**Univerzitet u Nišu**
**Elektronski fakultet**
**Katedra za računarstvo**

Seminarski rad iz predmeta Duboko učenje

# Preneseno učenje

**Profesor:** Prof. Dr Milosavljević Aleksandar

**Studenti:** Nikola Petrović 1466,
Stevan Grujić 1493

**Niš, 2023.**

# Sadržaj

# 1 Uvod

Veštačka inteligencija je veoma širok pojam, uključuje ogroman skup oblasti i ide u raznim smerovima, na primer NLP, DL, ML... Sve ove oblasti imaju nešto zajedničko, a to je težnja ka simuliranju prave inteligencije. Što nas vodi ka pitanju šta je zapravo inteligencija?

Izvori sa interneta kažu sledeće:
"Veoma opšta mentalna sposobnost, koja, između ostalog, uključuje sposobnost rasuđivanja, planiranja, rešavanja problema, apstraktnog razmišljanja, razumevanja složenih ideja, brzog učenja i učenja iz iskustva. To nije samo učenje iz knjiga, uska akademska veština ili uspešnost u polaganju testova. Umesto toga, ono odražava širu i dublju sposobnost za razumevanje našeg okruženja – 'hvatanje', 'smišljanje' stvari ili 'shvatanje' šta da radimo."

O ovom pitanju se može mnogo govoriti, istraživati i zaključivati ali se sa sigurnošću može tvrditi da su neke od karakteristika inteligencije, brzo učenje, učenje iz iskustva i povezivanje stvari. Veštačka inteligencija je na dobrom putu da simulira ove karakteristike. Brzo učenje se ostvaruje kroz kombinaciju jakih hardverskih karakteristika i optimalnih algoritama, učenje iz iskustva se postiže propagacijama unapred i unazad, povezivanje stvari kroz definisanje određenih pravila i uočavanje šablona. Da li postoji spoj ovih karakteristika? U poslednje vreme se pojavilo mnogo alata koji korišćenjem veštačke inteligencije uspevaju da nadmaše rad mnogih stručnjaka u određenim oblastima kao što su medicina, arhitektura, matematika.. Zaključak je da stvari funkcionišu dobro i da se sve brže i brže razvijaju. Za ovakve zadatke i rešenja, neophodne su ogromne količine podataka, ispravnih dijagnoza, urađenih zadataka, definisanih pravila, ovi alati sve to imaju, međutim, podaci nastaju iz dana u dan, proširuju se, nadograđuju, bez obzira na tačnost koja je postignuta i brzinu kojom se dobijaju rešenja, njihove sisteme svakog trenutka pobeđuje vreme, tačnije ažurnost informacija kojima raspolažu. Takođe jedan od većih problema predstavlja i nedostatak informacija o nekim oblastima, nedovoljno podataka za treniranje modela koje vodi ka nekonzistentnosti modela i slabijim karakteristikama.

Kako se rešavaju ovi problemi? Da bi ažurirali podatke u modelima potrebno je dodati nove podatke starim, zatim ponovo istrenirati modele i primeniti ih, ali vremenski gledano, u odnosu na količinu podataka koja se obrađuje, a količina je ogromna, jasno je da ovi sistemi ne mogu biti u koraku sa vremenom, jer treniranje takvih modela iziskuje dosta vremena. Pritom svaki model se trenira zasebno, bilo bi idealno spojiti znanja više modela, kako bi modeli bili sposobni za različite zadatke, ali i spojili znanja iz raznih oblasti. Sve ovo nas navodi na razmišljanje kako spojiti sve ove karakteristike, na brz, efikasan način, sa minimalnim vremenskim zahtevima, dok će se iskorišćenje znanja maksimizovati. Nedostatak podataka utiče na kvalitet modela, bilo bi idealno kada bi se na neki veći model koji sadrži skup nekih znanja, dodalo minimalno znanje o oblasti koja je od interesa, rezultati bi bili bolji. Postoji stara izreka u drevnoj Kineskoj knjizi "Analekti Konfučija":

"Ako čovek nastavi da neguje svoja stara znanja, a da neprestano stiče nova, on može biti učitelj drugih." – Konfučije

Ova rečenica implicira da često možemo dobiti novo razumevanje starog znanja ako ga pregledamo pre nego što naučimo novo. Oslanjajući se na ovu sposobnost, možemo postati učitelji. Štaviše, ova rečenica nam govori da nova znanja i sposobnosti ljudi često evoluiraju iz starih. Stoga, ako možemo pronaći veze i sličnosti između starog i novog, proces učenja novog znanja može postati mnogo efikasniji i efektivniji.

Dakle, kako možemo efikasno da iskoristimo sličnost između stvari da nam pomogne da rešimo nove probleme ili steknemo nove sposobnosti, a istovremeno se približimo pravoj veštačkoj inteligenciji?

Rešenje je preneseno učenje (eng. **Transfer Learning**).

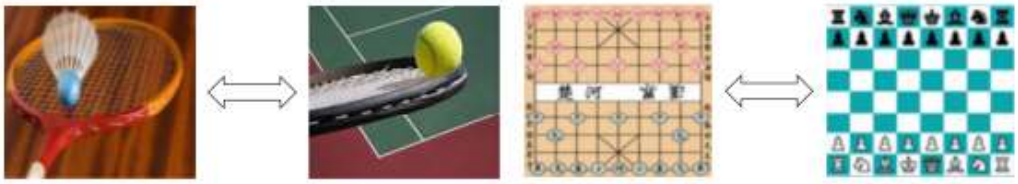
U ovom radu ce biti opisan princip rada prenesenog učenja, fino podešavanje, GPT (eng. Generative Pretrained Transformers). Biće reči i o analizi sentimenata kao podoblasti NLP-a. U praktičnom delu će biti opisan način pripreme podataka za fino podešavanje GPT modela, kao i odrađen primer analize sentimenata nad malim skupom podataka kako bi se reprezentovala snaga prenesenog učenja.

# 2 Transfer Learning

## 2.1 Osnovni koncepti

Princip u mašinskom učenju koji se zasniva na primeni već stečenih znanja za nove zadatke, predstavlja svojevrsnu nadogradnju znanja. Može se reći da predstavlja deo evolucije veštacke inteligencije ka simulaciji realne inteligencije, obzirom na karakteristike koje poseduje i na principe koje primenjuje.

