Podešavanje velikih jezičnih modela korištenjem tehnika poravnanja preferenci za identifikaciju investicijskih prilika

Josipović, Borna

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1597

PODEŠAVANJE VELIKIH JEZIČNIH MODELA KORIŠTENJEM TEHNIKA PORAVNANJA PREFERENCI ZA IDENTIFIKACIJU INVESTICIJSKIH PRILIKA

Borna Josipović

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

ZAVRŠNI RAD br. 1597

PODEŠAVANJE VELIKIH JEZIČNIH MODELA KORIŠTENJEM TEHNIKA PORAVNANJA PREFERENCI ZA IDENTIFIKACIJU INVESTICIJSKIH PRILIKA

Borna Josipović

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

ZAVRŠNI ZADATAK br. 1597

Pristupnik:	**Borna Josipović (0036539013)**
Studij:	Elektrotehnika i informacijska tehnologija i Računarstvo
Modul:	Računarstvo
Mentor:	doc. dr. sc. Mario Brčić

Zadatak: **Podešavanje velikih jezičnih modela korištenjem tehnika poravnanja preferenci za identifikaciju investicijskih prilika**

Opis zadatka:

U okviru završnog rada treba istražiti mogućnosti primjene tehnika izravne optimizacije preferenci (engl. Direct Preference Optimization, DPO) i podržanog učenja iz povratnih informacija (engl. Reinforcement Learning from Human Feedback, RLHF) za fino podešavanje velikih jezičnih modela (engl. Large Language Model, LLM) s ciljem poboljšanja procesa prepoznavanja investicijskih prilika koje proizlaze iz inovativnih događaja interesantnih za tržište dionica. Izvor podataka će biti e-bilteni (engl. newsletter). Odabrati postojeći utrenirani LLM otvorenog koda kao početnu točku razvoja. Čak i uz pretpostavku da model ima dobro generalno znanje i ponašanje, performanse u ovoj specifičnoj domeni mogu biti neadekvatne. Tuniranje modela za ovu specifičnu svrhu zahtijeva da model nauči preference vezane uz svrhu, u ovom slučaju uspješnosti u ulaganju. Ažuriranje modela treba provesti korištenjem DPO i RLHF tehnika finog podešavanja na temelju pametno stvorenih skupova podataka. Treba provesti eksperiment koji će usporediti neažurirani LLM, te DPO i RLHF podešene modele kroz njihove performanse u prepoznavanju inovativnih događaja za investicijske prilike.

Rok za predaju rada: 14. lipnja 2024.

Hvala doc. dr. sc. Mariu Brčiću na ugodnom mentorstvu, te obitelji na podršci.

Sadržaj

1. Uvod

Fino podešavanje velikih jezičnih modela (engl. Fine-tuning Large Language Models, LLms) revolucionariziralo je zadatke obrade prirodnog jezika (engl. Natural Language Processing, NLP). Obrada prirodnog jezika potpodručje je umjetne inteligencije koje proučava interakcije računala i prirodnog jezika. Neki od zadataka obrade prirodnog jezika su prepoznavanje govora, razumijevanje teksta, generiranje teksta, klasifikacija teksta i slično. Nama je iznimno bitna klasifikacija teksta, time se bavimo. Želimo da jezični model dani članak ispravno klasificira oznakama "signal" i "nije signal". S obzirom na to da jezični modeli imaju pristup vrlo širokom spektru znanja mi ih želimo navesti da odabiru odgovore i ponašanja koja se podudaraju s našim specifičnim ciljevima i željama. To ćemo postići koristeći dvije prominentne tehnike - podržano učenje iz povratnih informacija (engl. Reinforcement Learning from Human Feedback, RLHF) i izravnu optimizaciju preferenci (engl. Direct Preference Optimisation, DPO). Podržano učenje iz povratnih informacija standardna je tehnika poravnavanja preferenci jezičnog modela koja se bazira na maksimiziranju dobivene nagrade iz odgovora vraćenog od strane jezičnog modela. Takvo učenje sastoji se prvo od nadziranog učenja modela bez poravnavanja preferenci, taj model zatim postaje funkcija nagrade u petlji poravnavanja preferenci jezičnog modela s pomoću optimizicijskih algoritama (na primjer Proximal Policy Optimisation, PPO). Na kraju model bi trebao generirati odgovore označene s visokim nagradama. U drugu ruku izravna optimizacija preferenci je puno jednostavnija. Nije potrebno zasebno učiti model kao funkciju nagrada već se preference poravnavaju direktno u algoritmu. Cilj rada je provesti obje metode te usporediti dobivene rezultate.

2. Skup podataka

2.1. Uvod

Zašto je skup podataka iznimno bitan za učenje modela i što čini skup podataka kvalitetnim? Naš model bit će kvalitetan onoliko koliko je kvalitetan skup podataka za učenje i evaluaciju. Skup podataka mora biti dovoljno velik, te mora ispunjavati uvjete za kvalitetu. Za veličinu skupa podataka ne postoji eksplicitno definirano pravilo, no postoji praktično pravilo koje govori da bi skup podataka trebao biti za red veličine veći od parametara za učenje koje sadrži, što znači da ako skup podataka ima 10,000 parametara za učenje, onda on mora sadržavati 100,000 primjeraka. Veličina može jako varirati s obzirom na to što se želi postići učenjem, skup podataka može imati broj primjeraka veličine reda 100, a može ići i u milijune. No veličina skupa podataka nije jedino što je bitno. Podatci koje koristimo moraju biti kvalitetni ako želimo postići kvalitetne rezultate. Kvalitetu podataka teško je mjeriti, no ravnamo se po tome da biramo podatke koji će davati najbolji ishod. Podatci moraju biti pouzdani, a neke od značajka pouzdanosti su te da moramo znati da su naši podatci ispravno označeni, podatci ne smiju sadržavati previše šuma u značajkama, filtriranje mora biti uspješno izvedeno s obzirom na potrebe našeg problema, dupliciranе vrijednosti ne smiju se pojavljivati, i slično. Ako baratamo s nepouzdanim podacima vrlo je vjerojatno da model neće raditi ispravne i korisne predikcije. Podatci u našem skupu također moraju biti ispravno predstavljeni modelu (feature representation). Feature representation je ključan jer određuje kako će model primati i shvaćati ulazne podatke te kako će ulazne podatke povezivati s izlazom. Ako smo značajke dobro predstavili jezični model bit će u mogućnosti naučiti bitne informacije iz podataka te će to rezultirati boljom generalizacijom te preciznijim i relevantnijim odgovorima. Skup podataka za učenje mora biti kvalitetan te mora imati reprezentativne podatke koji će se odraziti na dobro generaliziranje te dobre rezultate na setu za testi-

ranje. Ako nam je skup podataka loš, to je jedan od čimbenika koji mogu dovesti do loše generalizacije modela i slabih rezultata na skupu za testiranje, što u konačnici može smanjiti pouzdanost i korisnost modela u stvarnom svijetu.

