungime egală cu 23 de cuvinte pe imagine, putând astfel defini decodificatorul.

Algoritmul este antrenat folosind optimizatorul *Adaptive Moment Estimation* (Adam) [22] cu configurările de bază a hiper-parametrilor. Pentru a evita supraîncadrarea, folosim un *callback* de oprire a antrenării în momentul în care eroarea pe setul de date de validare eșuează să scadă pentru 4 epoci consecutive. De asemenea, vom folosi *Learning Rate Decay*, un callback responsabil de reducerea pasului de antrenare cu un factor de 0.2 la fiecare două epoci consecutive în care eroarea peste setul de validare nu a scăzut.

Drept funcție pe care dorim să o minimizăm în procesul de antrenare, vom folosi *cross-entropy loss*

$$J(W) = -\frac{1}{N} \sum_{i=1}^M [y_i \log(\hat{y}_i) - (1 - y_i) \log(1 - \hat{y}_i)] \quad (24)$$

unde

- parametrul funcției de cost după care minimizăm, W , reprezintă parametrii modelului, ponderile învățate
- M reprezintă dimensiunea lotului
- y_i reprezintă clasa corectă
- \hat{y}_i reprezintă clasa prezisă

Având ca ieșire a algoritmului distribuția de probabilitate a fiecărui cuvânt din vocabular, fiecare token prezent în acesta reprezintă o clasă de clasificare.

Deoarece folosim reprezentarea cuvintelor prin maparea acestora la indecși în detrimentul reprezentării *one-hot*, ecuația 24 este adaptată și adusă la dimensiunile corespunzătoare.

În faza de antrenare, algoritmul primește la intrare secvența de cuvinte y_i permutate cu o poziție la dreapta. Astfel, în generarea fiecărei propoziții, algoritmul va avea ca intrare în faza incipientă tokenul de $[START]$. La fiecare moment t modelul nu are în scop tokenul corect, ci are în scop toate cuvintele corecte până la și incluzând $t - 1$. Astfel, obiectivul algoritmului va fi învățarea automată de generare a cuvântului y_t cunoscând toate cuvintele anterioare lui de la y_1 la y_{t-1} . Această euristică este cunoscută sub denumirea de Beam Search. Așadar, scopul algoritmului este învățarea:

$$\operatorname{argmax}_y \prod_{t=1}^{T_y} P(y_t / [START], y_1, y_2, \dots, y_{t-1}) \quad (25)$$

În scopul generării poveștilor, cu un număr considerabil de cuvinte pe propoziție, în procesul antrenării ecuația 25 va suferi de *numerical underflow*. Pentru rezolvarea acestei probleme, aplicăm operația logaritmică. Logaritmul este o funcție monoton crescătoare, lucru care va păstra obiectivul maximizării după y . Ecuația folosită devine:

$$\operatorname{argmax}_y \sum_{t=1}^{T_y} \log(P(y_t / [START], y_1, y_2, \dots, y_{t-1})) \quad (26)$$

Acest lucru însă nu rezolvă problema favorizării propozițiilor foarte scurte, probabilitatea propozițiilor fiind proporțional mai mică cu numărul de cuvinte de generat, cu T_y . Așadar, introducem o nouă euristică de normalizare iar ecuația 26 devine:

$$\operatorname{argmax}_y \frac{1}{T_y^\alpha} \sum_{t=1}^{T_y} \log(P(y_t / [START], y_1, y_2, \dots, y_{t-1})) \quad (27)$$

unde α reprezintă un hiperparametru care controlează gradul de normalizare aplicat.

Antrenând algoritmi în modul descris, mai departe ne propunem explorarea provocărilor survenite în folosirea acestuia în modul de inferență.

5.4 Detaliile algoritmului de inferență

În vederea generării de propoziții descriptive pe baza algoritmului descris în capitolul 5.3, avem nevoie de un mecanism de propagare atât a stărilor ascunse a celulelor LSTM cât și a memoriilor acestora. Acest lucru se datorează faptului că în modul de inferență nu mai avem la dispoziție texte descriptive "corecte" ca intrare.