Postoji puno primera iz realnog života koji su slični prenesenom učenju. Na primer, ako možemo da igramo badminton, onda možemo naučiti da igramo tenis, jer igranje badmintona i tenisa imaju slične strategije i trikove. Ako možemo da igramo kineski šah, onda možemo naučiti da igramo međunarodni šah pozajmljujući slična pravila između njih. Ako možemo da vozimo bicikl, onda možemo naučiti da vozimo motocikl pošto su ove dve radnje veoma slične. Takođe, profesionalni atletičar se vrlo lako može opredeliti za neki sport kao sto su trčanje, bacanje koplja, skok u dalj, razlog tome jeste upravo iskorišćavanje već postojeće fizičke spreme i  načina treninga koji se sa malim modifikacijama može prilagoditi novom sportu. Iznenađujuće je otkriti da koristeći sličnost između dve stvari, možemo izgraditi most preko kojeg se staro iskustvo ili znanje može preneti na novo, od koristi procesu učenja novog znanja.



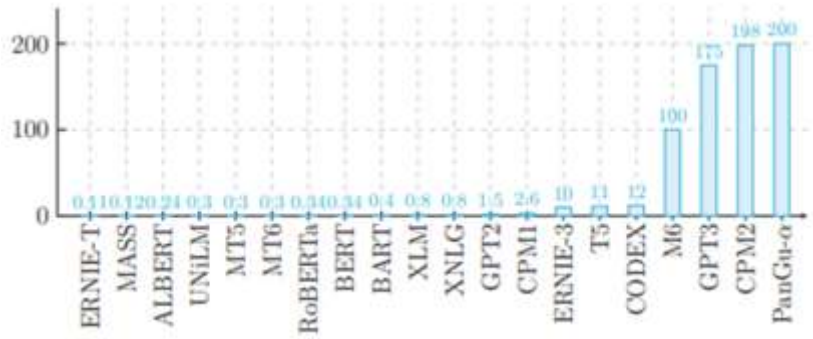
(a) (b)

*Slika 1. Transfer Learning u realnom životu a) badminton – tenis b) kineski šah – internacionalni šah*

Glavni izazov mašinskog/dubokog učenja u mnogim aplikacijama je to što modeli ne funkcionišu dobro u novim domenima zadataka. Razlog zašto ne funkcionišu dobro može biti jedan od nekoliko razloga: nedostatak novih podataka o obuci zbog malog izazova podataka, promena okolnosti i promena zadataka. Na primer, u novoj situaciji, visokokvalitetni podaci o obuci mogu nedostajati i vrlo cesto je nemoguće obezbediti još podataka kao u slučaju medicinske dijagnoze i podataka medicinskog snimanja. Modeli mašinskog učenja ne mogu da rade dobro bez dovoljno podataka o obuci. Dobijanje i označavanje novih podataka često zahteva mnogo truda i resursa u novom domenu aplikacije, što je glavna prepreka u realizaciji veštačke inteligencije u stvarnom svetu.

Razlozi za razvijanje prenesenog učenja su sledeći:

- Velike količine podataka i nedovoljno snažni hardverski resursi - Opšte je poznato da su ovi veliki podaci u svakodnevnom životu često potrebni veoma moćni računarski uređaji za čuvanje i izračunavanje. Nažalost, super računari su često ograničeni na one „super bogate“ kompanije kao što su Google, Meta, Microsoft, Amazon, itd., koje obični istraživači ne mogu priuštiti. Ovi tehnološki giganti imaju mnogo veće računarske moći za obuku veoma velikih modela za svoje proizvode i usluge. Na primer, u oblasti kompjuterskog vida, obuka ImageNet-a je izuzetno dugotrajna na normalnom hardveru, a u oblasti obrade prirodnog jezika, obuka BERT modela from scratches nije dostupna većini istraživača. Naučno istraživanje je dug put, kako bismo očekivali da opšti istraživači mogu imati koristi od takve istraživačke teme sa ograničenim resursima? Slika 2. pokazuje sve veće veličine unapred obučenih jezičkih modela tokom proteklih godina. Jasno vidimo da ovi modeli postaju sve veći i veći, zahtevajući mnogo moćniji hardver za obuku. Dakle, u ovom slučaju, kako većina običnih istraživača i studenata može dobro iskoristiti tehnološko otkriće (npr. unapred obučene modele) za sopstveno istraživanje? Jedan od verovatnih odgovora je Transfer Learning, koje nam omogućava da koristimo unapred obučene modele da olakšamo sopstveni zadatak i koristimo u našim sopstvenim istraživanjima.




*Slika 2. Veličine obučenih jezičkih modela (u milijardama parametara)*

- Nedovoljno podataka i nemogućnost generalizacije - Ključno očekivanje od mašinskog učenja je da se obuće modeli da prave ispravna predviđanja za ne već viđene skupove podataka, okruženja i aplikacije u budućnosti, tj. generalizaciju ili generalizaciju van distribucije. Zatim, postoji još jedan izazov, to je ograničeni podaci naspram zahteva generalizacije. Iako možemo da prikupimo više podataka, oni su uvek konačni i ograničeni u poređenju sa ogromnim okeanom velikih podataka. Model i dalje može patiti od novog okruženja. Uzimamo medicinsko mašinsko učenje kao primer. Poznato je da su medicinski podaci izuzetno ograničeni zbog nekoliko mogućih razloga kao što su ograničen broj pacijenata, komplikovani eksperimenti i operacije, kao i neprištivi troškovi istraživanja sopstvenih podataka. U ovom slučaju, ako možemo da iskoristimo prednosti prenesenog učenja da bismo izgradili modele koji se mogu generalizovati na osnovu ograničenog skupa podataka, mogli bismo završiti obuku za model koji ima dobru sposobnost generalizacije, tj. naš model može dobro predvideti nove podatke zasnovane na starim modelima.

- Modeli mašinskog učenja moraju da budu robusni - tradicionalno mašinsko učenje često pretpostavlja da su i podaci o obuci i testu izvučeni iz iste distribucije. Međutim, ova pretpostavka je previše jaka da bi se održala u mnogim praktičnim scenarijima. U mnogim slučajevima, distribucija varira u zavisnosti od vremena i prostora, i varira od situacije do situacije, tako da možda nikada nećemo imati pristup novim podacima o obuci za istu distribuciju testova. U situacijama koje se razlikuju od podataka za obuku, obučenim modelima je potrebna adaptacija pre nego što se mogu koristiti.

- Personalizacija i specijalizacija su važna pitanja - kritično je i isplativo ponuditi personalizovanu uslugu za svakog korisnika prema individualnim ukusima i zahtevima. U mnogim aplikacijama u stvarnom svetu možemo prikupiti vrlo malo ličnih podataka od pojedinačnog korisnika. Kao rezultat toga, tradicionalne metode mašinskog učenja pate od problema sa hladnim startom (eng. cold start) kada pokušavamo da prilagodimo opšti model specifičnoj situaciji. Dakle, kako dozvoliti da se opšti modeli mašinskog učenja prilagode različitim ličnim potrebama? Nemoguće je obučiti model za svaku osobu, što je i daleko skuplje. Možemo koristiti preneseno učenje da izvršimo prilagođavanje modela ili fino podešavanje (eng. fine-

tuning) na osnovu opštih modela. Nakon toga, takođe možemo iskoristiti Transfer Learning da izvršimo prenos znanja na mreži kako bismo promenili modele u skladu sa potrebama svake osobe.