2.2. Prikupljanje i filtriranje podataka

Zbog unikatnosti i kompleksnosti problema, bilo je neophodno stvoriti vlastiti skup podataka. Koristili smo scraper skripte koje smo sami napisali kako bismo prikupili relevantne članke i e-biltene s različitih izvora, poput BBC-a, The Guardiana, TechCruncha, TechRadara, Wccftecha i Reutersa.[3][4] Početni broj članaka bio je oko 300 tisuća. Kako bismo bili sigurni da naš skup podataka sadrži samo članke i podatke visoke kvalitete, te također iz razloga vremenskih ograničenja procesiranja podataka, odlučili smo provesti postupak filtriranja. Filtriranje je smanjilo broj članaka na malo više od 30 tisuća.

2.3. Korištenje Google Trendsa

Google Trends web stranica je koja nam dopušta pretraživanje interesa korisnika za po volji odabrane teme kroz određene periode. Kako bismo iskoristili tu informaciju, za svaki članak morali smo izdvojiti teme. Koristeći ChatGPT i pažljivo odabrani prompt, iz svakog članka izdvojili smo 10 tema različitih razina općenitosti od vrlo specifičnih do vrlo općenitih. Nakon toga smo razvili vlastitu skriptu koja prolazi kroz naš skup podataka te iz svakog primjerka skupa podataka, uzima 5 tema i provjerava interes korisnika kroz različite periode od 3, 5 i 10 godina. S obzirom na to da Google Trends nema vlastiti API koristili smo third-party API "pytrends". Proces prikupljanja trend podataka topicsa bio je iznimno vremenski iscrpan zbog ograničenja na broj requestova koji se mogu poslati preko API-ja, te nekonzistentnosti brzine obrade. Skup podataka podijelili smo na 10 jednakih dijelova te istovremeno vrtjeli dijelove na više različitih mašina. Ovaj proces trajao je preko tjedan dana. Nakon ovog procesa skup podataka bio je spreman za određivanje i obradu signala.

2.4. Određivanje i obrada signala

Jedan od izazova s kojima smo se susreli bio je određivanje kojim člancima se može pridružiti oznaka "signal", a kojima "nije signal". U skupu podataka "signal" je označen brojem 1, a "nije signal" je označen s brojem 0. Za rješavanje ovog problema odlučili smo primijeniti dvije tehnike. Prvo smo koristili ChatGPT API s pažljivo osmišljenim promptom kako bismo automatizirano odredili signale preko cijelog dataseta. Rezultate rada ove API petlje spremali smo u polje "gpt_signal". Ovom metodom postigli smo oko 10% signala. Idući korak u procesu bio je odrediti je li članak "signal" na temelju informacija dobivenih iz Google Trendsa. Za to smo ručno postavili kriterij - članak je označen kao signal ako pokazuje linearni rast interesa. Konkretno, smatrali smo da je članak signal ako je interes za njime kroz proteklih 5 godina bio 1.64 (skoro 5/3) puta veći nego kroz protekle 3 godine, te ako je interes kroz proteklih 10 godina bio 1.9 (skoro 10/5) puta veći nego kroz proteklih 5 godina, uz uvjet da samo interes kroz protekle tri godine smije biti nula. Implementacija ove skripte rezultirala je postotkom od malo više od 10% signala. Na kraju članak je signal ako je barem jedan od dva uvjeta ispunjen: analiza trendova pokazala da je članak signal ili ChatGPT je odredio da je članak signal. Naposljetku ovo je rezultiralo ukupnim postotkom signala od 20% preko cijelog skupa podataka, što, iako nije idealno, smatramo dovoljnim za provedbu eksperimenata.

2.5. Struktura skupa podataka

Dataset je u JSON formatu i sastoji se od liste elemenata sljedećih polja:

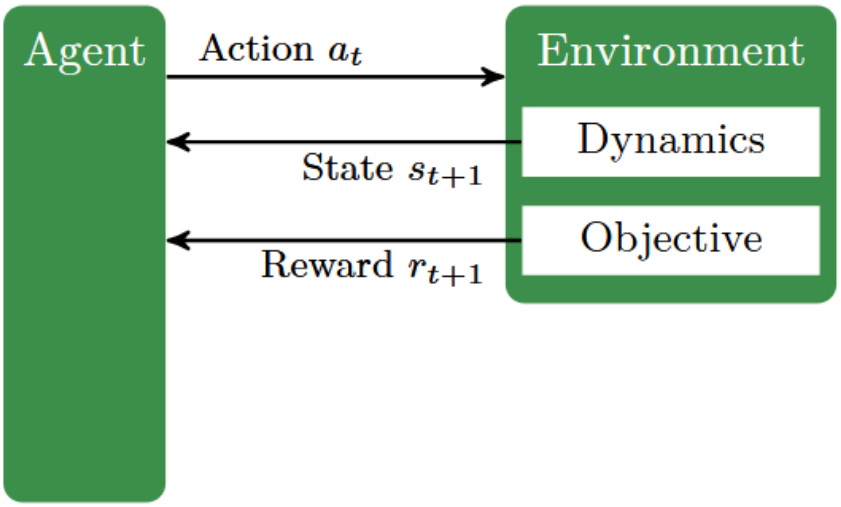
- **id**: Identifikacijski broj članka.
- **title**: Naslov članka.
- **date_published**: Datum kada je članak objavljen.
- **text**: Tekstualni sadržaj članka.
- **link**: Poveznica na izvorni članak.
- **gpt_signal**): Oznaka signala koju je dodijelio ChatGPT, 1 ako je signal, 0 ako nije signal.
- **topics**: Lista tema izvučenih iz članka.
- **growth**: Podatak o rastu interesa za pojedinu temu prema Google Trendsu.
- **signal**: Konačni podatak o značajnosti članka, 1 ako je signal, 0 ako nije signal