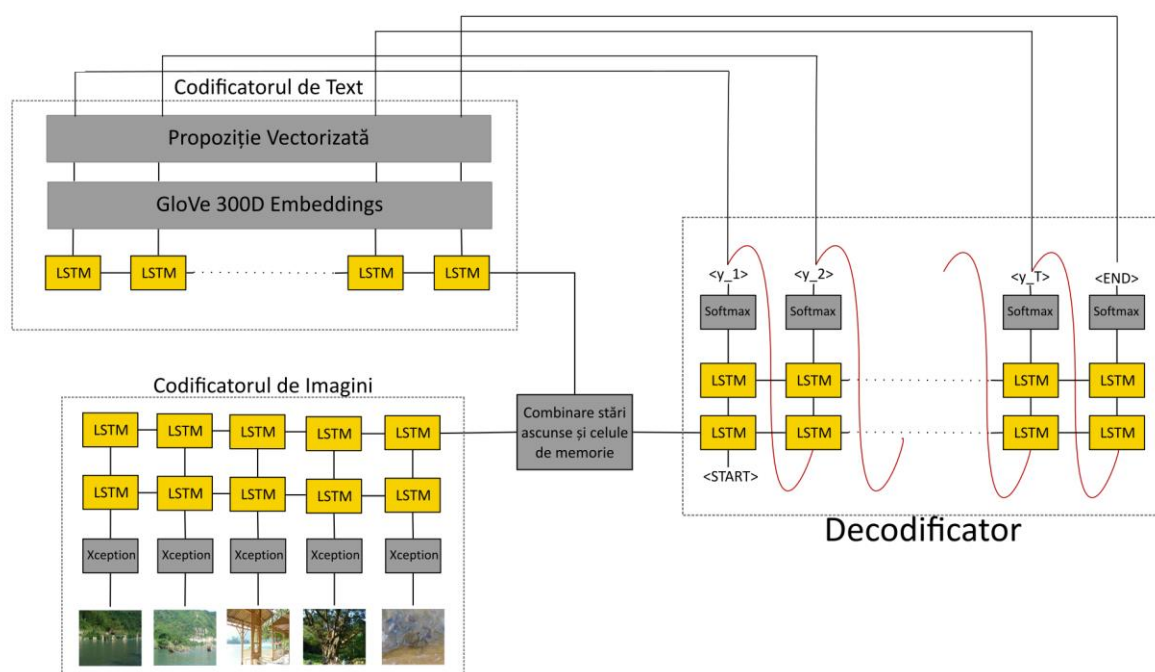


Figura 15 Schema modelului în modul de inferență.

Astfel, pentru rularea modelului antrenat în modul de inferență, la fiecare moment de timp t , decodificatorul va avea ca intrare cuvântul anterior prezis la momentul de timp $t - 1$. Pentru a păstra pe deplin funcționalitatea interioară a modelului din faza de antrenare, atât codificatorul de text cât și decodificatorul vor fi rulate pentru o singură unitate de timp și vor avea ca ieșire atât activările Softmax pentru cuvântul prezis la momentul t , cât și starea ascunsă și celula de memorie aferentă. Toate aceste 3 valori vor fi folosite în generarea cuvântului y_{t+1} .

Cum codificatorul de text are rolul de a genera codificarea propoziției anterioare și pasarea acesteia ca intrare a decodificatorului, pentru prima propoziție din secvență, decodificatorul

primește ca intrare doar codificarea secvenței de imagini. Abia după generarea primei propoziții, codificarea acesteia va fi folosită pentru generarea următoarei.

La fiecare propoziție generată, atât stările de ieșire cât și celulele de memorie ale codificatoarelor sunt gestionate într-o manieră aditivă înainte de a fi transmise către decodificator. Acest aspect asigură luarea în calcul a tuturor caracteristicilor propozițiilor generate anterior în prezicerea textului descriptiv curent.

Este discutabil dacă un întreg set de 5 imagini poate fi codificat doar cu un singur vector de caracteristici de 2048 de valori, așa cum este ilustrat și în figura 15. Atât AREL [8] cât și modelul introdus de Huang et al. [5] codifică întreaga secvență de imagini ca un singur vector de caracteristici și acesta este apoi trimis către decodificator în primul pas al rulării acestuia.

Drept urmare, am antrenat două modele separate:

- un model în care codificatorul de imagini trimite un singur vector de caracteristici reprezentativ pentru întreaga secvență de 5 imagini către decodificator în primul pas al acestuia - modelul cu codificare clasică (CC).
- un model în care codificatorul de imagini creează un vector de caracteristici de 2048 de valori pentru fiecare imagine în parte și este transmis apoi către decodificator la momentul incipient generării propoziției aferente imaginii respective. Totodată cu acest model am experimentat configurări mai agresive ale straturilor de *dropout* pentru a analiza beneficiile acestuia în antrenarea algoritmilor de procesare de limbaj natural și în încercarea de a aduce la un echilibru codificările imaginilor cu cele ale textelor - modelul cu codificări în secvență (CS).

Având astfel două modele antrenate ca punct de plecare, mai departe urmărim evaluarea performanțelor obținute ale acestora pe baza mai multor experimente atât tehnice, obiective cât și dintr-un punct de vedere subiectiv.

6 EVALUAREA REZULTATELOR

Acest capitol prezintă rezultatele obținute cât și comparația cu alte abordări prezente în literatura de specialitate. Capitolul 6.1 prezintă atât mediul folosit cât și o introducere în metricile utilizate pentru experimentele ce vor fi prezentate în subcapitolul 6.2.

6.1 Metricile folosite

Având în vedere caracterul complex și subiectiv ce stă la baza generării de povești dintr-un flux de date vizuale, cea mai bună metodă de evaluare a calității poveștilor este părerea umană.