- Privatnost korisnika i bezbednost podataka su važna pitanja - često u našim aplikacijama moramo da radimo sa drugim organizacijama koristeći više skupova podataka. Često ovi skupovi podataka imaju različite vlasnike i ne mogu se otkriti jedni drugima iz razloga privatnosti ili bezbednosti. Kada zajedno gradimo model, bilo bi poželjno da izdvojimo „suštinu" svakog skupa podataka i prilagodimo ih u izgradnji novog modela. Na primer, ako možemo da prilagodimo opšti model na „edge" mreže uređaja, onda podaci uskladišteni na uređaju ne moraju da se učitavaju da bi se poboljšao opšti model, tako se može obezbediti privatnost edge uređaja [1].
-

| Contradiction | Traditional ML | Transfer learning |
|---|---|---|
| Big data vs. less annotation | Expensive human labeling | Transfer knowledge from existing fields |
| Big data vs. poor computation | Exclusive computation device | Model transfer |
| Limited data vs. generalization ability | Poor performance for OOD data | Domain generalization, meta-learning, etc. |
| Pervasive vs. personal need | Cannot satisfy personal needs | Adapt to every person |
| Specific applications | Cold start cannot be solved | Data transfer from other fields |

*Slika 3. Razlozi za razvijanje prenesenog učenja* [2]

Preneseno učenje ima za cilj da reši novi problem korišćenjem sličnosti podataka (zadatka ili modela) između starog i novog problema da bi se izvršio transfer znanja (iskustva, pravila, itd.) između njih.

Preneseno učenje se odnosi na paradigmu mašinskog/dubokog učenja u kojoj algoritam izvlači znanje iz jednog ili više scenarija aplikacije kako bi pomogao da se poboljša učinak učenja u ciljnom scenariju. U poređenju sa tradicionalnim mašinskim učenjem, koje zahteva velike količine dobro definisanih podataka o obuci na ulazu, preneseno učenje se može shvatiti kao nova paradigma učenja. Preneseno učenje je takođe motivacija za rešavanje takozvanih problema oskudnosti podataka i *cold start* u mnogim velikim i onlajn aplikacijama (npr. označenih podataka o oceni korisnika u onlajn sistemima preporuka može biti premalo da bi omogućili ovim onlajn sistemima da izgrade sistem preporuka kvaliteta). Preneseno učenje može pomoći u promovisanju veštačke inteligencije u manje razvijenim oblastima primene, kao i u manje tehnički razvijenim oblastima, čak i kada u takvim oblastima nije dostupno mnogo označenih podataka.

| Aspect | Traditional machine learning | Transfer learning |
|---|---|---|
| Data distribution | Training and test data are i.i.d. | Training and testing data are non-i.i.d. |
| Data annotation | Huge amount of annotations | Less annotations |
| Model | Train from scratch for every task | Model transfer between tasks |

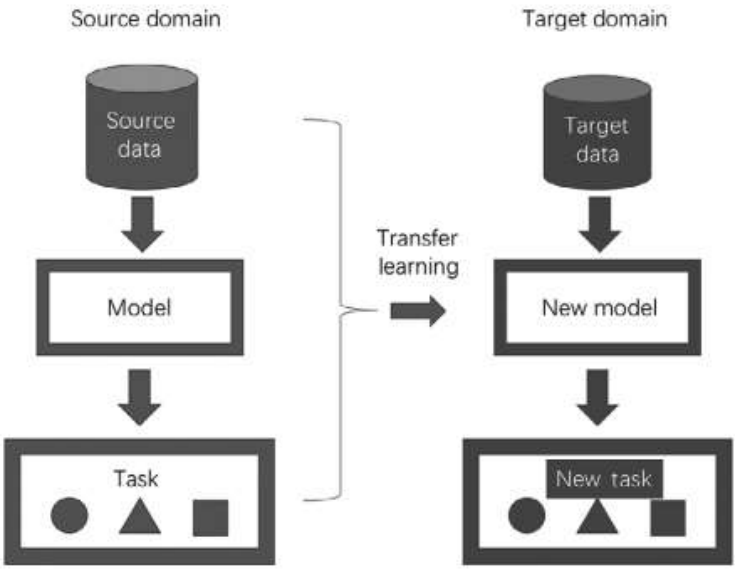
*Slika 4. Razlika između tradicionalnog mašinskog učenja i prenesenog učenja* [2]

Na osnovu metodologija prenesenog, kada dobijemo dobro razvijen model u jednom domenu, možemo koristiti ovaj model u korist drugih sličnih domena. Zbog toga je neophodno imati tačnu meru „udaljenosti“ između bilo kojih domena zadatka u razvoju dobre metodologije Transfer Learning-a. Ako je razdaljina između dva domena velika, onda možda nećemo želeti da primenjujemo preneseno učenje, jer učenje može da proizvede negativan efekat. S druge strane, ako su dva domena „blizu“, preneseno učenje se može plodonosno primeniti. U mašinskom učenju, razdaljina između domena se često može meriti u smislu karakteristika koje se koriste za opisivanje podataka. U analizi slike, karakteristike mogu biti pikseli ili zakrpe u uzorku slike, kao što su boja ili oblik. U NLP-u, karakteristike mogu biti reči ili fraze. Kada znamo da su dva domena blizu jedan drugom, možemo osigurati da se AI modeli mogu širiti od dobro razvijenih domena do manje razvijenih domena, čineći primenu AI manje zavisnom od podataka. A ovo može biti dobar znak za uspešne aplikacije za učenje prenosa.

Mogućnost prenošenja znanja iz jednog domena u drugi omogućava sistemima mašinskog/dubokog učenja da prošire svoj opseg primenljivosti izvan prvobitnog kreiranja. Ova sposobnost generalizacije pomaže da veštačka inteligencija bude pristupačnija i robusnija u mnogim oblastima gde AI talenti ili resursi kao što su računarska snaga, podaci i hardver mogu biti oskudni. Na neki način, preneseno učenje omogućava promociju AI kao inkluzivnije tehnologije koja služi svima.