3. Podržano učenje iz povratnih informacija (RLHF)

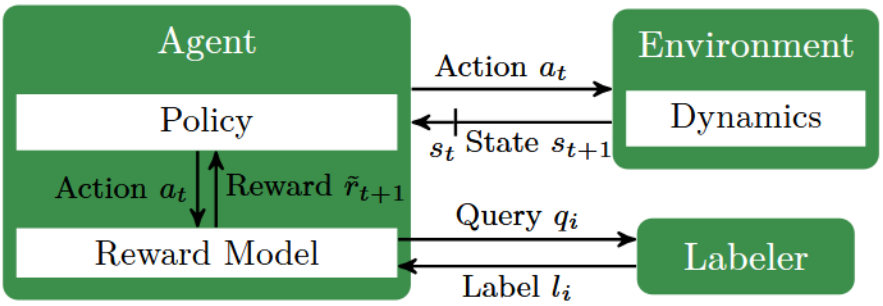
3.1. Uvod

Podržano učenje potpodručje je učenja jezičnih modela koje se bavi usmjeravanjem modela prema ponašanju i odgovorima koji maksimiziraju nagradu. Jezični model donosi odluke tijekom interakcije s okolinom te implicitno dobiva povratne informacije iz okoline, nagrade ili kazne, iz kojih uči. Podržano učenje korisno je kada se ponašanje može jasno klasificirati na binaran način ("dobro" ili "loše"). Podržano učenje povratnom informacijom od ljudi (RLHF) tehnika je podržanog učenja u kojoj agent prima nagrade ili kazne od nagradnog modela (reward model). Nagradni model uči se na skupu podataka koji sadrži upit te parove odgovora. Ljudi označavaju jedan od odgovora kao korisniji, bilo to "dobro"/"loše", brojevna skala od 1 do 5 ili slično. Ovime se u skup podataka ugrađuje strategija koju želimo da model prati, drugim riječima određuje kako bi se model trebao ponašati u određenim situacijama. Ova tehnika omogućava jezičnom modelu da uči direktno iz povratnih informacija ljudi. Korištenje ljudskog znanja, pogotovo stručnjaka iz određenih područja iznimno je korisno kada je teško jasno odrediti nagrade ili kada imamo složenu okolinu i problem kojega treba riješiti. Cilj podržanog učenja je maksimizirati dobivenu nagradu za svaku akciju, usmjeravajući model prema željenoj strategiji ponašanja. Podržano učenje može se formalizirati kao Markovljev proces odlučivanja (MDP) [5]. U Markovljevom procesu odlučivanja, imamo agenta koji promatra svoje trenutno stanje, čini neku akciju koja mijenja stanje njegove okoline te model prima nagradu za svoju akciju. Kod RLHF-a funkcija nagrade nije poznata već se uči iz povratne informacije ljudi učeći nagradni model. Ovaj proces najčešće prethodi petlji učenja politike, te može biti potpuno odvojen od istog. RLHF se temelji na postavci Markovljevih procesa odlučivanja temeljenih na preferencijama, koji se mogu definirati kao

MDP model bez funkcije nagrade, ali s dostupnim usporedbama putanja.[5].



Slika 3.1. Standardna okolina podržanog učenja
[5]



Slika 3.2. Najčešća okolina RLHF
[5]

3.2. Učenje nagrada

Najčešći pristup učenju funkcije nagrada iz parova usporedbi putanja, gdje se jedna putanja preferira nad drugom, temelji se na Bradley-Terry modelu,[6]:

$$\mathbb{P}(\tau_1 \succ \tau_2) = \frac{1}{1 + \exp(R(\tau_2) - R(\tau_1))}. \tag{3.1}$$

Putanja (trajectory) je slijed stanja i akcija koje agent prolazi tijekom okoline, označena je s τ. Znak $\succ$ označava "preferira se nad", a $R(\tau)$ označava povrat (nagradu od τ), to je zamjenska funkcija nagrade. Za tu funkciju pretpostavlja se da uzrokuje istu politiku kao i prava funkcija nagrade. Funkcija nagrade R_ψ parametrizirana s ψ u obliku:

$$R_\psi(\tau) = \sum_{h=0}^{H-1} \gamma^h R_\psi(s_h, a_h), \tag{3.2}$$

može se naučiti pomoću principa maksimalne izglednosti:

$$\max_\psi \prod_{i=1}^{N} \frac{1}{1 + \exp(R_\psi(\tau_2^i) - R_\psi(\tau_1^i))}. \tag{3.3}$$

Koristeći (3.3) možemo naučiti model da aproksimira pravu funkciju nagrade R. Za razliku od klasičnog podržanog učenja, naučena funkcija nagrade koristi se za učenje agenta umjesto za izravno uspoređivanje putanja.[5]

3.3. Petlja podržanog učenja iz povratnih informacija ljudi

U posljednjoj fazi RLHF-a moramo s pomoću prije fino podešenog modela nagrada optimirati naš model. Koristit ćemo se algoritmom proximal policy optimisation (PPO)[1]. Za konkretnu implementaciju PPO-a koristit ćemo PPO Trainer iz TRL paketa.[**?**]. Fino podešavanje modela koristeći PPO sastoji se od tri koraka:

1. Za prompt generiramo odgovor od jezičnog modela. U našem slučaju odgovor će biti klasifikacija članka kao "signal" ili "nije signal". Prompt i odgovor konkateniramo.

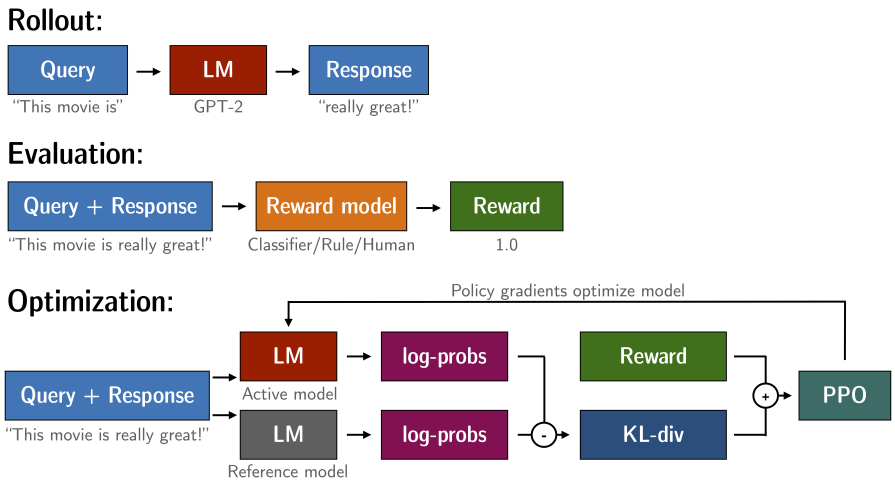
2. Konkatenirani prompt i odgovor prosljeđujemo prije naučenom modelu nagrada koji mu dodjeljuje skalarnu vrijednost.

3. Zadnji je korak optimizacije. Parovi prompta i odgovora koriste se za izračunavanje logaritamske vjerojatnosti tokena u slijedu. Ovo se provodi na učenom i referentnom modelu, koji je uobičajeno isti model samo prije finog podešavanja. Logaritamske vjerojatnosti koristimo u izračunu Kullback-Leibler (KL) divergencija.

   KL divergencija razlika je između dvije distribucije vjerojatnosti, a računa se kao:

   $$D_{KL}(\pi_\theta \parallel \pi_{\theta_{\text{old}}}) = \sum_t \pi_\theta(a_t|s_t) \log\left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)}\right)$$

   Izračunata razlika koristi se kao dodatak nagradi kako bi se osiguralo da generirani odgovori ne odstupaju previše od referentnog modela. Na kraju, naš model se uči koristeći PPO. Iz algoritma PPO uzimaju se gradijenti ciljne funkcije te se s pomoću njih ažuriraju parametri politike, usmjeravajući je prema našem cilju.