AREL [8] demonstrează că metricile automate bazate pe potrivirea șirurilor de caractere sunt departe de a fi perfecte și nu reușesc să evalueze caracteristicile semantice ale poveștilor precum expresivitatea sau coerența. Marea majoritate a lucrărilor din literatura de specialitate ce abordează problema generării de povești critică capacitatea acestor metrici de a evalua performanța unui model și apelează la metode de *crowdsourcing* pentru a obține o evaluare corectă.

Cu toate acestea, metricile definite de o singură valoare sunt esențiale în orice proiect de învățare automată. Aceste metrici de evaluare automată oferă o referință rapidă care accelerează procesul iterativ de dezvoltare a unei asemenea aplicații. Așadar, ne vom folosi de două metrici.

6.1.1 Scorul *METEOR*

Metric for Evaluation of Translation with Explicit Ordering (METEOR) este o metrică folosită preponderent pentru probleme de tradurece automată.

Deși la baza acestei metrici se află procedura standard de potrivire exactă a cuvintelor, *METEOR* aduce și unele caracteristici mai importante precum potrivirea sinonimelor și potrivirea cuvintelor cu aceeași rădăcină sau proveniență.

Scorul final al acestei metrici este dată de media armonică dintre precizie și *recall* după cum urmează:

$$F_{mediu} = \frac{10PR}{R + 9P} \quad (28)$$

unde P reprezintă precizia și este calculată ca

$$P = \frac{m}{w_t} \quad (29)$$

iar R este referit ca *recall* și ia forma:

$$R = \frac{m}{w_r} \quad (30)$$

Termenul m se referă la numărul de cuvinte din unitatea testată (propoziția candidat) care se regăsesc și în propoziția de referință. w_t ia valoarea numărului de cuvinte din propoziția candidat, iar w_r reprezintă numărul de cuvinte din propoziția de referință.

Măsurile introduse până în acest punct au în vedere doar legături de la cuvânt la cuvânt și nimic din ceea ce privește secvențele de cuvinte. Pentru aceasta, *METEOR* se folosește de potriviri mai lungi pentru a calcula o penalizare P direct proporțională cu numărul de astfel de asocieri care nu sunt adiacente în propoziția candidat și referință.

6.1.2 Scorul *ROUGE*

Recall-Oriented Understudy for Gisting Evaluation (ROGUE) este o metrică care, la fel ca *METEOR*, ajută la evaluarea traducerilor automate.

Această metodă urmărește numărul de suprapuneri directe ale cuvintelor din ipoteză și a celor din referință. *ROUGE-L* ia în considerare strict similaritatea structurii la nivel de propoziție și oferă un scor evaluând lungimea aparițiilor comune. O caracteristică de notat este faptul că nu necesită potriviri consecutive ci potriviri în secvență care reflectă și evaluează ordinea cuvintelor în loturi de lungime n .

După cum putem deja observa, cele două metrici sunt departe de a avea capacitatea de a evalua coerența unei povești. Natura în sine a problemei studiate nu permite evaluarea rezultatelor cu o metrică descrisă printr-o simplă valoare. Caracterul subiectiv și chiar abstract al obiectivului propus de problema generării de povești cere în final o evaluare umană.

Cu toate acestea, mai departe vom analiza rezultatele obținute pe baza acestora în vederea efectuării unor experimente comparative.

6.2 Rezultate

Corectitudinea modelului obținut în această lucrare poate fi confirmată prin evaluarea punctuală și subiectivă a poveștilor generate peste setul de date de validare.

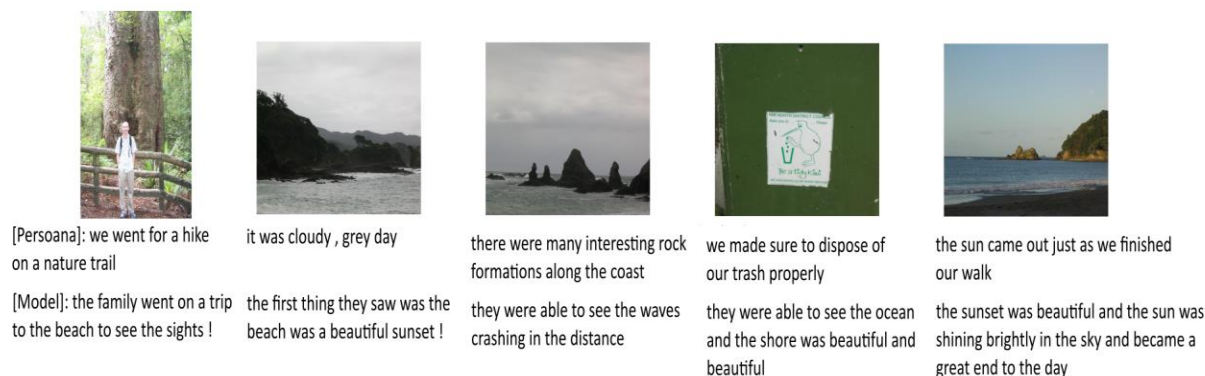


Figura 16 Exemplu de poveste generată (al doilea rând) de modelul nostru (CC) față de povestea "corectă" (primul rând) pentru o intrare din setul de date de validare

În figura 16 putem vedea rezultatul rulării algoritmului nostru, al modelului ce codifică întreaga secvență de imagini într-un singur vector de caracteristici. Una din cele mai importante caracteristici ale acestui model poate fi remarcată în predicția acestuia pentru a patra imagine. În momentul în care modelul întâmpină o imagine anormală sau neîntâlnită în raport cu setul de date peste care acesta este antrenat, algoritmul eșuează în generarea unui text punctual descriptiv pentru imaginea respectivă, dar păstrează ca scop obiectivul general al poveștii, reușind astfel să ducă la capăt o istorisire coerentă care se pretează peste imaginile de intrare.