## 2.2 Definicija prenesenog učenja

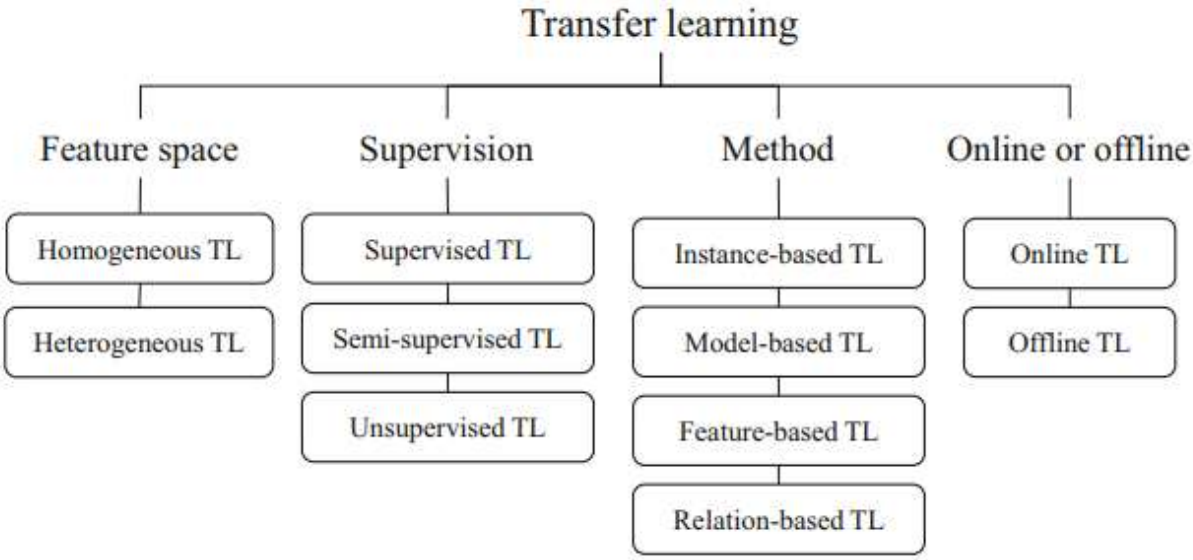
**Definicija 1.** Uzimajući u obzir izvorni domen $D_s$ i zadatak $T_s$, ciljni domen $D_t$ i zadatak učenja $T_t$, preneseno učenje ima za cilj da pomogne u poboljšanju učenja ciljne funkcije predviđanja ft(.) za ciljni domen koristeći znanje iz $D_s$ i $T_s$, gde $D_s \neq D_t$ ili $T_s \neq T_t$.




*Slika 5. Proces prenesenog učenja*

Proces prenesenog učenja je ilustrovan na slici 5. Proces sa leve strane odgovara tradicionalnom procesu mašinskog učenja. Proces sa desne strane odgovara procesu prenesenog učenja. Kao što vidimo, preneseno učenje koristi ne samo podatke u ciljnom domenu zadatka kao ulaz u algoritam učenja, već i bilo koji proces učenja u izvornom domenu, uključujući podatke o obuci, modele i opis zadatka. Ova slika pokazuje ključni koncept prenesenog učenja: on se suprotstavlja problemu nedostatka podataka o obuci u ciljnom domenu sa više znanja stečenog iz izvornog domena.

## 2.3 Taksonomija prenesenog učenja




*Slika 6. Taksonomija prenesenog učenja*

Uobičajeni pristup istraživanju prenesenog učenja je **prostor karakteristika** (eng. Feature space). Prema ovoj taksonomiji, preneseno učenje se može kategorisati u dve glavne klase:

1. Homogeno preneseno učenje
2. Heterogeno preneseno učenje

Ovo je prilično intuitivna taksonomija, ako su semantika karakteristika i dimenzije iz različitih domena iste, onda to pripada homogenom transfernom učenju, inače spada u heterogeno preneseno učenje. Na primer, preneseno učenje između različitih domena slike je homogen transfer, dok se preneseno učenje sa slike na tekst smatra heterogenim zadatkom

Slično kategorizaciji mašinskog učenja, preneseno učenje se može kategorizovati u tri klase **prema labelama u ciljnom domenu**:

1. Nadgledano (eng. Supervised) preneseno učenje
2. Polu-nadgledano (eng. Semi-supervised) preneseno učenje
3. Nenadgledano (eng. Unsupervised) preneseno učenje

Očigledno je da su situacije sa malim resursima (tj. polu-nadgledano ili nenadgledano preneseno učenje) izazovnije i stoga jedna od najpopularnijih istraživačkih tema u ovoj oblasti.

Taksonomija putem učenja na mreži ili van mreže. **Prema šemi učenja**, preneseno učenje se može kategorisati u dve klase:

1. Offline preneseno učenje
2. Online preneseno učenje

Trenutno, većina algoritama i aplikacija za preneseno učenje usvaja offline šemu, koja se odnosi na situaciju da su izvorni i ciljni domeni dati unapred. Ovoj šemi nedostaje fleksibilnost učenja iz više onlajn podataka, za šta verujemo da bi trebalo da bude buduća faza u kojoj model može da vrši ažuriranje na mreži kada stignu novi podaci. Online preneseno učenje bi trebalo da bude budućnost ove oblasti.

**Taksonomija prema metodologiji učenja**. Pošto je naš fokus tehnologija učenja koja stoji iza prenesenog učenja, onda je prirodna šema klasifikacije kategorizacija postojećih algoritama prenesenog učenja prema metodologiji učenja:

1. Preneseno učenje zasnovano na instanci (eng. *Instance-based*)
2. Preneseno učenje zasnovano na karakteristikama (eng. *Feature-based*)
3. Preneseno učenje zasnovano na modelu (eng. *Model-based*)
4. Preneseno učenje zasnovano na relacijama (eng. *Relation-based*)

Ova kategorizacija je izvedena nizom instanci, karakteristika i modela, što je sasvim očigledno, što je još važnije i relacije visokog nivoa između domena se takođe mogu smatrati važnom kategorijom.

Preneseno učenje zasnovano na **instanci** ima za cilj da izvrši transfer znanja putem ponovnog menjanja tezine instance ili ponovno izvlačenje uzoraka. Dakle, možemo dati različite težine različitim instancama. Na primer, ako je instanca sličnija podacima našeg ciljnog domena, daćemo joj veću težinu nego drugim instancama. Ova ideja je sasvim prirodna i jednostavna.

Preneseno učenje zasnovano na **karakteristikama** vrši transfer znanja koristeći transformaciju karakteristika ili učenja reprezentacija. Pretpostavimo da karakteristike izvornog i ciljnog domena nisu u istom prostoru karakteristika, kako onda možemo iskoristiti učenje reprezentacije da ih transformišemo u isti, gde su njihove reprezentacije karakteristika slične. Ova kategorija metoda je prilično popularna i u akademskim i industrijskim oblastima.