Slika 3.3. Grafički prikaz postupka opisanog iznad
[?]

3.3.1. Proximal Policy Optimisation (PPO)[1]

Pri implementaciji PPO algoritma mogu se koristiti razni pristupi, jedan od tih pristupa je "prilagodljivi koeficijent kazne Kullback-Leibler divergencije" pri kojem se kazneni koeficijent prilagođava sve dok ne postignemo željenu KL divergenciju d_{targ}.
Pri svakom ažuriranju politike odvijaju se idući koraci:

1. Prvi korak je optimizacija cilja:

$$L_{\text{KL-PEN}}(\theta) = \hat{\mathbb{E}}_t \left[\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_{\text{old}}}(a_t|s_t)} \hat{A}_t - \beta \text{KL}[\pi_{\theta_{\text{old}}}(\cdot|s_t), \pi_\theta(\cdot|s_t)] \right]$$

   Ova funkcija računa gubitak (loss) i osigurava nam da nova politika neće previše odstupati od prave. Formula se sastoji od omjera vjerojatnosti trenutne akcije u stanju trenutno politike i vjerojatnosti trenutne akcije u stanju prošle politike pomnožene s procjenom prednosti, koja je mjera koliko je trenutna akcija bolja od prosječne akcije za trenutno stanje umanjen za KL divergenciju prošle i trenutne politike pomnoženu s koeficijentom kazne.

2. Drugi korak je izračun KL divergencije s pomoću formule: $d = \hat{\mathbb{E}}_t[\mathrm{KL}[\pi_{\theta_{\text{old}}}(\cdot|s_t), \pi_\theta(\cdot|s_t)]]$ te prilagođavanje parametra β, kojeg ćemo koristiti u idućim ažuriranjima politike, po idućim kriterijima:

   - Ako je $d < \frac{d_{\text{targ}}}{1.5}$, tada $\beta \leftarrow \beta/2$
   - Ako je $d > d_{\text{targ}} \times 1.5$, tada $\beta \leftarrow \beta \times 2$

Pojednostavljeni pseudokod algoritma PPO izgleda ovako:

Algorithm 1 PPO pseudokod

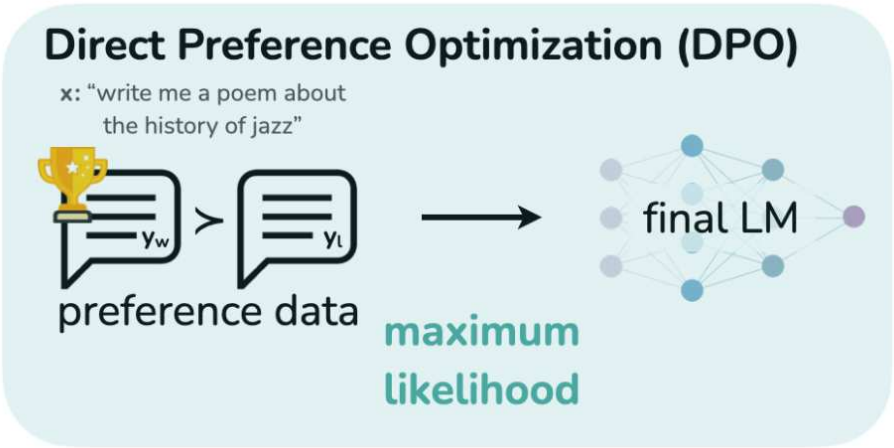
for iteration = 1, 2, … **do**
  for actor = 1, 2, …, N **do**
    Run policy $\pi_{\theta_{\text{old}}}$ in environment for T timesteps
    Compute advantage estimates $\hat{A}_1, \ldots, \hat{A}_T$
  end for
  Optimize the KL-penalized objective L
  $\theta_{\text{old}} \leftarrow \theta$
end for

Koristeći PPO omogućavamo RLHF algoritmu da stabilno optimira model, maksimizrajući dobivene nagrade uz osiguranje da politika neće previše odstupati od referentne.

4. Izravna optimizacija preferenci (DPO)

4.1. Uvod

Izravno poravnavanje preferenci koje provodimo u ovome eksperimentu bazira se direktno na radu "Direct Preference Optimization: Your Language Model is Secretly a Reward Model" [2]. Za razliku od RLHF-a, algoritam DPO koristi puno jednostavniji pristup poravnavanja preferenci. Umjesto učenja modela nagrade kojeg potom koristimo tijekom optimizacije u petlji podržanog učenja, cilj DPO-a je pronalazak optimalne politike bez petlje podržanog učenja i modela nagrade. To se postiže transformacijom formule gubitka nagrada u formulu gubitka politika. U ovome slučaju politika predstavlja i jezični model i model nagrade[2].



Slika 4.1. Pojednostavnjeni prikaz DPO
[2]

4.2. Petlja izravne optimizacije preferenci

Petlja izravne optimizacije preferenci sastoji se od par koraka:

1. Izgradnja skupa podataka

    - Za sve upite generiraju se odgovori koje ocjenjuju i označavaju ljudi. Ti upiti i odgovori moraju se strukturirati u pravilnom formatu kako bi se mogli koristiti za učenje s pomoću DPO Trainera. Skup podataka mora biti dictionary s tri polja: prompt (upiti), chosen (odabrani, to jest preferirani odgovori) i rejected (odbijeni, to jest nepoželjni odgovori). U našem slučaju automatizirali smo označavanje upita s pomoću ChatGPT API-ja. Binarnu oznaku koju je ChatGPT dao (0 ili 1) iskoristili smo kao chosen, a kao rejected smo odabrali suprotnu vrijednost.

2. Optimizacija modela

    - Pokrećemo optimizaciju modela s ciljem minimiziranja funkcije gubitka za referentni model, skup podataka i odabrani parametar β. Konkretno, koristeći DPO Trainer potrebno je inicijalizirati objekt DPOTrainer sa svim potrebnim parametrima te nad njim pozvati metodu train.

4.3. Cilj i izvedba[2]

Koristeći istu podlogu kao i u poglavlju RLHF, bazirati ćemo se na koeficijent ukazne KL divergencije. Cilj naše politike je minimizacije funckije gubitka:

$$\mathcal{L}_{\text{DPO}}(\pi_0; \pi_{\text{ref}}) = -\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}} \left[\log \sigma \left(\beta \log \frac{\pi_0(y_w \mid x)}{\pi_{\text{ref}}(y_w \mid x)} - \beta \log \frac{\pi_0(y_l \mid x)}{\pi_{\text{ref}}(y_l \mid x)} \right) \right]$$

Ova formula sastoji se od:

- Razlike skaliranog logaritamskog omjera vjerojatnosti. U formuli su logaritamski omjeri vjerojatnosti za preferirani i nepoželjni ishod pomnoženi s koeficijentom skaliranja β.
- Sigmoide, koja mapira dobivenu skaliranu razliku na raspon od 0 do 1, kako bi se ta vrijednost mogla interpretirati kao vjerojatnost.
- Negativnog logaritma. Negiramo ga jer je logaritam broja između 0 i 1 negativan broj. Logaritam nam govori koliko je preferirani ishod vjerojatniji. Ako je izlaz iz sigmoide blizu 1, to jest ako je preferirani ishod vjerojatniji onda će logaritam biti blizu nuli, to jest izlaz funkcije gubitka bit će minimalan, to smo htjeli i postići.
- Očekivanja, to jest prosjeka preko distribucije podataka $\mathbb{E}_{(x,y_w,y_l)\sim\mathcal{D}}$. Gdje je x stanje, y_w preferirani ishod, a y_l nepoželjni ishod.