Povestea generată de algoritmul nostru prezentat în figura 16 poate fi considerată un succes, reprezentând o întreagă poveste coerentă care se leagă direct de elemente spațiale prezentate în imagini. Cu toate acestea, obținem un scor *METEOR* de doar 0.11 și un scor *ROUGE* egal cu valoarea 0.17 pentru povestea afișată în figura 16.

Pentru o evaluare corectă în raport cu lucrările de specialitate, Huang et al. [5] precizează prin intermediul unei sesiuni de răspundere a celor mai frecvente întrebări⁷ faptul că evaluarea performanțelor modelelor propuse sunt în raport cu cea de-a doua jumătate a setului de date de validare. Pentru o raportare corectă, toate metricele prezentate mai departe se raportează la acest subset de date. Mai mult decât atât, valorile prezentate sunt obținute prin intermediul rulării aceluiași unelte de evaluare a scorurilor *METEOR* [23] și *ROUGE* [24] cu aceiași parametri folosiți și în lucrarea originală [5] ce a introdus acest set de date. Așadar, pentru evaluarea performanțelor ne-am folosit de un subset de 2238 de intrări din cele 4988 prezente în setul de validare, subset obținut după filtrarea poveștilor constituite din una sau mai multe imagini corupte.

⁷ <https://visionandlanguage.net/VIST/faq.html>, accesat la (25-06-2021)

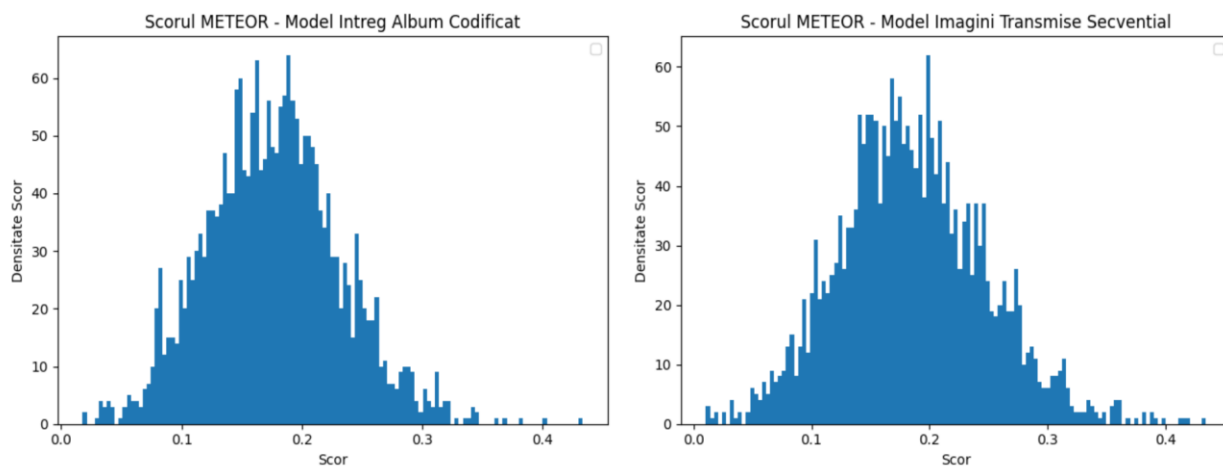


Figura 17 Histogramele valorilor metricilor METEOR ale celor două modele antrenate pentru a doua jumătate a setului de validare. Modelul CC aflându-se în stânga, iar modelul CS în dreapta

Deși similare arhitectural, metoda trimiterii codificărilor imaginilor într-o manieră secvențială prezintă o creștere de aproximativ 3% în performanța *METEOR*, așa cum putem observa și în tabelul 6.