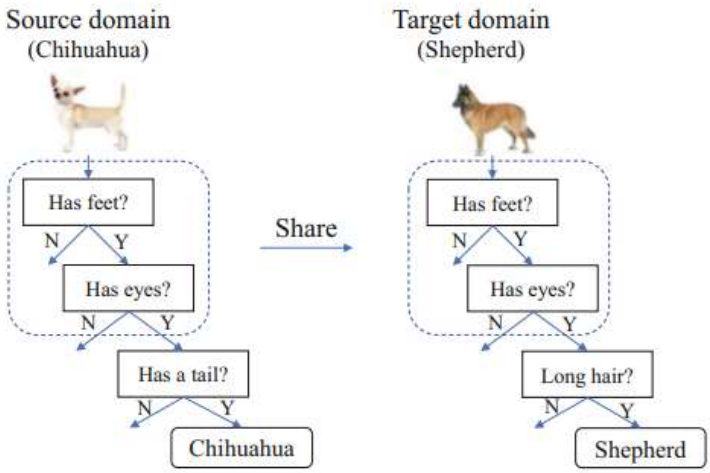
*Model-based transfer learning* (preneseno učenje zasnovano na **modelima**) vrši prenos znanja deljenjem parametara između modela iz različitih domena. Parametri težine (eng. weight) jednog modela mogu biti deljeni sa drugim modelom, kao što su težine i konstante (eng. *bias*) neuronske mreže. Konkretno za model neuronske mreže, njegove težine mogu se lako preneti korišćenjem šeme prethodnog treniranja i finog podešavanja (eng. *pre-train-fine-tune*), koja je veoma popularna. I nama je od interesa zbog toga sto se primenjuje kod dubokih neuronskih mreža.

## 2.4 Duboke neuronske mreže i preneseno učenje

Nakon dokazivanja korisnosti prenesenog učenja u veštačkoj inteligenciji uopšteno, jasno je da je sledeći korak polje konkretnije primene i polje koje će u budućnosti biti sve više razvijano i istraživano, a to je preneseno učenje u dubokom učenju, tj. u dubokim neuronskim mrežama.

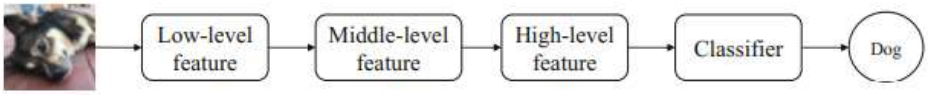
### 2.4.1 Preneseno učenje zasnovano na modelu i duboke neuronske mreže

Kako bismo mogli da pričamo o metodama prenesenog učenja u dubokim neuronskim mrežama, moramo obraditi neizbežne teme bez kojih savremeno preneseno učenje ne može da funkcioniše, a to su **pre-training** i **fine-tuning**.




*Slika 7. Ilustracija Model-based transfer learning-a*

Pretpostavimo da je ulaz pas. U procesu propagacije unapred, mreže uče samo neke informacije, kao sto su ivice, o psu na osnovnim slojevima (eng. *low-level*), koje nazivamo karakteristikama niskog nivoa. Zatim, mreže počinju da otkrivaju neke linije i oblike na srednjim slojevima, koji su očigledniji od karakteristika niskog nivoa i njih nazivamo karakteristikama srednjeg nivoa (eng. *middle-level*). Konačno, mreže mogu otkriti semantičke karakteristike psa kao što su lice ili nos.. koje su karakteristike visokog nivoa (eng. *high-level*).




*Slika 8. Kako duboke neuronske mreže ekstrahuju fičere*

Kako možemo iskoristiti takvo zapažanje da bismo razumeli **transfer learning dubokih mreža**?
Popularno objašnjenje je sledeće: U dubokoj neuronskoj mreži, osnovni (low-level) slojevi uče opšte karakteristike (npr. informacije o ivici i obliku), dok njeni dublji slojevi uče specifične karakteristike (npr. noge i lice). Kako mreže idu dublje, karakteristike idu od opštih ka konkretnijim. Ovo je prilično intuitivno objašnjenje i lako je razumeti. Ako možemo tačno da znamo koji slojevi u mreži uče opšte karakteristike, a koji specifične karakteristike, onda možemo da koristimo takvo svojstvo za obavljanje prenesenog učenja. Pošto slojevi za učenje opštih karakteristika nisu ograničeni na specifične zadatke, možemo ih koristiti za uobičajene osnovne zadatke. Na primer, na slici 7., možemo direktno koristiti osnovne slojeve za učenje karakteristika niskog nivoa za mačke pošto su ove karakteristike takođe opšte za mačke. Na ovaj način možemo smanjiti troškove obuke za klasifikator mačaka.

Kako možemo odrediti koji su slojevi za karakteristike niskog nivoa, a koji slojevi za specifične karakteristike?

Predloženo je nekoliko koncepata za bolje tumačenje rezultata finog podešavanja, a to su AnB, BnB, AnB+ i BnB+:

- AnB se koristi za testiranje performansi prenosa prvih n slojeva iz mreže A u mrežu B. Na početku je dobijeno prvih n slojeva unapred obučene mreže A. Zatim su kopirani parametri ovih n slojeva od A do B. Zatim je zamrznuto prvih n slojeva za B i obučeno samo preostalih 8 − n slojeva.

- BnB se koristi za testiranje performansi mreže B. Konkretno, zamrznuto je njenih prvih n slojeva i obučeno ostalih 8 − n slojeva. Ovo je za poređenje sa AnB.

- AnB+ i BnB+ označavaju situaciju da nije zamrznuto prvih n slojeva B, već je izvršeno fino podešavanje (eng. fine-tuning) na ovim slojevima.

Dobijeni su sledeći zaključci:

- Što se tiče BnB-a, oni mogu direktno da koriste prva tri sloja za preneseno učenje. Kada su u pitanju četvrti i peti sloj, tačnost opada. Objašnjeno je da karakteristike na četvrtom i petom sloju postaju sve konkretnije, pa tačnost opada. Za BnB+, otkriveno je da fino podešavanje zapravo pomaže u postizanju boljih performansi.

- Što se tiče AnB, ne šteti performansama B ako direktno kopiramo prva tri sloja iz A u B, što ukazuje da prva tri sloja uče opšte karakteristike. Ali tačnost opada sa četvrtog i petog sloja, što znači da karakteristike nisu opšte.