Model se ažurira s pomoću gradijenta funkcije gubitka. Gradijent funkcije gubitka s obzirom na parametre modela Θ je:

$$\nabla_\theta L_{DPO}(\pi_\theta; \pi_{ref}) = -\beta \mathbb{E}_{(x,y_w,y_l)\sim D} \left[\sigma\left(\hat{r}_\theta(x, y_l) - \hat{r}_\theta(x, y_w)\right)\left(\nabla_\theta \log \pi(y_w|x) - \nabla_\theta \log \pi(y_l|x)\right)\right]$$

Gradijent funkcije gubitka je skalirano očekivanje za trenutni element skupa podataka koje se sastoji se od umoška:

- Ponderirane razlike implicitnih funkcija nagrade definiranih kao:

  $$\hat{r}_\theta(x, y) = \beta \log \frac{\pi_\theta(y|x)}{\pi_{ref}(y|x)}.$$

  Ove funkcije nagrade implicitno se baziraju na KL divergenciji, koja nam osigurava da se naučeni model neće previše razlikovati od referentnog, te time se umanjuje šansa da se model degradira. Razliku funkcije nagrade za nepoželjni i poželjni odgovor ponderiramo s pomoću sigmoide kako bi dobili uvid u to koliko pogrešno model rangira odgovore.

- Razlike gradijenta logaritamskih vjerojatnosti s obzirom na parametre Θ. Nagib i predznak gradijenta govori nam koliko i kako moramo izmijeniti parametre modela da bi povećali vjerojatnosti poželjnih odgovora dok istovremeno smanjujemo vjerojatnosti nepoželjnih odgovora. Razlika je prisutna kako bi se stvorio jači signal za učenje, osiguravajući da model jasnije razlikuje između poželjnih i nepoželjnih odgovora.

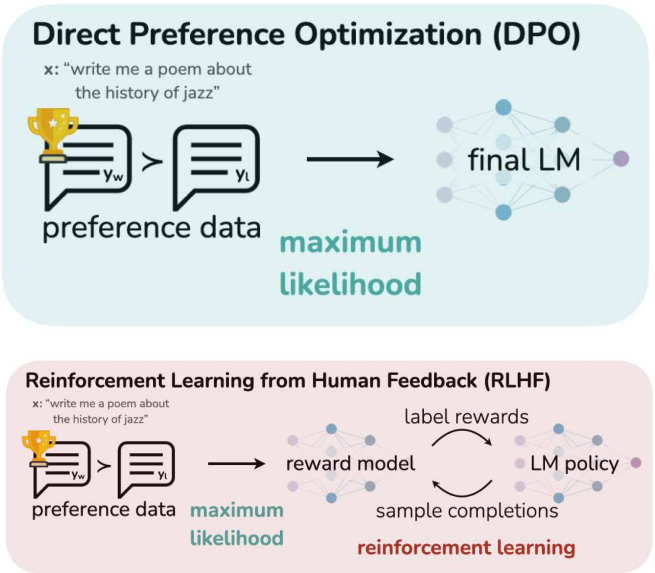
Skalar β određuje koliki će utjecaj ponderirani primjeri imati na učenje modela, te u kombinaciji s KL divergencijom osigurava se efikasno i stabilno učenje modela koji će pružati raznolike i učinkovitije odgovore.

4.4. DPO ili RLHF?

Kroz prijašnja poglavlja opisali smo najvažnije značajke oba algoritma te dobili smo dobar uvid u njihove razlike, no postoji li jasno definirana prednost jednog algoritma nad drugim?

DPO pruža jednostavniju implementaciju i stabilniju implementaciju jer ne zahtijeva učenje u više faza te općenito zahtijeva manje resursa uz bržu podosta bržu konvergenciju s jednako dobrim ili boljim rezultatima od RLHF. U drugu ruku RLHF omogućava puno fleksibilnije i kompleksnije definiranje struktura nagrade što bi moglo dovesti do bolje kontrole nad izlazima jezičnog modela. Također dok se DPO najčešće bazira na binarnim preferencama odgovora, RLHF može primati puno raznovrsnije oblike skupa podataka[7].

Najvažnije je temeljito i s razumijevanjem proučiti problem kojeg rješavamo te na temelju toga odabrati pogodnu metodu jer iako su obje metode slične po tome da poravnavaju model po preferencama povratne informacije ljudi ipak postoje specifične situacije, kao na primjer problem kojeg rješavamo u ovom završnom radu, gdje nema smisla primijeniti određenu metodu, dok druga ima smisla i postiže rezultate.



Slika 4.2. Jednostavna usporedba DPO i RLHF

5. Eksperiment

5.1. Opis problema

Zadatak je bio provesti učenje jezičnog modela tako da točno prepoznaje investicijske prilike na temelju članaka i e-biltena, pod uvjetom da članak i ta vijest nije nešto što se već dobro zna i o čemu se dulje vrijeme priča. Drugim riječima, model mora dovoljno rano i precizno prepoznati da je nešto investicijska prilika. Zadane su bile dvije metode, izravna optimizacija preferenci i podržano učenje s pomoću povratne informacije ljudi. Obje metode služe da usmjere i specijaliziraju veliki jezični model da izvršava neku specifičnu akciju, ili da je vješt s određenim skupom informacija. Specifikacija i izrada skupa podataka objašnjena je u poglavlju "Skup podataka".

