Odgovori na pitanja koriscenjem modela zasnovanih na transformer arhitekturi

Simona Jevtovic 139/2019

September 27, 2024

Sadrzaj

1 Uvod

Odgovaranje na pitanja predstavlja cest NLP(Natural Language Processing) problem, jedna od njegovih varijanti bice obradjena u ovom radu. Zadatak obuhvata identifikovanje relevantnih informacija iz konteksta koji se prosledjuje modelu i ekstrakciju odgovora iz dela teksta koji je najbolji za prosledjeno pitanje koriscenjem modela dubokog ucenja zasnovanog na transformer arhitekturi. Koriste se modeli koji su prvo trenirani da imaju bazicno razumevanje teksta, nakon toga potrebno ih je obuciti za odredjeni posao(fine tuning) za sta je potrebna manja kolicina podataka u odnosu na pocetno treniranje modela(pretrainig) koji zahteva dosta podataka i resursa.

Za resavanje ovog proble najcesce su korisceni BERT modeli. BERT ima mogucnost da cita ulazni tekst sa obe strane(s leva na desno i obrnuto) sto je znacajno unapredjenje u odnosu na klasicne jezicke modele. Razumevanje konteksta u kojoj se rec nalazi i razumevanje odnosa izmedju reci u recenici su njegove bitne karakteristike. Glavna ideja u konkretnom zadatku odgovaranja na pitanja je da izvuce ogovor na pitanje iz konteksta koji mu je prosledjen tj. da pronadje pocetni i krajnji token u kontekstu koji odgovara trazenom odgovoru.

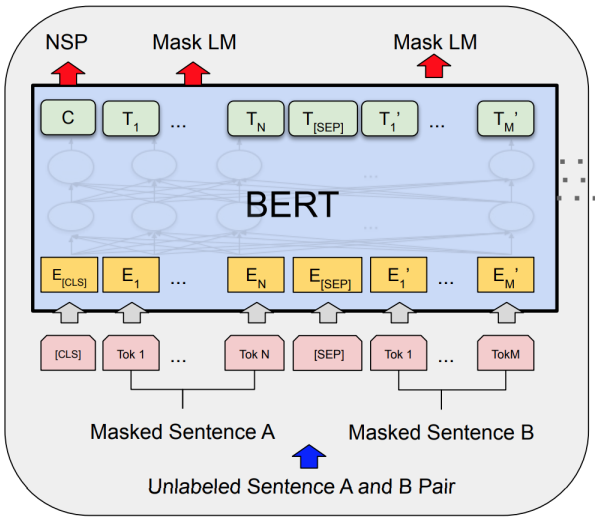

Figure 1: BERT next sequence prediction

Prva faza treniranja modela(pre-training) se sastoji iz dva dela:

- **Masked ML**: prilikom prosledjivanja recenice BERT-u, neke reci bice maskirane specijalnim tokenom [MASK], zadatak modela je da predvidi sakrivenu rec posmatrajuci reci koje se nalaze oko nje
- **Next sentence prediction**: modelu se prosledjuju dve recenice A i B, njegov zadatak je da odredi da li recenica B sledi nakon recenice A, u stvari cilj je da se vidi da li su one uopste povezane

Sto se tice arhitekture, BERT se sastoji iz enkodera za razliku od nekih modela koji u svojoj arhiterturi imaju i dekodere. Razlog za to je jer je BERT vise fokusiran na zadatke u kojima je bitno razumevanje konteksta i donosenje zakljucaka na osnovu toga, a ne na generisanje novog teksta.