- Što se tiče AnB+, njegove performanse su najbolje kada je dodato fino podešavanje. **Ovo implicira da prethodna obuka i fino podešavanje zaista povećavaju performanse dubokog učenja.**

Ovo istraživanje pokazuje da niži slojevi duboke mreže uglavnom izdvajaju opšte karakteristike, dok viši slojevi izdvajaju specifične karakteristike. Što je još važnije, istraživači su istakli da sličnost domena igra ključnu ulogu u dubokom prenesenom učenju: što su domeni sličniji, to su bolje performanse prenosa.
Konačno, opšti zaključak o prenesenom učenju u dubokim neuronskim mrežama je:

- *Niži slojevi duboke mreže izdvajaju opšte karakteristike koje se mogu koristiti za preneseno učenje.*
- *Fino podešavanje zaista pomaže mreži da postigne bolje performanse.*

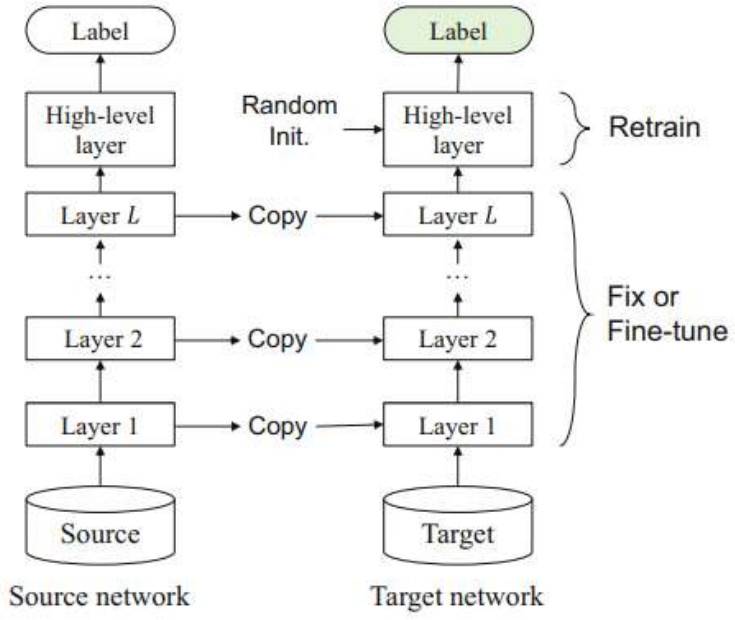
## 2.5 *Pre-training i fine-tuning*

U ovom delu fokusiramo se na savremene metode zasnovane na parametrima: pre-training i fine-tuning. Metode dubokog učenja se fokusiraju na to kako da dizajniraju bolje mrežne arhitekture i funkcije gubitaka na osnovu unapred obučene mreže. Prethodna obuka i fino podešavanje spadaju u kategoriju metoda prenesenog učenja zasnovanih na parametrima/modelima koje vrše prenos znanja deljenjem nekih važnih informacija o strukturama modela. Osnovna pretpostavka je da postoje neke zajedničke informacije između izvornih i ciljnih struktura koje se mogu deliti, tj. između modela koji se trenira i modela koji se koristi kao osnova za prenos znanja.

**Definicija 2.** (Pre-training i fine-tuning) S obzirom na ciljni skup podataka $D$ sa ograničenim označenim podacima, pre-training i fine-tuning imaju za cilj da nauče funkciju $f$ parametrizovanu sa $\theta$ koristeći prethodno znanje (parametar) $\theta_0$ iz istorijskih zadataka:

$$\theta^* = \arg\min_{\theta} \mathcal{L}(\theta|\theta_0, \mathcal{D}).$$

Prethodna obuka i fino podešavanje se razlikuju od prilagođavanja domena jer nije potrebno da izvorni i ciljni domeni imaju identične kategorije. U stvari, u većini aplikacija za prethodnu obuku i fino podešavanje njihove kategorije nisu iste. U stvarnim situacijama često ne treniramo od nule za novi zadatak, što je izuzetno zamorno i skupo, posebno u slučaju kada označeni podaci o obuci nisu dovoljno veliki. Stoga, trebamo da koristimo prethodnu obuku i fino podešavanje.




*Sika 9. Način rada pre-training-a i fine-tuning-a*

Koraci u finom podešavanju su sledeći:

1. Prvo se obuci model neuronske mreže, tj. izvorni model, na izvornom skupu podataka.

2. Kreiranje novog modela neuronske mreže, odnosno ciljnog modela. Ovo kopira sve dizajne modela i njihove parametre na izvorni model osim izlaznog sloja. Pretpostavljamo da ovi parametri modela sadrže znanje naučeno iz izvornog skupa podataka i ovo znanje će takođe biti primenljivo na ciljni skup podataka. Takođe pretpostavljamo da je izlazni sloj izvornog modela usko povezan sa oznakama izvornog skupa podataka, stoga se ne koristi u ciljnom modelu.

3. Dodavanje izlaznog sloja ciljnom modelu, čiji je broj izlaza broj kategorija u ciljnom skupu podataka. Zatim nasumično inicijalizovanje parametara modela ovog sloja.

4. Obučavanje ciljnog modela na ciljnom skupu podataka. Izlazni sloj će biti obučen od nule, dok se parametri svih ostalih slojeva fino podešavaju na osnovu parametara izvornog modela.

Zašto nam je potrebno fino podešavanje?

Zato što modeli koje obučavaju drugi možda nisu pogodni za naše zadatke.
Skupovi podataka mogu pratiti različite distribucije, mreža drugih zadataka može biti previše komplikovana za naše zadatke ili njihova struktura može biti preteška. Na primer, ako želimo da obučimo mrežu da klasifikuje mačke i pse, onda možemo da koristimo mrežu prethodno obučenu na *CIFAR-100*. Međutim, *CIFAR-100* ima 100 klasa, a nama su potrebne samo dve. Prema tome, trebalo bi da popravimo opšte slojeve unapred obučene mreže, a zatim da modifikujemo izlazni sloj da odgovara našem zadatku.
Uopšteno govoreći, fino podešavanje ima sledeće prednosti:

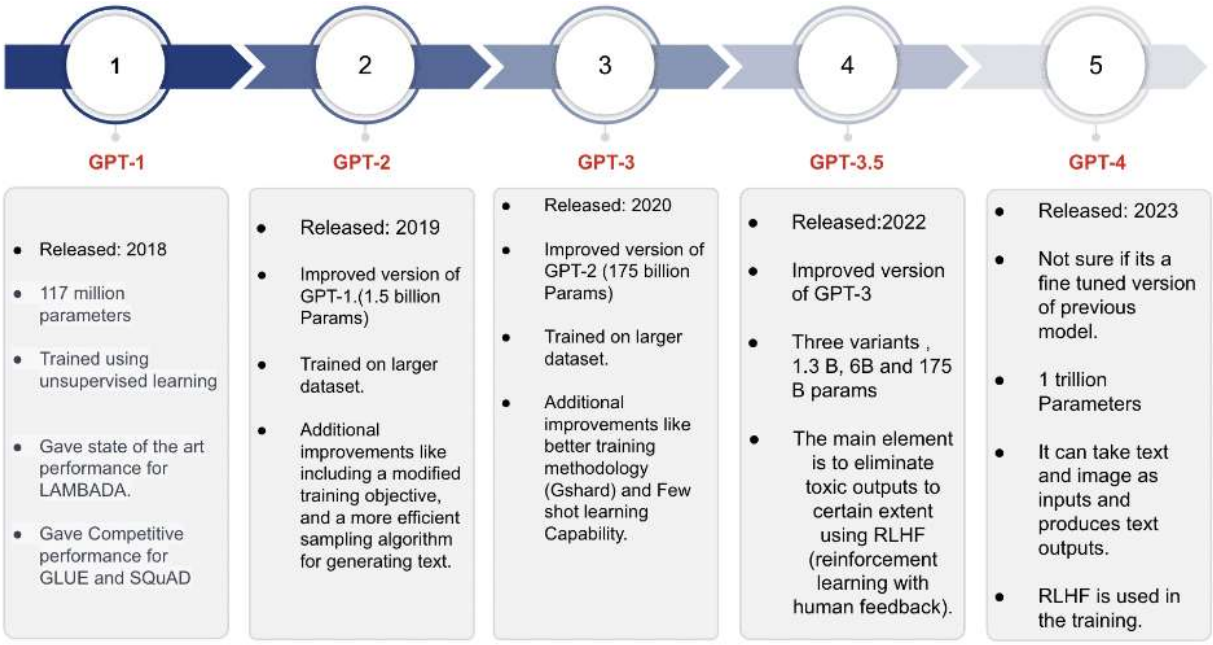
- Ne moramo da treniramo od nule, što nam štedi vreme i troškove.
- Prethodno obučeni modeli se često obučavaju na velikim skupovima podataka, što povećava sposobnost generalizacije našeg modela kada se vrši fino podešavanje nizvodno.
- Fino podešavanje je lako primeniti i potrebno je samo da se fokusiramo na svoje zadatke.

Što se tiče *pre-traininga*, prednosti su sledeće:

- Performanse pre-traininga na velikim skupovima podataka će ograničiti konačne rezultate na *downstream* zadacima, tj. što je bolji prethodno obučeni model, bolji su rezultati.
- Unapred obučeni modeli neće značajno poboljšati performanse na *fine-tuning* zadacima.

- U poređenju sa nasumičnom inicijalizacijom, dobitak od *pre-treninga* i finog podešavanja će postati manji kako *downstream* skup podataka postaje sve veći, tj. prethodna obuka i fino podešavanje donose velika poboljšanja na *downstream* zadacima manjeg obima.

# 3 Evolucija GPT modela




*Slika 10. Evolucija GPT modela* [3]

## 3.1 GPT-1

Ovo je prvi model u seriji GPT modela i treniran je na oko 40GB tekstualnih podataka. Model je postigao rezultate koji su smatrani vrhunskim za zadatke modeliranja kao što su *LAMBADA*, i pokazao je konkurentne performanse za zadatke poput *GLUE* i *SQuAD*. Sa maksimalnom dužinom konteksta od 512 tokena (oko 380 reči), model može zadržati informacije za relativno kratke rečenice ili dokumente po zahtevu. Impresivne sposobnosti modela za generisanje teksta i snažne performanse na standardnim zadacima su bile pokretač za razvoj kasnijih modela u seriji.

## 3.2 GPT-2

Nastao iz GPT-1 modela, GPT-2 model zadržava iste arhitektonske karakteristike. Međutim, prolazi kroz trening na još većem korpusu tekstualnih podataka u poređenju sa GPT-1. Posebno, GPT-2 može da primi duplo veću količinu ulaznih podataka, omogućavajući

obradu obimnijih uzoraka teksta. Sa skoro 1.5 milijardi parametara, GPT-2 pokazuje značajno povećanje kapaciteta i potencijala za modeliranje jezika.

Evo nekih glavnih poboljšanja u GPT-2 u odnosu na GPT-1:

- **Modified Objective Training** je tehnika koja se koristi tokom faze pre-treninga radi poboljšanja jezičkih modela. Tradicionalno, modeli predviđaju sledeću reč u sekvenci samo na osnovu prethodnih reči, što može dovesti do potencijalno nespojivih ili irelevantnih predviđanja. Modifikovani objektivni trening rešava ovaj problem uključivanjem dodatnog konteksta, kao što su delovi govora (imenica, glagol, itd.) i identifikacija subjekta-objekta. Iskorišćavanjem ovih dodatnih informacija, model generiše izlaze koji su koherentniji i informativniji.

- **Normalizacija slojeva** je još jedna tehnika koja se koristi radi poboljšanja treninga i performansi. Uključuje normalizaciju aktivacija svakog sloja unutar neuronske mreže, umesto normalizacije ulaza ili izlaza mreže kao celine. Ova normalizacija ublažava problem internog kovarijacionog pomeranja, što se odnosi na promenu distribucije aktivacija mreže usled promena parametara mreže.

- GPT-2 takođe koristi unapređene algoritme za semplovanje (process izbora sledeće reci u tekstu) u poređenju sa GPT-1. Ključna poboljšanja uključuju:
    - **Top - p sampling** - samo se tokeni čija ukupna verovatnoća prelazi određeni prag uzimaju u obzir tokom semplinga. Ovo izbegava uzorkovanje iz niskoverovatnih tokena, rezultirajući u generisanju teksta koji je raznolikiji i koherentniji.
    - **Temperature scaling** logita (sirove izlazne vrednosti neuronske mreže pre Softmax-a) kontroliše nivo nasumičnosti u generisanom tekstu. Niže temperature daju konzervativniji i predvidljiviji tekst, dok više temperature proizvode kreativniji i neočekivani tekst.
    - **Unconditional sampling** (random sampling), koja korisnicima omogućava istraživanje generativnih sposobnosti modela i može proizvesti izvrsne rezultate.

GPT-2 je transformers model koji je unapred obučen na veoma velikom korpusu engleskog jezika po samo-nadgledanom (eng. self-supervised) principu. To znači da je bio obučen samo na sirovim tekstovima, bez ikakvog ljudskog obeležavanja (zbog čega može koristiti velike količine javno dostupnih podataka), sa automatskim procesom generisanja ulaza i oznaka iz tih tekstova. Preciznije, obučen je da pogodi sledeću reč u rečenicama.
Još preciznije, ulazi su sekvence kontinuiranog teksta određene dužine, a ciljevi su ista sekvenca, pomerena za jedan token (reč ili deo reči) udesno. Model interno koristi mehanizam maske da bi se osigurao da predikcije za token i koriste samo ulaze od 1 do i, ali ne i buduće tokene.
Na ovaj način, model uči unutrašnju reprezentaciju engleskog jezika koja se potom može koristiti za izdvajanje korisnih karakteristika za nizvodne zadatke. Model je, međutim, najbolji u onome za šta je unapred obučen, a to je generisanje tekstova iz podsticaja.
Ovo je najmanja verzija GPT-2, sa 124 miliona parametara.

| Parameters | Layers | $d_{model}$ |
|---|---|---|
| 117M | 12 | 768 |
| 345M | 24 | 1024 |
| 762M | 36 | 1280 |
| 1542M | 48 | 1600 |

*Slika 11. Arhitekture GPT-2 modela (podaci sa istraživanja)* [4]

Povezani modeli: GPT-Veliki, GPT-Srednji i GPT-XL, koji imaju 774 mil., 355 mil. i 1.5 milijardi parametara, respektivno (podaci sa *Hugging face* sajta).