5.2. Primjena metoda

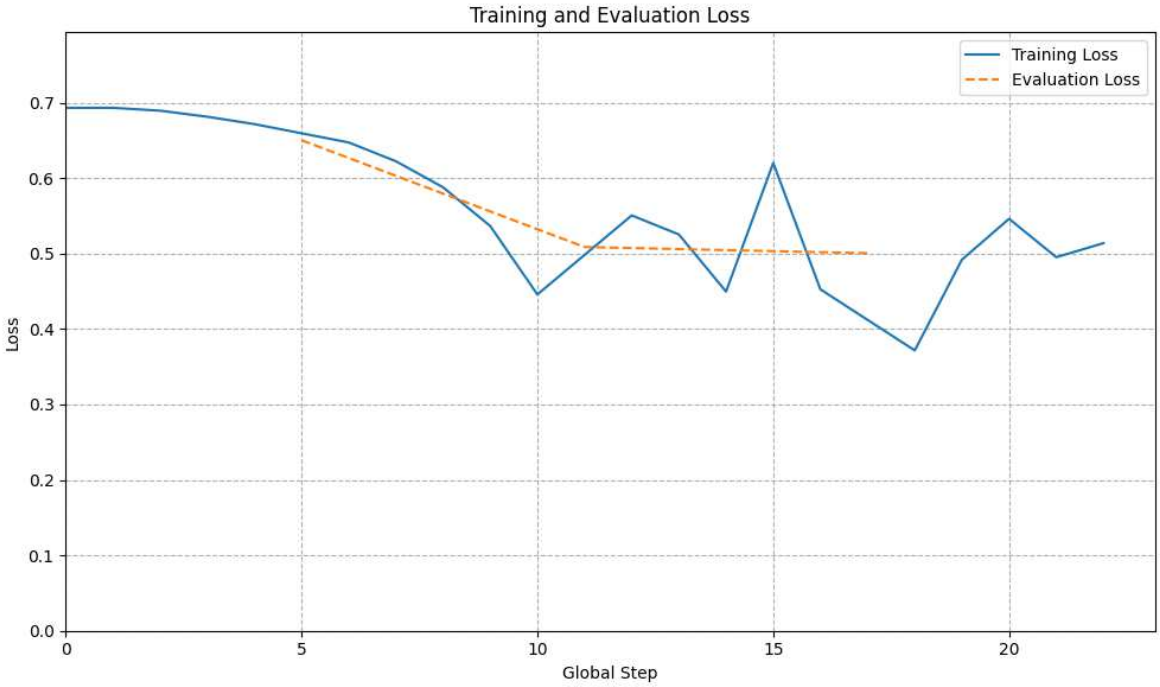
5.2.1. Implementacija metode DPO

Za konkretnu implementaciju i učenje koristeći izravno poravnavanje preferenci koristili smo DPO Trainer iz paketa Transformer Reinforcement Learning (TRL)[8]. Također, koristili smo stranicu Weights and Biases pomoću koje smo pratili napredak našeg učenja te lako i efikasno dobili grafove za razne metrike performanse modela. Okruženje koje sam koristio je Google Colab jer omogućava spajanje na T4 grafičku karticu s pomoću koje smo izvršavali učenje. Model za kojeg sam se odlučio je Mistral 7B v0.1 [9]. Odabran je pretragom LMSYS Chatbot Arena Leaderboarda, uzimajući u obzir omjer performanse i zahtjevnosti resursa, pošto Google Colab ima svoja ograničenja, morao sam paziti da se model može učitati u VRAM memoriju grafičke kartice. Model i njegov tokenizer smo učitali koristeći paket transformers s 4-bitnom kvantizacijom jer je bio prevelik da se u originalnom formatu učita u VRAM grafičke kartice. Skup podataka podijelili smo na train i test u omjeru 90-10. Na weights and biases smo slali napredak svaki korak epohe, a na petom, desetom i petnaestom koraku smo radili evaluaciju, ukupno smo proveli 22 koraka učenja. Ovo nije idealno, ali proces je iznimno vremenski iscrpan, a Google Colab ima ograničenja na maksimalno vrijeme izvođenja s obzirom na količinu slobodnih resursa, što znači da se izvršavanje može terminirati nakon neodređene količine sati bez ikakvog upozorenja. Nakon što smo postavili sve potrebne parametre za konfiguraciju učenja i izravno za DPOTrainer spremni smo za učenje modela koje se izvršava jednostavnim pozivom metode trainer nad objektom DPOTrainer. učenje trajalo je otprilike 10 sati, a najviše vremena potrošeno je na evaluaciju.

5.2.2. Rezultati metode DPO

S Weights and biases preuzeo sam podatke u obliku csv datoteka te u Pythonu samostalno generirao grafove radi bolje preglednosti i kontrole nad prikazanim podatcima.

Gubitak učenja i evaluacije



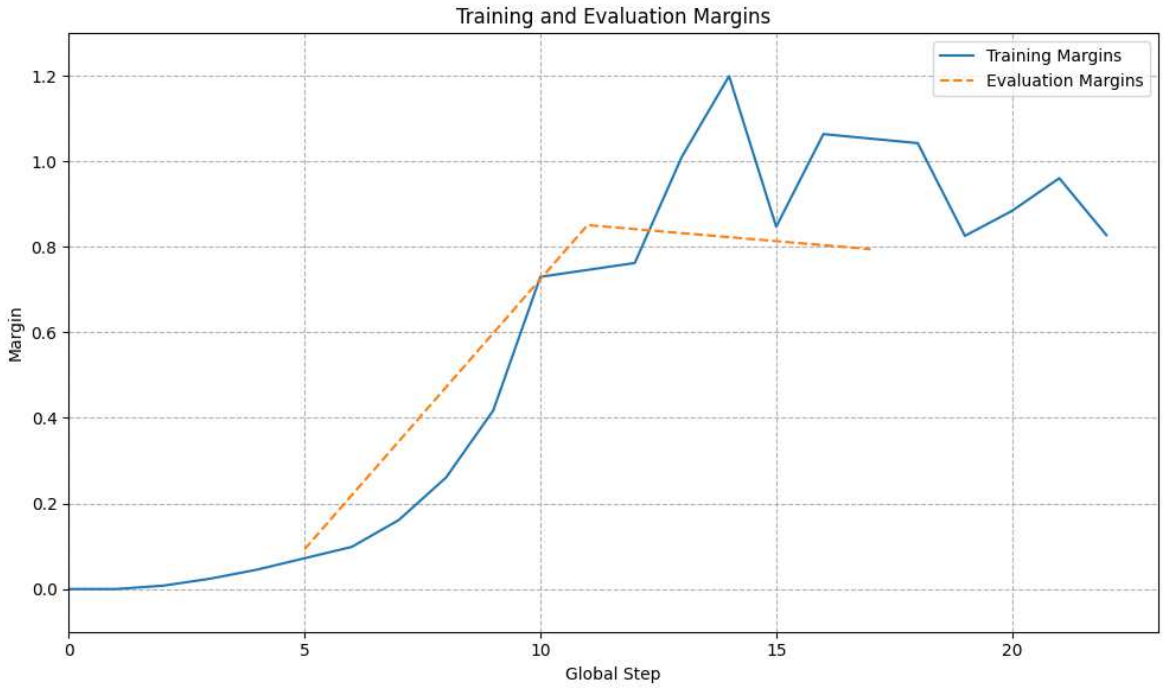
Slika 5.1. Gubitak pri učenju i evaluaciji

Početna vrijednost gubitka učenja je oko 0.7 što nije iznenađujuća vrijednost. Od početnog do 10. koraka primjećujemo stabilan pad gubitka, što je očekivano jer model uči. Između 10. i 15. koraka pojavljuju se podosta drastične oscilacije gubitka, što ukazuje na nestabilnost učenja. Između 15. i 22. koraka postižemo novi globalni minimum u vrijednosti od otprilike 0.38, te nakon toga ponovno imamo oscilacije do kraja učenja. Oscilacije su očekivane, posebno s obzirom na ograničen broj koraka učenja. Jedan od mogućih razloga za oscilacije je premalen batch size, u ovome slučaju jedan, koji nam govori da će model nakon svake iteracije učenja ažurirati svoje težine, što može pridonijeti niskoj stabilnosti učenja jer svaki uzorak neće imati proporcionalan utjecaj na promjene težina modela. S većim brojem koraka gubitak bi se vjerojatno nastavio smanjivati, što bi dovelo do boljih performansa modela.

