Image Captioning - Projekat iz Računarske inteligencije

Stefan Kerkoč
Bogdan Stojadinović

Septembar 2024

1 Uvod

Image captioning je proces u *computer vision*-u i obradi prirodnog jezika koji predstavlja generisanje tekstualnog opisa slike. Cilj je analizirati sadržaj slike i proizvesti rečenicu ili skup rečenica koje opisuju scenu, objekte, akcije i kontekst prikazan na slici.

Ovaj proces obično kombinuje:

- **Vizuelna obrada**: Razumevanje šta je prisutno na slici, bilo da su to objekti, akcije ili događaji (korišćenjem tehnika kao što su detekcija objekata i ekstrakcija karakteristika).
- **Generisanje jezika**: Formulisanje smislenih rečenica koje opisuju sliku (korišćenjem modela za obradu prirodnog jezika).

Ova tehnologija se koristi u različitim aplikacijama, poput pomaganja osobama sa oštećenim vidom, organizovanja i pretrage velikih baza slika, i poboljšanja pristupačnosti na platformama društvenih mreža.

2 Opis rešenja

Za rešavanje ovog problema koristili smo dva tipa modela: *Long Short Term Memory* i *Transformer*

2.1 *Long Short Term Memory* modeli

Long Short-Term Memory (LSTM) je tip arhitekture **rekurentnih neuronskih mreža (RNN)** dizajnirane za obradu i predviđanje sekvenci podataka, održavajući memoriju prethodnih unosa. LSTM rešava ograničenja tradicionalnih RNN modela koji se suočavaju sa problemom dugoročnih zavisnosti zbog **problema nestajućeg gradijenta**—gde gradijenti opadaju tokom vremena, što otežava učenje iz podataka koji zahtevaju dugoročni kontekst.

Glavni koncepti *LSTM*-a:

LSTM modeli se sastoje od posebne jedinice zvane **memorijska ćelija** i sadrže tri *gate*-a koja kontrolišu protok informacija:

1. ***Forget Gate***: Odlučuje koje informacije iz prethodnog stanja ćelije treba zaboraviti. Izlaz ovog sloja je između 0 i 1, gde 0 znači "potpuno zaboravi", a 1 "potpuno zadrži".

   $$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

   Gde su:

   - f_t izlaz vrata zaboravljanja u trenutku t

- h_{t-1} skriveno stanje iz prethodnog koraka
- x_t trenutni ulaz
- W_f i b_f težine i pristrasnosti vrata zaboravljanja
- σ sigmoidna aktivacija

2. ***Input Gate***: Određuje koje nove informacije treba sačuvati u stanju ćelije. Sastoji se iz dva dela: sigmoidnog sloja koji odlučuje koje vrednosti treba ažurirati i tanh sloja koji generiše potencijalne vrednosti za dodavanje.

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

3. ***Output Gate***: Kontroliše koliko stanja ćelije treba preneti u skriveno stanje. Omogućava da mreža šalje samo relevantne informacije u svakom koraku.

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t \cdot \tanh(C_t)$$

Cell State

Cell State, C_t, funkcioniše kao pokretna traka koja prolazi kroz čitavu sekvencu, regulisana vratima zaboravljanja i ulaza. Ovo omogućava *LSTM*-u da zadrži važne dugoročne informacije dok odbacuje nevažne podatke.

Kako *LSTM* funkcioniše:

1. **Zaboravi nevažne informacije**: Vrata zaboravljanja odlučuju koje delove stanja ćelije treba ukloniti.

2. **Dodaj nove relevantne informacije**: Vrata ulaza određuju koje nove podatke treba sačuvati.

3. **Ažuriraj izlaz**: Vrata izlaza računaju koje sledeće skriveno stanje treba biti, što će uticati na predikcije ili sledeći element sekvence.

Prednosti *LSTM*-a:

- **Dugoročna memorija**: *LSTM*-ovi efikasno hvataju dugoročne zavisnosti u sekvencama, što ih čini idealnim za zadatke poput vremenskih serija, modeliranja jezika i prepoznavanja govora.

- **Izbegavanje nestajućeg gradijenta**: Interna vrata i struktura memorijske ćelije omogućavaju *LSTM*-u da zadrži informacije preko dugih sekvenci bez problema sa nestajućim gradijentom.