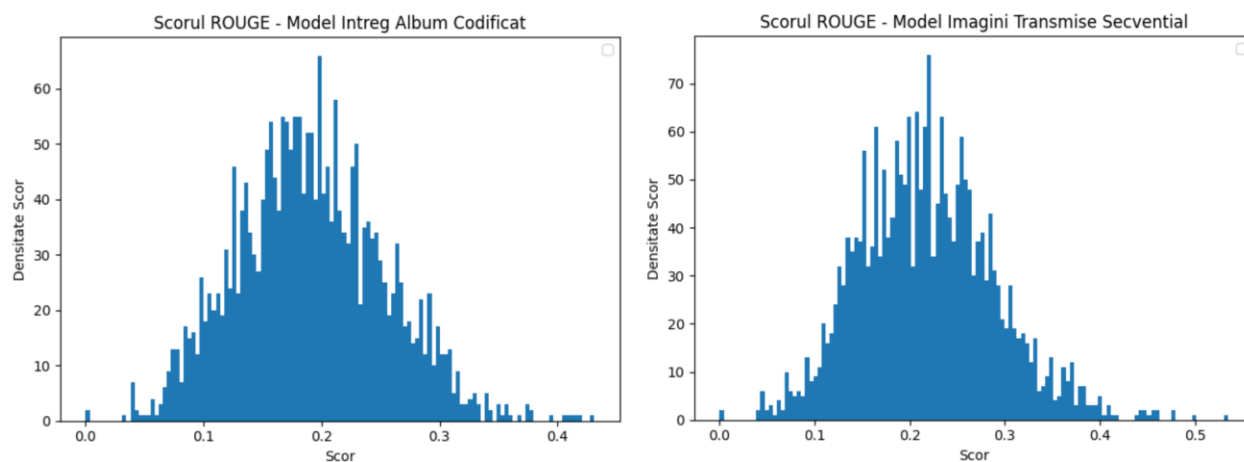


Figura 18 Histogramele valorilor metricilor ROUGE ale celor două modele antrenate pentru a doua jumătate a setului de validare. Modelul CC aflându-se în stânga, iar modelul CS în dreapta

Chiar și în cazul evaluării *ROUGE* tendința rămâne aceeași ca în cazul *METEOR*. Aici, această tendință este totuși mai accentuată, mediana distribuției Gauss permutându-se vizibil. În

cazul evaluării *ROUGE*, modelul CS obține o performanță considerabil mai bună, depășind modelul CC cu aproximativ 13%.

Reamintind caracterul de bază de evaluare a diferențelor din punct de vedere sintactic ale celor două metrice, aceste rezultate nu pot garanta superioritatea unui model față de celălalt. Unul dintre factorii majori ce împing scorul modelului CS este faptul că acest model are ca ieșire un număr mai mare de cuvinte ce referă punctual anumite obiecte aflate în imagine, în timp ce modelul CC poate fi descris ca având un caracter predictiv a acțiunilor ce urmează în pozele viitoare din album, așa cum am putut observa și în figura 16.

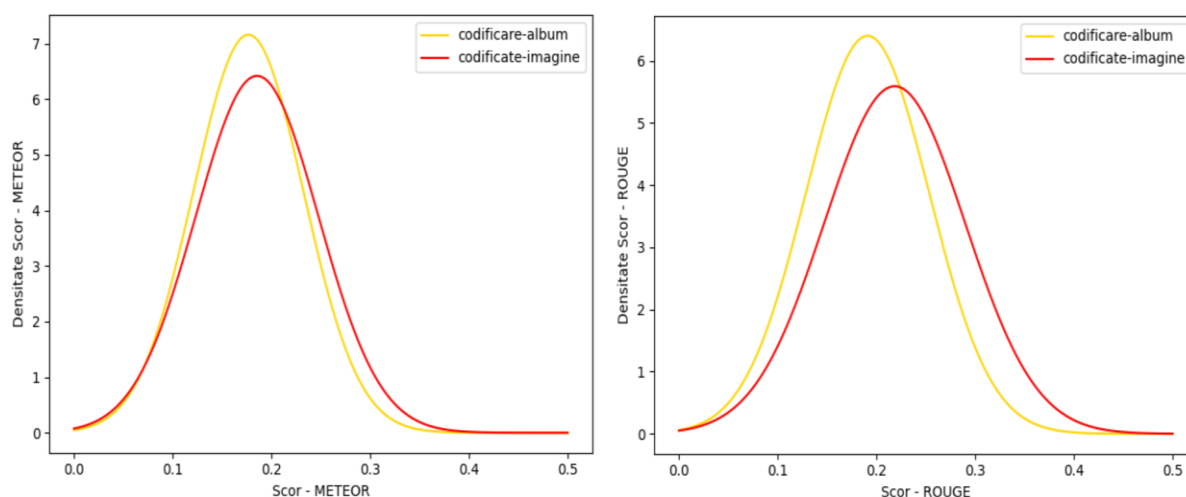


Figura 19 Distribuția Gauss a valorilor scorurilor celor două metrice pentru cele două modele antrenate. Distribuția scorurilor METEOR se află în stânga, iar ROUGE în dreapta

Așadar, modelul obține un scor mai mare în evaluarea efectuată și vizualizată în figura 19, obținând mai multe secvențe de cuvinte asemănătoare în raport cu textele descriptive ale oamenilor, dar acest lucru nu poate garanta coerența și logica poveștii, nici chiar acordurile verbale de-a lungul propozițiilor generate.



[CC]: the fireworks show started off with a bang
and a red and white firework show in the sky !