Prvu evaluaciju napravili smo nakon 5. koraka s gubitkom od otprilike 0.65, zatim primjećujemo pad do 10. koraka te jako blagi pad do poslije 15. koraka koji bi mogao ukazivati na stabilizaciju gubitka Accuracy je brzo porastao te stabilizirao se oko 80%, s finalnom vrijednosti od 78.13%.

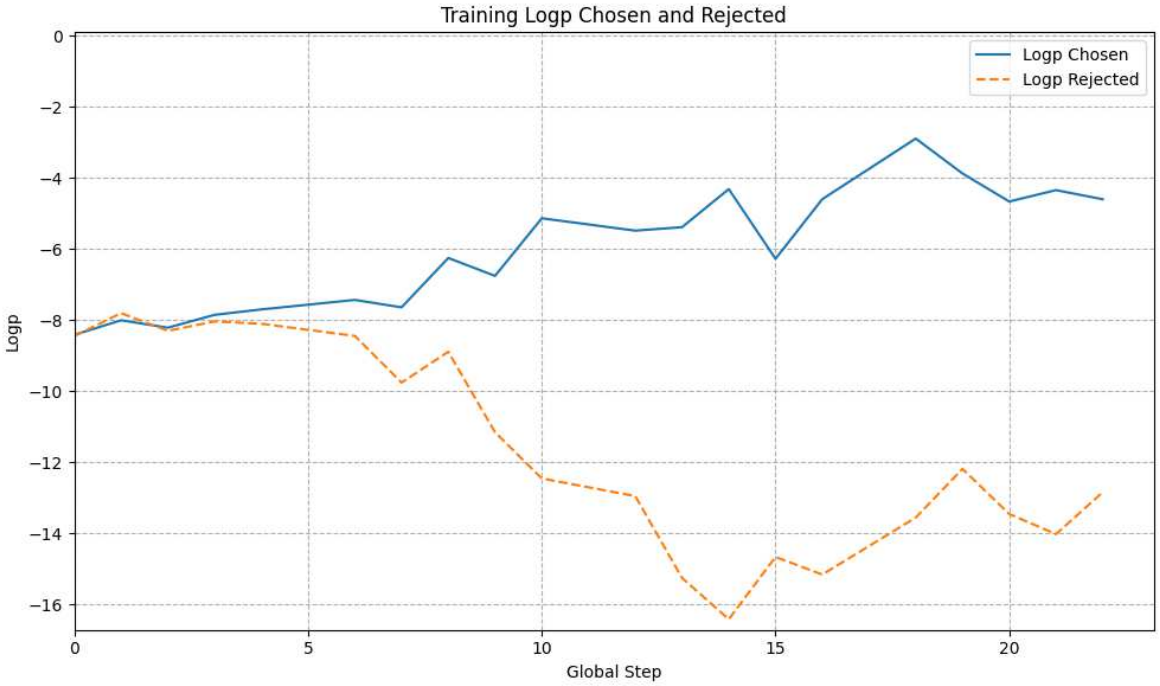
Primjećujemo pad gubitka na oba grafa, što je dobar znak i ono što smo htjeli postići, uzevši u obzir dostupne resurse i mogućnosti.

Dodatne metrike



Slika 5.2. Margine pri učenju i evaluaciji

Margine su razlike između gubitaka za odabrane i odbijene odgovore. Ovaj graf govori nam koliko je jezični model siguran u svoje odluke. Pri početku učenju modela margine su podosta niske, što je očekivano jer model tek kreće učiti. Do 14. koraka prisutan je stabilan rast nakon čega dolazi do oscilacija koje ukazuju na potencijalnu nestabilnost učenja, no unatoč tome margine konstantno osciliraju između vrijednosti 0.8 i 1 što ukazuje da model održava relativnu sigurnost odluka. Kod evaluacije vidimo rast margina do 11. koraka nakon čega nastupa jako blagi pad. Ovo nam govori da model postaje sve sigurniji čak i na podatcima koje nije vidio do sad, no blagi pad može biti indikator prenaučavanja.



Slika 5.3. Logaritamske vjerojatnosti pri učenju

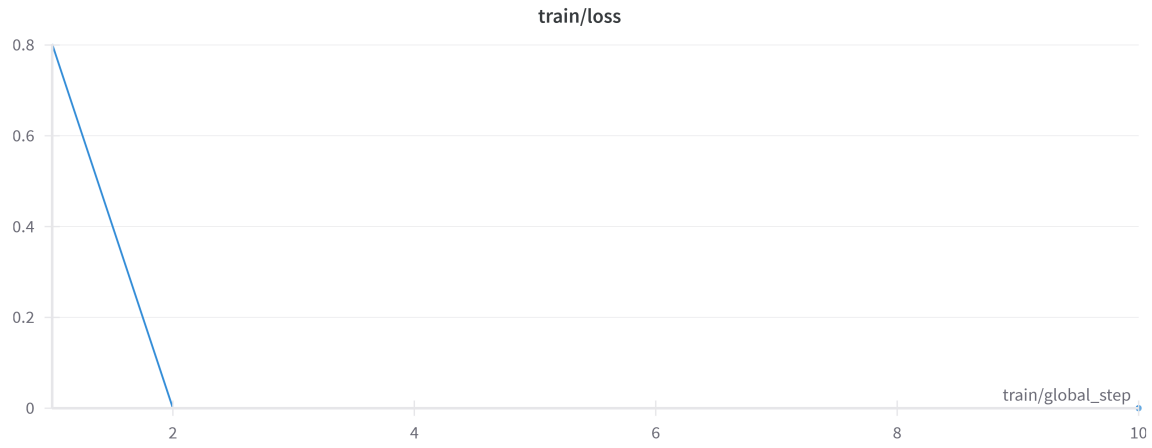
Graf prikazuje logaritamske vjerojatnosti odabira poželjnih i nepoželjnih odgovora tijekom učenja. Na početku vjerojatnosti su iste, što je očekivano jer model tek počinje učiti. Kroz korake vidimo da vjerojatnosti chosen odgovora rastu, a vjerojatnosti rejected odgovora istovremeno padaju. Ovaj graf pokazuje pozitivan napredak u učenju modela.