Primene *LSTM*-a:

- **Obrada prirodnog jezika (*NLP*)**: Za zadatke kao što su generisanje teksta, mašinsko prevođenje i analiza sentimenta.
- **Predikcija vremenskih serija**: Prognoziranje cena akcija, vremenskih obrazaca i drugih podataka zasnovanih na sekvencama.
- **Prepoznavanje govora**: Hvatanje vremenskih zavisnosti u audio podacima.
- **Obrada videa**: Razumevanje i generisanje sekvenci frejmova u videima.

2.2 *Transformer* modeli

Transformer modeli su tip arhitekture dubokog učenja dizajnirane za obradu sekvencijalnih podataka, kao što je tekst, oslanjajući se potpuno na mehanizam koji se zove *self-attention*. Široko se koriste u zadacima obrade prirodnog jezika (NLP), kao što su mašinsko prevođenje, generisanje teksta i modeliranje jezika. U pitanju je moderniji model u odnosu na *LSTM*, ali zato i komplikovaniji.

Ključni koncepti *Transformer* modela:

1. Mehanizam *self-attention*:

Self-attention je glavna inovacija *Transformer* modela. Umesto da obrađuju sekvence jednu po jednu (kao što to rade RNN), *self-attention* omogućava modelu da simultano proceni važnost svake reči u sekvenci u odnosu na sve ostale reči.

Za svaku reč u ulazu, *self-attention* računa **ponderisani zbir** svih reči u sekvenci. To omogućava modelu da se fokusira na različite delove sekvence u zavisnosti od konteksta.

- *Query (Q)*, *Key (K)* i *Value (V)*: Svaka reč u ulaznoj sekvenci se transformiše u tri vektora: query, key i value. Ovi vektori se koriste da izračunaju koliko pažnje svaka reč treba da posveti svakoj drugoj reči u sekvenci.

Self-attention se računa kao:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

gde je d_k dimenzionalnost key vektora.

2. *Multi-head* pažnja:

Umesto da računa pažnju samo jednom, *Transformeri* koriste *multi-head attention*, što znači da računa pažnju više puta paralelno. Svaka pažnja (glava) uči različite odnose između reči.

Izlazi svih pažnji se konkateniraju i transformišu kako bi se dobile bogatije, kontekstualnije reprezentacije.

3. Poziciono kodiranje:

Pošto *Transformer* modeli ne obrađuju podatke sekvencijalno kao RNN-ovi, potrebno je kodirati poziciju svake reči u sekvenci. Poziciono kodiranje se dodaje ulaznim urezima kako bi se modelu pružio osećaj redosleda.

Obično se koristi sinusoida i kosinusoida različitih frekvencija:

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$
$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{2i/d_{model}}}\right)$$

4. *Encoder-Decoder* arhitektura:

Originalna *Transformer* arhitektura se sastoji iz dva glavna dela: **enkoder** i **dekoder**.

- **Enkoder**: Enkoder je niz slojeva koji obrađuju ulaznu sekvencu. Svaki sloj se sastoji od dva podsloja: *multi-head* pažnja i feedforward neuronska mreža.
- **Dekoder**: Dekoder, slično strukturiran, generiše izlazne sekvence. Uključuje i dodatni mehanizam pažnje koji se fokusira na izlaz enkodera, omogućavajući dekoderu da se poziva na ulaznu sekvencu tokom generisanja izlaza.

I enkoder i dekoder se sastoje od više slojeva (6 u originalnom *Transformer* modelu).

5. Feedforward neuronska mreža:

Nakon mehanizma pažnje, svaka pozicija u sekvenci prolazi kroz feedforward neuronsku mrežu (FFN), koja se sastoji iz dva potpuno povezana sloja sa ReLU aktivacijom između. Ovo dodaje nelinearnost i transformiše izlaze pažnje u apstraktnije reprezentacije.

6. Normalizacija sloja i rezidualne veze:

Transformer model koristi **normalizaciju sloja** i **rezidualne veze** kako bi stabilizovao i poboljšao obuku. Rezidualne veze omogućavaju modelu da prenese informacije između slojeva, osiguravajući da niže nivoe informacije ne budu izgubljene.

Kako *Transformer* funkcioniše:

1. **Ulazna reprezentacija**: Svaka reč u ulaznoj sekvenci se konvertuje u urezivanje, a poziciono kodiranje se dodaje kako bi se omogućio osećaj reda.