[CS]: the fireworks were very large



[CC]: the fireworks were intense against the bang

[CS]: the red ones were very loud



[CC]: fireworks were lit up and lit up with a brilliant
red explosion and red and blue colors

[CS]: and the colors were amazing



[CC]: vibrant and colorful to look at and lit up
the night sky !

[CS]: and the colors were amazing and it was a blast



[CC]: the finale was spectacular and the finale was the
best part of the night

[CS]: the finale was a blast !

Figura 20 Comparație directă între cele două modele pentru un exemplu din setul de date de validare

Figurile 20, 21 și 22 dovedesc cu atât mai mult cele susținute anterior. Modelul cu scoruri inferioare atât *METEOR* [23] cât și *ROUGE* [24] obține totuși povești mult mai bogate. CS favorizează punctual codificările imaginilor în detrimentul codificărilor textuale. Un motiv posibil pentru acest comportament este preponderența vectorilor de caracteristici definiți de codificatorul de imagini care ajung să eclipseze codificările textuale folosite de decodificator. De asemenea, se dovedește și ineficiența straturilor de *dropout* în procesul de antrenare care obligă modelul să nu se bazeze pe dependențe textuale, rezultând astfel în povești mult mai superficiale pentru modelul bazat pe trimiterea secvențială a codificărilor vizuale.

În obiectivul pur de a atinge scoruri mai bune în cele două metrice, definim ca CCSA modelul supra-antrenat al modelului cu codificări în stil clasic. Această idee este inspirată din rezultatele obținute de Huang et al. [5] care prezintă un model care are la bază în suficientă proporție o arhitectură similară și obține rezultate din ce în ce mai bogate ce conțin specificații mult mai exacte ale obiectelor prezentate în imagine odată cu scăderea erorii pe setul de antrenare. De asemenea, modelul prezentat până în acest punct este caracterizat de doar 12 epoci de antrenare finalizate, lucru care se datorează callback-ului de oprire a procesului de antrenare de îndată ce eroarea peste setul de validare stagnează. Așadar, am reluat procesul de antrenare pentru acest model, ignorând stagnarea valorii erorii peste setul de validare.

Antrenându-se pentru un număr de 10 epoci adiționale, valoarea erorii a modelului a scăzut de la 1.78 la 1.32.

Tabelul 5 Exemplu al predicțiilor celor două modele pentru același album de poze extras din setul de validare

CC	CCSA
the family got together for a reunion , and the family was excited to celebrate the new child 's birthday ,	the family got together for a reunion
the family reunion was a big hit , and the family was excited to see each other forever in the living room ,	they had a lot of fun playing
they all had a wonderful time at the party , and enjoyed each others company and talked about the new addition of the	the kids loved the outdoors
family was there , and they were so happy to be there and then they shared a few drinks and had a good	the kids were playing games
time at the reception , and the adults were happy to be there and they had a great time and enjoyed each others	sometimes the adults were there

Obținem rezultatul așteptat, noul model suferind de *supraîncadrare*. Acest lucru duce la generarea unor povești mult mai punctuale și superficiale. Analizând mai în detaliu rezultatele obținute, acesta nu se comportă la fel de bine ca modelul CC nici în categoriile de coerență sau acorduri semantice.

Deși performanța din punct de vedere subiectiv este în urmă, noul model obține rezultate mai bune în ambele metrice. Acesta obține un scor de 0.179 *METEOR*, față de 0.176 obținut pe cel antrenat pentru doar 12 epoci și un scor *ROUGE* de 0.192 față de 0.190, obținând astfel un scor mai mare cu $\approx 1.5\%$ doar supraîncadrând modelul.

Huang et al. [5] demonstrează faptul că introducând o simplă euristică precum evitarea producerii repetate, în aceeași poveste, a cuvintelor cu același conținut poate aduce un avantaj foarte mare din punct de vedere al celor două scoruri (modelul *Dups* [5]). Așadar, evitarea explicită a secvențelor precum

"They had a blast. And they had a blast."

, secvențe des întâlnite și în modelul nostru, aduce o îmbunătățire de 0.023 de puncte *METEOR*.

O altă euristică experimentată este gruparea tuturor albumelor în categorii definite de un set de cuvinte învățate de un al doilea model antrenat peste setul de date DII, obținând un model de etichetare a pozelor și de licențiere a cuvintelor potrivite. Astfel, în modul de inferență al modelului principal, prezicerile acestuia sunt condiționate de acest mecanism de licențiere (modelul *Grounded* [5]).

Deși obținând scoruri mult mai impresionante folosind astfel de euristici, considerăm că nu folosirea a astfel de metode reprezintă calea corectă pentru problema generării de povești.