5.2.3. Izvedba RLHF

Za naš zadatak nema smisla provoditi RLHF metodu. Koristeći RLHF model uči kroz niz interakcija koje vode prema optimiziranom cilju i iznimno je korisna za kompleksne interaktivne zadatke koji zahtijevaju sekvencijalno odlučivanje kao na primjer dijaloški agent.

Naš zadatak je vrlo jednostavna binarna klasifikacija teksta, i za to je DPO prikladna metoda jer preko nje direktno optimiramo model koristeći parove jasno definiranih preferenci. Naši podatci su jasno definirani i dolaze u obliku "signal" ili "nije signal" što je prejednostavna i prestatična struktura podataka za RLHF mehanizme.

Provođenjem učenja nagradnog modela gubitak instantno pada, to se događa jer je zadatak prejednostavan, model nagrade zna savršeno nagraditi odgovore jer su binarne prirode i ne postoji nikakva varijacija - ili je signal ili nije.



Slika 5.4. Gubitak pri učenju modela nagrade

6. Zaključak

Cilj ovog rada bio je provesti podešavanje velikog jezičnog modela za prepoznavanje investicijskih prilika koristeći dvije metode podržanog učenja - izravno poravnavanje preferenci i podržano učenje iz povratnih informacija ljudi. Učenje proveli smo na skupu podataka kojeg smo samostalno izradili koristeći se raznim metodama scrapeanja.

RLHF metoda, koja je iznimno moćan alat za poravnavanje preferenci kod složenih problema koji zahtijevaju sekvencijalno odlučivanje, pokazala se kao nepotrebno komplicirana i neprikladna za naš problem jednostavne binarne klasifikacije. Provođenje ove metode iskazalo se kao nepotreban trošak resursa i vremena.

DPO metoda, u drugu ruku, pokazala se kao preferabilno i jednostavno primjenjivo rješenje za naš problem. Mogućnost izravnog učenja modela s pomoću preferenci bez dodatnih složenosti RLHF metode omogućile su učinkovito podešavanje jezičnog modela koje je rezultiralo značajnim poboljšanjem točnosti modela i njegovoj sigurnosti donošenja odluka što je vidljivo kroz razne metrike prezentirane u ovome radu poput pada vrijednosti funkcije gubitka te rasta i stabiliziranja točnosti.

Iako nismo proveli podešavanje s pomoću obje metode, ovaj rad svejedno je koristan te ukazuje nam na važnost odabira ispravne metode za specifični problem kojeg rješavamo. Iako RLHF ima svoje primjene na određenim domenama problema, DPO se pokazao kao superiornija metoda za naš zadatak binarne klasifikacije. Rad nije bez mana i smatram da je najveća skup podataka. Točnije količina signala u skupu podataka trebala je biti veća kako bi model imao bolju reprezentativnost što je signal, a što nije. Taj problem mogao bi se ukloniti ručnim probiranjem članaka umjesto automatizacijom, no to je vrlo vremenski iscrpan proces. Iz ovoga rada može se zaključiti da je DPO prikladna metoda za učenje modela prema našim preferencama.

Literatura

[1] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, i O. Klimov, "Proximal policy optimization algorithms", 2017.

[2] R. Rafailov, A. Sharma, E. Mitchell, S. Ermon, C. D. Manning, i C. Finn, "Direct preference optimization: Your language model is secretly a reward model", 2023.

[3] A. Gulli, "Ag's corpus of news articles", http://groups.di.unipi.it/~gulli/AG_corpus_of_news_articles.html, accessed: 2024-06-06.

[4] itsayush_k, "Links to all articles of many tech sites (800k)", https://www.kaggle.com/datasets/itsayushk/links-to-all-articles-from-big-tech-news-sites, accessed: 2024-06-06.

[5] T. Kaufmann, P. Weng, V. Bengs, i E. Hüllermeier, "A survey of reinforcement learning from human feedback", 2024.

[6] R. A. Bradley i M. E. Terry, "Rank analysis of incomplete block designs: I. the method of paired comparisons", *Biometrika*, sv. 39, br. 3/4, str. 324–345, 1952. [Mrežno]. Adresa: http://www.jstor.org/stable/2334029

[7] S. Arya, "Dpo vs rlhf: A battle for fine-tuning supremacy in language models", 2023. [Mrežno]. Adresa: https://medium.com/@sinarya.114/d-p-o-vs-r-l-h-f-a-battle-for-fine-tuning-supremacy-in-language-models-04b273e7a173

[8] "Dpo trainer from trl - transformer reinforcement learning", https://huggingface.co/docs/trl/dpo_trainer, accessed: 2024-06-06.

[9] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. de las Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, L. R. Lavaud, M.-A. Lachaux, P. Stock, T. L. Scao, T. Lavril, T. Wang, T. Lacroix, i W. E. Sayed, “Mistral 7b”, 2023.

7. Sažetak

Podešavanje velikih jezičnih modela korištenjem tehnika poravnanja preferenci za identifikaciju investicijskih prilika

Cilj ovog rada bio je istražiti metode podržanog učenja kako bi podesili veliki jezični model za prepoznavanje investicijskih prilika. Fokus je bio na dvije metode: izravno poravnavanje preferenci (DPO) i podržano učenje iz povratnih informacija ljudi (RLHF). Skup podataka izgrađen je koristeći vlastite scraper skripte te filtriranjem i procesiranjem članaka. Skup podataka u završnom obliku imao je oko 35 tisuća elemenata. Metoda DPO se pokazala kao jednostavno i učinkovito rješenje za naš problem, dok je metoda RLHF nepotrebno kompliciranа i prezahtjevna. Rezultati su pokazali poboljšanje modela koristeći metodu DPO. Također je naglašena važnost prepoznavanja i upotrebe pravilne metode za specifični problem koji se rješava.

Ključne riječi: Veliki jezični modeli, podržano učenje, izravno poravnavanje preferenci, podržano učenje iz povratne informacije ljudi, binarna klasifikacija, prepoznavanje investicijskih prilika

8. Abstract

Fine-tuning large language models using preference alignment techniques for identification of financial investment opportunities

The goal of this bachelors thesis was to explore methods of reinforcement learning to fine-tune a large language model to recognize financial investment oppportunities. The focus was on two methods: direct preference optimisation (DPO) and reinforcement learning from human feedback (RLHF). The datawas was built using self-made scraper tools and methods of filtering and processing the articles. The final version of the dataset had around 35 thousand articles. DPO showed itself to be a very simple, useful and efficient method to implement, while RLHF turned out to be too complex and taxing. The results showed that the model was learning using the DPO method and that it is crucial to recognize and pick a method which aligns itself with our goals and problems.

Key words: Large language models, reinforcement learning, direct preference optimisation, reinforcement learning from human feedback, binary classification, recognition of investment opportunities