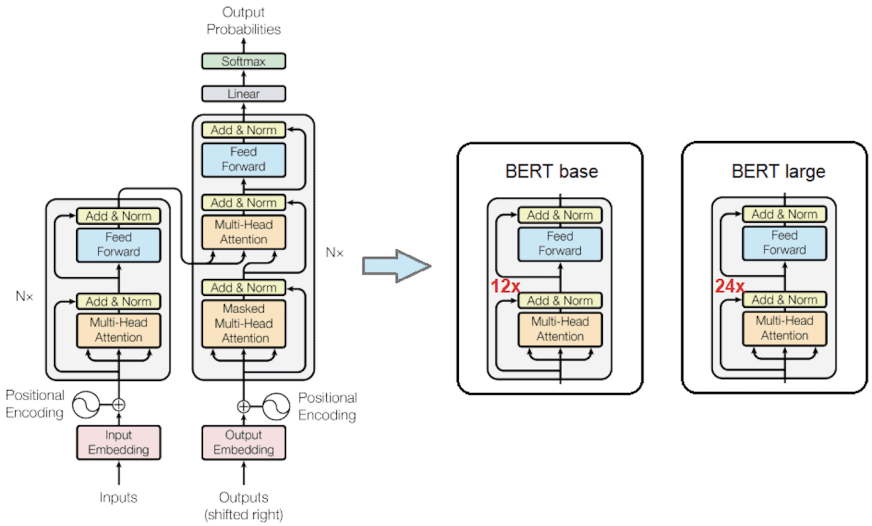

Figure 2: BERT arhitektura

2 Opis problema

2.1 Skup podataka i pretprocesiranje

Skup podataka koriscen za treniranje modela je SQUAD. To je kolekcija pitanja i odgovora, gde se za svako pitanje odgovor nalazi u kontekstu koji je takodje deo skupa podataka. Sadrzi 100.000 pitanje-odgovor parova, medjutim zbog velike kolicine vremena koja bi bila potrebna da se model trenira na tom skupu podataka, model je limitiran na manji skup od 5.500 pitanje-odgovor parova(5000 za trening i 500 za validaciju modela).

Koriscen je pretrenirani model DistilBERT koji je manja i brza verzija BERT-a. Ima 40% manje parametara i 60% je brzi u odnosu na BERT-a. Za treniranje modela od koristi je Hugging Face biblioteka, ona pruza i pristup pretreniranom modelu DistilBERT.

```
squad_train[0]
```

```
{'id': '572e86b8dfa6aa1500f8d0aa',
 'title': 'Richard_Feynman',
 'context': 'Feynman has been called the "Great Explainer". He gained a reputation for taking great care when giving explanations to his stu
dents and for making it a moral duty to make the topic accessible. His guiding principle was that, if a topic could not be explained in a fr
eshman lecture, it was not yet fully understood. Feynman gained great pleasure from coming up with such a "freshman-level" explanation, for
example, of the connection between spin and statistics. What he said was that groups of particles with spin ½ "repel", whereas groups with i
nteger spin "clump". This was a brilliantly simplified way of demonstrating how Fermi–Dirac statistics and Bose–Einstein statistics evolved
as a consequence of studying how fermions and bosons behave under a rotation of 360°. This was also a question he pondered in his more advan
ced lectures, and to which he demonstrated the solution in the 1986 Dirac memorial lecture. In the same lecture, he further explained that a
ntiparticles must exist, for if particles had only positive energies, they would not be restricted to a so-called "light cone".',
 'question': 'Feynman believed that if a topic was not easily accessible to freshmen than it was not yet what?',
 'answers': {'text': ['fully understood'], 'answer_start': [298]}}
```

Figure 3: Primer jedne instance skupa za trening

Podatke je potrebno transformisati u ulaz koji moze da razume model, to

je uradjeno koriscenjem tokenizer-a. On pretvara sirove podatke u tokene(na primer svaku rec pretvara u token), svaki token je neka numericka vrednost koju model moze da razume ako se ona nalazi u njegovom vokabularu. Postoje i specijalni tokeni koji razdvajaju deo reci koji odgovara kontekstu od dela koji sadrzi pitanje. Jos jedna bitna stvar koju tokenizer radi je standardizovanje duzine ulaznih podataka, dodavanjem takozvanog padding-a ako je ulaz kraci od odredjene duzine ili skracivanje ukoliko je duzi. Posto je postavljen parametar truncation="only_second" ako ulaz prevazilazi datu velicinu, deo konteksta ce biti odsecen a pitanje ce uvek ostati celo. Izlaz iz funkcije pretprocesiranja su tokeni koji oznacavaju pocetne i krajnje pozicije na kojima se nalazi odgovor na pitanje u kontekstu.

Funkcija koja vrsi pretprocesiranje podataka za validaciju se malo razlikuje u odnosu na funkciju za pretprocesiranje podataka za trening. Ako je kontekst veci od predvidjenog maksimalnog ulaza u model, prilikom pretprocesiranja podataka za validaciju on ce se podeliti u vise delova, necemo ga u potpunosti zanemariti jer nam je od znacaja za validaciju podataka.

Dakle, prilikom obrade podataka za trening potrebno je obratiti paznju na odredjivanje pocetne i krajnje pozicije tokena u kontekstu, dok je prilikom obrade validacionih podataka potrebno obratiti vise paznje na obradu dugackih konteksta kako ne bi doslo do gubitka podataka.