2. **Enkoder**: Enkoder obrađuje ulaznu sekvencu paralelno, primenjujući *self-attention* kako bi izračunao odnose između reči i prosledi izlaz kroz feed-forward mrežu. Svaki sloj enkodera gradi dublje razumevanje ulaza.

3. **Dekoder**: Dekoder generiše izlaznu sekvencu, koristeći *multi-head* pažnju da se fokusira i na prethodno generisane izlazne tokene i na reprezentacije ulazne sekvence.

4. **Izlaz**: Na svakom koraku, dekoder proizvodi sledeći token u izlaznoj sekvenci dok se cela sekvenca ne generiše.

Ključne inovacije i prednosti *Transformera*:

- **Paralelizacija**: Za razliku od RNN-ova, koji obrađuju ulaz sekvencijalno, *Transformeri* obrađuju celu ulaznu sekvencu odjednom, što omogućava bržu obuku i efikasnije korišćenje modernih hardverskih resursa (kao što su GPU-ovi).

- **Hvatanje dugoročnih zavisnosti**: Mehanizam *self-attention* omogućava *Transformer* modelima da bolje modeluju odnose između udaljenih reči u sekvenci nego RNN-ovi, koji se bore sa dugim zavisnostima.

- **Skalabilnost**: *Transformeri* su veoma skalabilni i doveli su do modela u stanju umetnosti kao što su *BERT*, *GPT*, *T5* i *BART* za različite NLP zadatke.

- **Pre-obuka i transferno učenje**: Pre-obučeni *Transformeri*, kao što su *BERT* i *GPT*, mogu se fino prilagoditi za specifične zadatke, što ih čini veoma fleksibilnim i široko korišćenim.

Primene *Transformer* modela:

- **Mašinsko prevođenje**: Pretvaranje teksta sa jednog jezika na drugi.

- **Sažimanje teksta**: Generisanje sažetaka dužih tekstova.

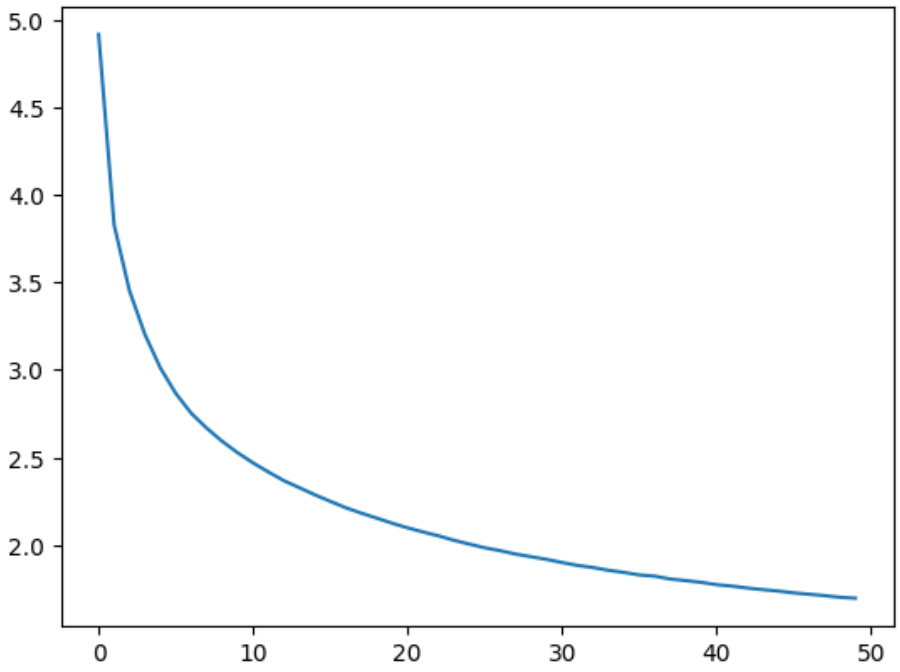
- **Generisanje teksta**: Modeli kao što je *GPT* mogu generisati tekst sličan ljudskom.

- **Odgovaranje na pitanja**: Modeli kao što je *BERT* su odlični u pružanju tačnih odgovora na pitanja na osnovu datog teksta.