Tabelul 6 Comparatie a valorilor metricilor obținute în raport cu AREL [8] și modelul original [5]

Model/Euristică	Modelele noastre			Ting-Hao (Kenneth) Huang, Francis Ferraro, et al. [1]			AREL [3]	
	CC	CCSA	CS	Greedy	Dups	Grounded	XE-ss	GAN
METEOR	0.176	0.179	0.184	0.277	0.301	0.314	0.348	0.350
ROUGE	0.190	0.192	0.219	-	-	-	0.297	0.295

Deși putem spune cu încredere că obținem o multitudine de povești notabile generate dintr-o vastă colecție de albume fotografice, în tabelul 6 putem observa ca obținem valori ale metricilor mai slabe, în special în categoria *METEOR* [23]. Luând în considerare faptul că nu putem defini o metrică caracterizată de o singură valoare capabilă să evalueze corect caracterul complex și subiectiv al poveștilor dar și critica autorilor acelorași lucrări prezentate în tabelul 6, aceste rezultate trebuie privite cu scepticism.

De remarcat este însă faptul că nu ne poziționăm departe în categoria metricii *ROUGE* față de modelul AREL [8]. Din punctul de vedere al complexității arhitecturale, modelul nostru se plasează în apropierea modelelor *Greedy* [5], deoarece nu folosim euristici definite anterior, și în apropierea modelului *XE-ss* al AREL [8], model care nu prezintă o implementare ce se folosește de arhitecturi generative adversarial.

Încă o remarcă ce susține slaba capacitate a acestor metrici este reprezentată de faptul că modelul *XE-ss*, deși este modelul ce stă la baza lucrărilor ulterioare AREL [8], obține cele mai bune rezultate în metrica *ROUGE* [24]. Acesta surclasează atât modelul generativ cât și toate modelele de învățare pe bază de recompensă ulterior dezvoltate.

Așadar, plasând modelul nostru într-un clasament corect, putem spune cu destulă certitudine că am obținut de asemenea și un scor rezonabil în metrica *ROUGE*.

7 CONCLUZII

Problema generării de text descriptiv sub formă de povești rămâne o problemă deschisă în continuare. Deși de-a lungul ultimilor ani soluțiile pentru probleme vizual-lingvistice au căpătat din ce în ce mai multă atenție în lumea învățării automate, detectarea și expunerea lucrurilor ieșite din concret necesită în continuare rafinare iar rațiunea omenească continuă să aibă un foarte mare avantaj. De la detecție de obiecte la descrierea pozelor în izolare, algoritmi de învățare automată au dat dovadă de un succes remarcabil.

Prezenta lucrare împinge aptitudinea acestei clase de algoritmi și extinde potențialul remarcabil ai algoritmilor bazați pe arhitectura codificator-decodificator, obținând un model funcțional de la un capăt la altul, o rețea neuronală profundă capabilă de generarea a întregi povești care din punct de vedere semantic pot fi considerate bogate în context și substrat, iar din punct de vedere sintactic de o lungime remarcabilă. Mai mult decât atât, discutăm și propunem o alternativă mai simplă care atinge performanțe demne de reamintit.

Din punct de vedere al metricilor de evaluare ne plasăm pe un loc inferior față de abordările populare din literatura de specialitate a domeniului generării de povești dintr-un flux de date vizuale. Deși discutabilă potrivirea acestor metrice de evaluare automată, obținem o performanță descrisă de un raport de aproximativ 67% din modelul original [5] pe una din metrice, iar în același timp obținem și o fracțiune de 74% din performanța celui mai popular și complex model [8], rapoarte care evaluează și au în vedere modelul ce poate fi considerat cel mai slab din cele dezvoltate în această lucrare din punct de vedere al calității poveștilor printr-o prismă subiectivă.

În esență, dovedim cu succes posibilitatea obținerii unui model capabil de generare de povești și evidențiem nepotrivirea metricilor descrise de o singură valoare în aprecierea calității acestora. Pe deasupra, consolidăm capacitatea de antrenare a rețelelor neuronale recurente pe bază de codificator-decodificator în încă o clasă de probleme de generare de text în limbaj natural.