2.2 Treniranje modela

Kao sto je vec pomenuto, koriscen je pretreniran model distilBert, koji je vec treniran nad velikom kolicinom podataka kako bi stekao mogucnost razumevanja teksta. U nastavku treniranja(fine tuning) su isprobane razlicite kombinacije parametara kako bi pronasli koja kombinacija daje najbolje rezultate prilikom evaluacije modela o cemu ce biti reci kasnije. Parametri koji su menjani su:

- **learning_rate**: vazan hiperparametar koji odredjuje brzinu ucenja modela tj. kojom brzinom ce model konvergirati, isprobane vrednosti su 1e-5 i 3e-5
- **batch_size**: odredjuje broj podataka koje ce model obraditi pre nego sto dodje do promene parametara, veliki batch_size moze dovesti do preprilagodjavanja, isprobane vrednosti su 8 i 16
- **num_epochs**: odnosi se na broj kompletnih ciklusa treninga koje ce model odraditi, broj epoha koje su koriscene su 2 i 3

Prilikom treniranja modela, pracena je vrednost funkcije gubitka(loss function) nad trening podacima, koja nam nakon svake iteracije od 200 koraka daje uvid u to kako model reaguje na trening. Ovo je primer kako se vrednost funkcije gubitka menja u odnosu na iteracije:

Posmatranjem ovih vrednosti, mozemo da zakljucimo da gubitak opada periodicno kroz iteracije, sto predstavlja dobar znak da se model poboljsava vremenom. Medjutim mozemo primetiti blagi skok na prelazu sa 800. na 1000. iteraciju i sa 1000. na 1200. sto moze ukazivati na nestabilnost u treningu.

Step	Training Loss	Validation Loss
200	2.264600	No log
400	2.076800	No log
600	1.958700	No log
800	1.581100	No log
1000	1.661000	No log
1200	1.706900	No log
1400	1.566100	No log
1600	1.476100	No log
1800	1.399800	No log

Figure 4: Primer vrednosti funkcije gubitka

2.3 Evaluacija modela

Model se evaluira za svaku kombinaciju parametara, mere koje se koriste pri evaluaciji su:

- **F1 Score**: odnosi se na broj zajednickih reci u predvidjenom odgovoru i tacnom odgovoru tj. meri njihovo preklapanje
- **Exact match(EM)**: broj predvidjenih odgovora koji se u potpunosti preklapaju sa tacnim odgovorom

Potrebno je koristiti dodatnu funkciju **compute_metrics** koja pomaze u tumacenju predikcije modela. Prilikom predvidjanja model ce generisati vrednosti start_logits i end_logits koji predstavljaju numericku vrednost tj. verovatnocu da je odredjeni token pocetak odnosto kraj trazenog odgovora. Ova vrednost ce se generisati za svaki token. Ove vrednosti potrebno je dodatno obraditi u funkciji. U funkciji ce se pronaci tokeni koji imaju najvecu verovatnocu da budu pocetni i krajnji token odgovora na pitanje, u odnosu na predikcije procenjene od strane modela. Konstruise se lista koja sadrzi potencijalne odgovore(oni koji su najbolje rangirani). Iz liste odgovora izdvaja se onaj koji ima najbolju procenjenu vrednost. Takodje, potrebno je izdvojiti tacan odgovor iz originalnog teksta kako bi ga uporedili sa predvidjenim odgovorom. Izlaz iz funkcije je evaluacija predvidjenog odgovora.

Ispod su prikazane evaluacije modela na skupu podataka za evaluaciju u odnosu na kombinaciju razlicitih parametara: learning rate, batch size i num of epochs.

	learning_rate	batch_size	num_epochs	exact_match	f1
0	0.00001	8	2	36.8	48.619631
1	0.00001	8	3	46.2	57.479887
2	0.00001	16	2	49.4	59.695797
3	0.00001	16	3	50.4	61.534543
4	0.00003	8	2	50.0	61.144712
5	0.00003	8	3	47.4	59.998342
6	0.00003	16	2	48.8	62.320345
7	0.00003	16	3	49.0	61.272149

Figure 5: Evaluacija modela

Neki od zakljucaka koje mozemo doneti iz ovih rezultata:

- Povecanje batch_size parametra sa 8 na 16, dovodi do povecanja vrednosti obe metrike f1 i EM. S ozbirom na porast performansi modela sa povecanjem batch_size parametra bilo bi dobro isprobati jos vece vrednosti kao sto su 32,64.
- Iako je generalno bolje koristiti vise epoha za treniranje, u ovom slucaju ako posmatramo rezultate gde je learning_rate = 3e-5, performanse su blago smanjene ako se predje sa 2 na 3 epohe.
- Najbolji EM skor je 50.4, ta vrednost se dobija sa parametrima learning_rate = 1e-5, batch_size = 16 i num_epochs = 3.
- Najbolji F1 skor je 62.3, ta vrednost se dobija sa parametrima learning_rate = 3e-5, batch_size = 16 i num_epochs = 2.