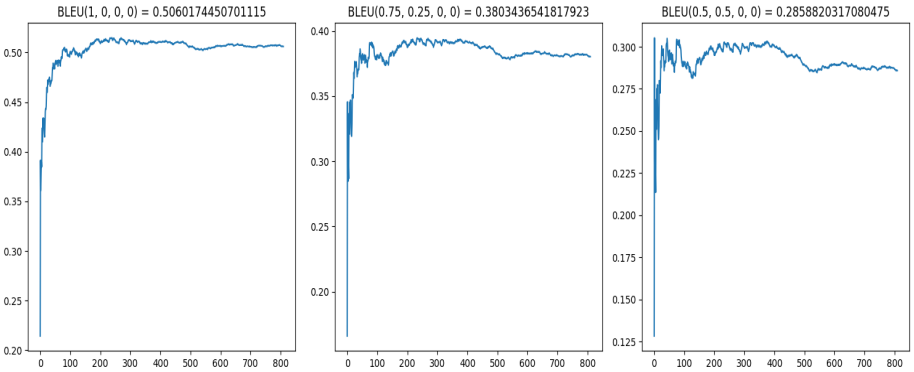
- **Obrada govora i slika**: Varijante *Transformera*, kao što su Vision *Transformers* (ViTs), primenjuju se na zadatke poput klasifikacije slika i prepoznavanja govora.

3 Rezultati

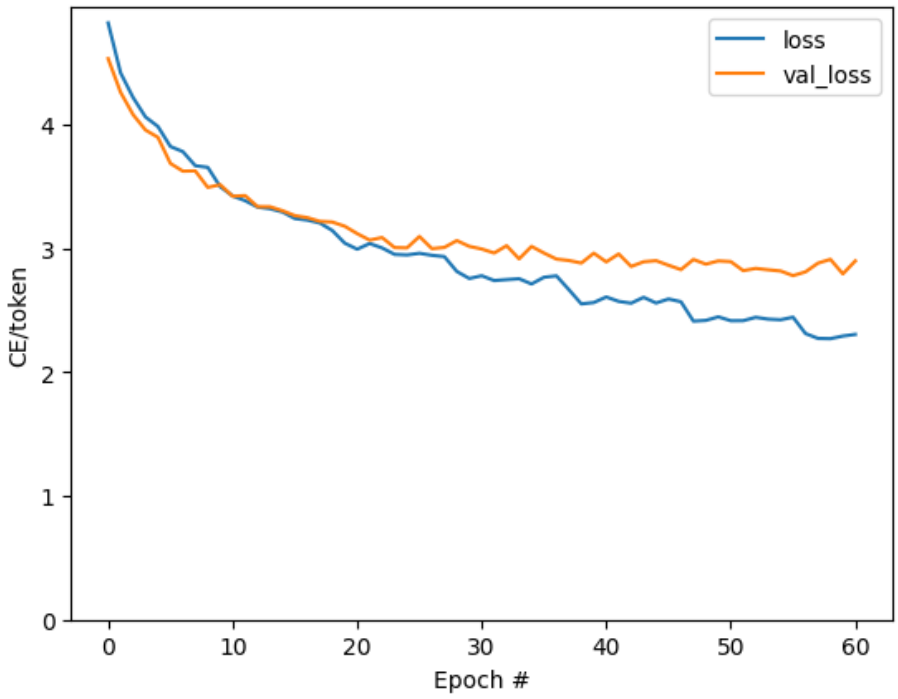
Napomena: U opisu slika prvo su navedeni izlazi *LSTM*, a zatim *Transformer* modela (nakon zapete).