8 BIBLIOGRAFIE

- [1] V. Oriol, T. Alexander, B. Samy și E. Dumitru, „Show and Tell: Lessons learned from the 2015 MSCOCO Image Captioning Challenge,” *arXiv:1609.06647v1*, 2016.
- [2] V. Ashish, S. Noam, P. Niki, U. Jakob, J. Llion, G. Aidan N. și K. Lukasz, „Attention Is All You Need,” *arXiv:1706.03762v5*, 2017.
- [3] D. Jacob, C. Ming-Wei, L. Kenton și T. Kristina, „BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding,” *arXiv:1810.04805v2*, 2019.
- [4] C. Francois, „Xception: Deep Learning with Depthwise Separable Convolutions,” *arXiv:1610.02357v3*, 2017.
- [5] H. Chao-Chun, C. Zi-Yuan, H. Chi-Yang, L. Chih-Chia, L. Tzu-Yuan, H. Ting-Hao și K. Lun-Wei, „Visual Storytelling,” *arXiv:1604.03968*, vol. 1, 2016.
- [6] H. Chao-Chun, C. Zi-Yuan, H. Chi-Yang, L. Chih-Chia, L. Tzu-Yuan, H. Ting-Hao și K. Lun-Wei, „Knowledge-Enriched Visual Storytelling,” *arXiv:1912.01496v1*, 2019.
- [7] R. Shaoqing, H. Kaiming, G. Ross și S. Jian, „Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *arXiv:1506.01497v3*, 2016.
- [8] W. Xin, C. Wenhui, W. Yuan-Fang și W. William Yang, „No Metrics Are Perfect: Adversarial Reward Learning for Visual Storytelling,” *arXiv:1804.09160v2*, 2018.
- [9] S. Ilya, V. Oriol și L. Quoc V., „Sequence to Sequence Learning with Neural Networks,” *arXiv:1409.3215v3*, 2014.
- [10] H. Sepp și S. Jurgen, „Long Short-Term Memory,” *Neural Computation*, 1997.
- [11] G. Felix A., S. Jurgen și C. Fred, „Learning to Forget: Continual Prediction with LSTM,” în *Ninth International Conference on Artificial Neural Networks ICANN 99*, 1999.
- [12] L. Yann, B. Leon, B. Yoshua și H. Patrick, „Gradient-Based Learning Applied to Document Recognition,” în *Proceedings of the IEEE*, 1998.
- [13] L. Hao, X. Zheng, T. Gavin, S. Christoph și G. Tom, „Visualizing the Loss Landscape of Neural Nets,” *arXiv:1712.09913v3*, 2018.
- [14] P. Jeffrey, S. Richard și M. Christopher D., „GloVe: Global Vectors for Word Representation,” în *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, Doha, 2014.

- [15] H. Charles R., M. K. Jarrod, v. d. W. Stefan J., G. Ralf, V. Pauli, C. David, W. Eric, T. Julian, B. N. J. S. Sebastian, K. Robert, P. Matti, H. Stephan și v. K. Marten H., „Array programming with NumPy,” *Nature*, vol. 585, 2020.
- [16] G. Van Rossum, The Python Library Reference, release 3.9.5, Python Software Foundation, 2021.
- [17] C. Francois, „Keras,” 2015. [Interactiv]. Available: <https://keras.io>. [Accesat 20 06 2021].
- [18] A. Martin, A. Ashish, B. Paul, B. Eugene, C. Zhifeng, C. Craig, C. Greg S., D. Andy, D. Jeffrey, D. Matthieu, G. Sanjay, G. Ian, H. Andrew, I. Geoffrey, I. Michael și J. Rafal, „TensorFlow: Large-scale machine learning on heterogeneous systems,” *arXiv:1603.04467v2*, 2015.
- [19] H. J. D., „Matplotlib: A 2D Graphics Environment,” *Computing in Science Engineering*, vol. 9, nr. 3, pp. 90-95, 2007.
- [20] NVIDIA, P. Vingelmann și F. FHP, „CUDA, release: 11.2,” NVIDIA, 2021. [Interactiv]. Available: <https://developer.nvidia.com/cuda-toolkit>. [Accesat 20 06 2021].
- [21] C. Sharan, W. Cliff, V. Philippe, C. Jonathan, T. John, C. Bryan și S. Evan, „cuDNN: Efficient Primitives for Deep Learning,” *arXiv:1410.0759v3*, 2014.
- [22] K. Diederik P. și B. Jimmy Lei, „Adam: A Method for Stochastic Optimization,” *arXiv:1412.6980v9*, 2017.
- [23] B. Satanjeev și L. Alon, „METEOR: An Automatic Metric for MT Evaluation with Improved Correlation with Human Judgments,” în *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, Michigan, 2005.
- [24] L. Chin-Yew, „ROUGE: A Package for Automatic Evaluation of Summaries,” în *Text Summarization Branches Out*, Barcelona, Association for Computational Linguistics, 2005, pp. 74-81.
- [25] P. Umesh, „Image Processing in Python,” *CSI Communications*, vol. 23, 2012.

9 ANEXE



[CC] the old building was very interesting to see in the city !

[CS] we visited the ancient castle



[CC] the grounds were very tall and majestic !

[CS] we saw a castle



[CC] the streets were full of brick and ornate and vibrant colors and structures !

[CS] we saw the cathedrals



[CC] the grounds were full of marble and ornate buildings ,

[CS] we had a lot of statues on the walls



[CC] and the stained glass windows were truly striking and captivating !

[CC] we saw a statue of a statue of a statue

Figura 21 Comparație directă între cele două modele pentru un exemplu din setul de date de validare



[CC] the bride and groom were very happy to be married !

[CS] the band was giving a speech



[CC] the couple exchanged vows and kissed each other the ring was beautiful !

[CS] and the people were there



[CC] the bride and groom shared a moment of happiness and happiness !

[CS] the speakers were given a prayer



[CC] the bride and groom shared their vows and the preacher smiled as she watched as the preacher read to the groom

[CS] the audience was happy



[CC] who smiled as they read their vows to each other and had a good time at the reception

[CC] and the speaker was very proud of his speech

Figura 22 Comparație directă între cele două modele pentru un exemplu din setul de date de validare