Rezultati su prikazani i na graficima, za laksu procenu vrednosti metrika. Prvi grafik prikazuje promenu F1 skora u odnosu na parametar learning_rate, zasebno za broj batch-eva 8 i 16. F1 skor se povecava, prilikom koriscenja vece vrednosti learning_rate 3e-5.

Drugi grafik na isti nacin prikazuje promenu F1 skora u odnosu na parametar batch_size, zasebno za learning_rate 1e-5 i 3e-5. Kao sto je ranije zakljuceno F1 skor se poboljsava koriscenjem veceg broja batch-eva.

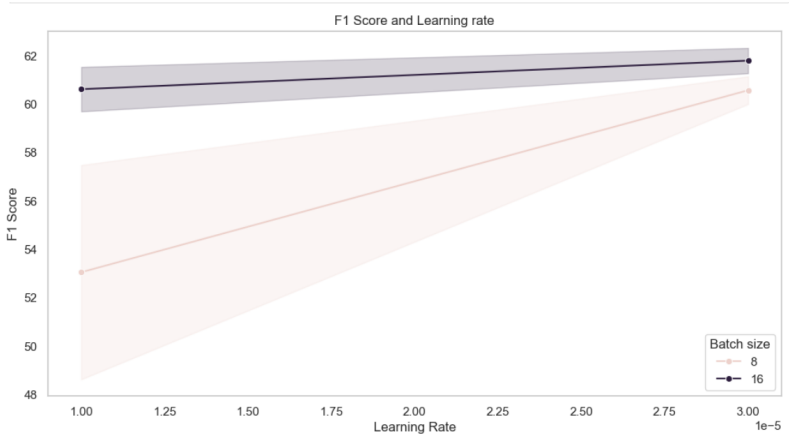

Figure 6: Grafik F1 skor u odnosu na parametar learning rate

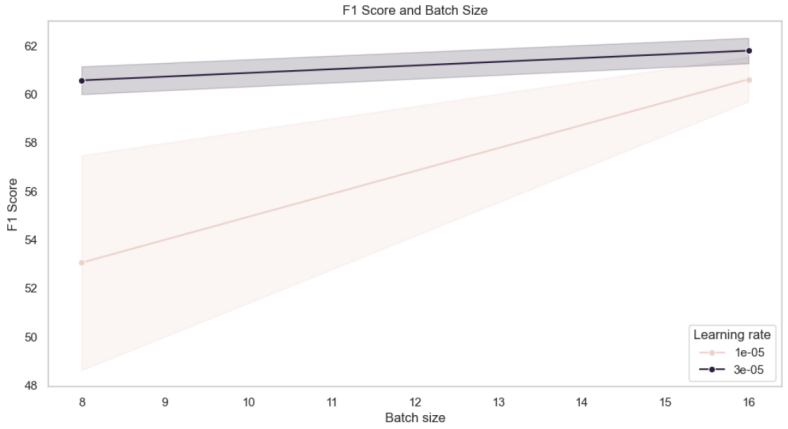

Figure 7: Grafik F1 skor u odnosu na parametar batch size

3 Zakljucak

Generalne performanse modela posmatrajuci evaluaciju rezultata su prihvatljive s obzirom da je model treniran nad malom kolicinom podataka za trening, ali ima prostora za poboljsanje. Svakako treniranje nad vise podataka bi dovelo do poboljsanja F1 i EM skora. Ono sto je zakljuceno iz datih rezultata je da bi model imao bolje performanse ukoliko bi se koristile vece vrednost parametra batch_size zato sto model ima konstantno poboljsanje performansi za veci batch_size. Ostali parametri ne daju toliko jasnu sliku u kom smeru treba ici, ali eksperimentisanje sa drugim kombinacijama parametara i vecim skupom podataka bi dovelo do boljih rezultata.

4 Reference

[1] *Keras library, hugging face library.*
`https://keras.io/examples/nlp/question_answering/`

[2] *Comparative Study of Transformer-Based Language Models.*
`https://arxiv.org/pdf/2110.03142`

[3] *SQUAD dataset.*
`https://arxiv.org/pdf/1606.05250`

[4] *DistilBERT.*
`hhttps://arxiv.org/pdf/1910.01108`

[5] *Illustrated transformer.*
`https://jalammar.github.io/illustrated-transformer/`