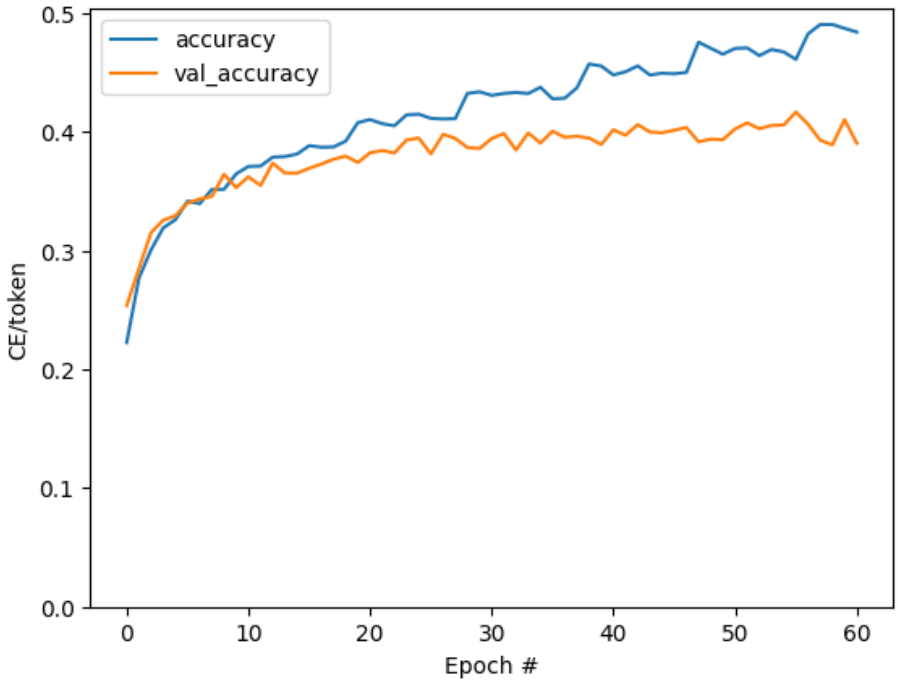
Slika 1: Grafik *loss* funckije kroz epohe kod *LSTM* modela



Slika 2: Grafik funkcije BLEU metrike kroz vreme kod *LSTM* modela



Slika 3: Grafik *loss* funckije kroz epohe kod *Transformer* modela



Slika 4: Grafik funkcije tačnosti kroz vreme kod *Transformer* modela

Slika 5:
black and white dog jumps over fence,
a black dog jumps over a white hurdle on the grass

Slika 6:
boy in wetsuit jumps into pool,
a little girl playing with a boy jumping into a swimming pool

Slika 7:
man wearing skis is skiing,
the skier is at the track

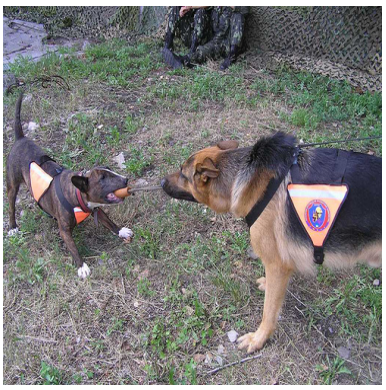
Slika 8:
man is snowboarding down snowy hill,
a snowboarder leaps up on his way up a mountain track

Slika 9:
three dogs swimming in water in the water,
three dogs are running into water at the pond

Slika 10:
two people silhouetted against lake in the sunset,
two people are standing next to each other at the lake

Slika 11:
two dogs are playing in the snow,
two dogs try to compete in the
snow

Slika 12:
bicyclist gets in the air above the
front of brick landscape,
a jumping bike racer in midair



Slika 13:
two dogs are fighting over each other on the grass,
brown and black dog play each other in grassy area

Slika 14:
the child is laying on the slide,
a man and a woman sitting by a car

Slika 15:
duck is flying in the air with hands in its mouth,
a bird takes a bride from a man watches

Slika 16:
two dogs leap into the snow,
a lake jumps over a hill by trees dog

4 Zaključak

Posmatrajući izlaze nismo sigurni koje rešenje je bolje, međutim loss funkcija nam sugeriše da je *LSTM* model bolji. Iako neočekivano s obzirom da je *Transformer* model moderniji, ipak je nekada jednostavnije rešenje bolje. Svakako na to je moglo uticati dosta faktora, kao što su skup podataka i dužina treniranja.

Literatura

[1] https://www.kaggle.com/datasets/adityajn105/flickr8k/data.

[2] https://data-flair.training/blogs/python-based-project-image-caption-generator-cnn/.

[3] https://www.analyticsvidhya.com/blog/2021/12/step-by-step-guide-to-build-image-caption-generator-using-deep-learning/.

[4] https://www.hackersrealm.net/post/image-caption-generator-using-python.

[5] https://medium.com/@raman.shinde15/image-captioning-with-flickr8k-dataset-bleu-4bcba0b52926.

[6] https://github.com/Raman-Raje/ImageCaptioning.

[7] https://www.tensorflow.org/text/tutorials/transformer.

[8] Bogdan Stojadinović Stefan Kerkoč. image-captioning. https://github.com/bogdans55/image-captioning, 2024.