## 3.3 GPT-3

GPT-3 model je evolucija GPT-2 modela, prevazilazi ga u nekoliko aspekata. Treniran je na značajno većem korpusu tekstualnih podataka i ima maksimalno 175 milijardi parametara. Može generisati visokokvalitetan tekst koji je ne samo gramatički ispravan i semantički koherentan, već takođe pokazuje veći stepen kreativnosti i originalnosti. Takođe može generisati širi spektar vrsta teksta, uključujući poeziju, viceve i opise proizvoda. Pored toga, GPT3 je sposoban da obavi zadatke poput prevođenja, sumiranja i odgovaranja na pitanja

Uz povećanje veličine, GPT-3 je uveo nekoliko značajnih poboljšanja:

- **GShard** (*Giant-Sharded* model parallelism) - omogućava da se model deli na više akceleratora. Ovo olakšava paralelno treniranje i zaključivanje, posebno za velike jezičke modele sa milijardama parametara.

- **Mogućnost učenja bez uzorka** (*zero-shot learning*) omogućava GPT-3 da izvodi zadatke za koje eksplicitno nije bio treniran. To znači da može generisati tekst kao odgovor na nove upite, koristeći svoje opšte razumevanje jezika i dati zadatak.

- **Mogućnost učenja sa malim brojem primera** (*few-shot learning*) omogućava ovom modelu da se brzo prilagodi novim zadacima i domenima sa minimalnim treniranjem. Pokazuje impresivnu sposobnost da uči na osnovu malog broja primera.

- **Višejezička podrška** vest je u generisanju teksta na oko 30 jezika, uključujući engleski, kineski, francuski, nemački i arapski. Ova široka višejezička podrška čini ga veoma svestranim jezičkim modelom za razne primene.

- **Unapređeno semplovanje** koristi poboljšani algoritam uzorkovanja koji omogućava podešavanje nasumičnosti u generisanom tekstu, slično kao i GPT-2. Dodatno, uvodi opciju "*prompted*" *sampling*-a, koja omogućava generisanje teksta na osnovu korisnički definisanih promptova ili konteksta.

## 3.4 GPT-3.5

Slično kao i njegovi prethodnici, modeli GPT-3.5 serije izvedeni su iz GPT-3 modela. Međutim, ključna karakteristika GPT-3.5 modela leži u tome što se pridržavaju specifičnih politika zasnovanih na ljudskim vrednostima, koje su inkorporirane pomoću tehnike nazvane Reinforcement Learning with Human Feedback (RLHF) – Pojačano učenje uz povratne informacije od ljudi. Primarni cilj bio je uskladiti modele sa namerama korisnika, smanjiti toksičnost i dati prioritet istinitosti u generisanom izlazu. Ova evolucija označava svesni napor da se unapredi etička i odgovorna upotreba jezičkih modela radi pružanja sigurnijeg i pouzdanijeg korisničkog iskustva. Sve ovo je nastalo zbog toga sto je nakon objave GPT-3 modela došlo do masovnog globalnog straha i velikog nezadovoljstva među svetskom populacijom.

Poboljšanja u odnosu na GPT-3:

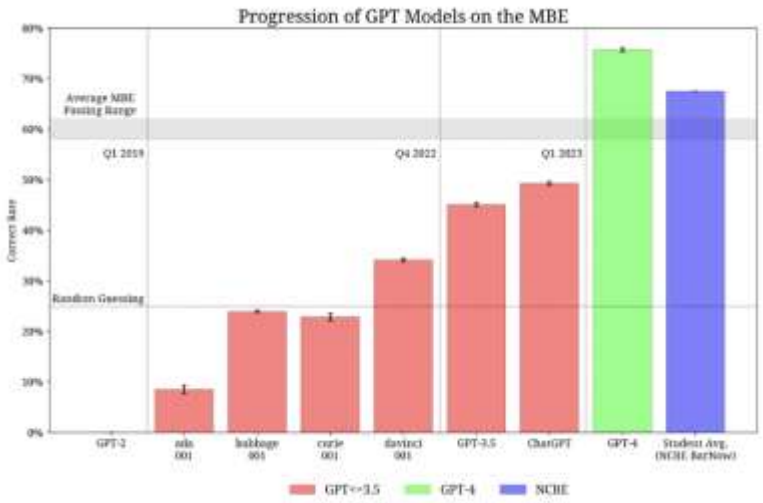
OpenAI je koristio Reinforcement Learning uz povratne informacije od ljudi da bi fino podešavao GPT-3 i omogućio mu da prati širok skup instrukcija. Tehnika RLHF podrazumeva obuku modela primenom principa pojačanog učenja, pri čemu model dobija nagrade ili kazne na osnovu kvaliteta i usklađenosti generisanih izlaza sa ljudskim mišljenjem. Integracijom ovih povratnih informacija u proces obuke, model stiče sposobnost da uči iz grešaka i unapređuje svoje performanse, što na kraju rezultira tekstualnim izlazima koji su prirodniji i etički ispravniji.

## 3.5 GPT-4

GPT-4 predstavlja najnoviji model u GPT seriji koji uvodi multimodalne sposobnosti koje mu omogućavaju obradu tekstualnih i slikovnih ulaza prilikom generisanja tekstualnih izlaza. On podržava različite formate slika, uključujući dokumente sa tekstom, fotografije, dijagrame, grafikone, šeme i snimke ekrana.

Iako *OpenAI* nije otkrio tehničke detalje kao što su veličina modela, arhitektura, metodologija treniranja ili težine modela za GPT-4, neke procene ukazuju da model ima skoro jednu milijardu parametara. Osnovni model prati cilj treniranja sličan prethodnim modelima, sa ciljem predviđanja sledeće reči na osnovu niza reči. Proces treniranja uključivao je upotrebu ogromnog korpusa javno dostupnih podataka sa interneta i licenciranih podataka.

GPT-4 je pokazao superiorne performanse u poređenju sa GPT-3.5 u internim evaluacijama faktičnosti izazovnih podataka u *OpenAI*-u, kao i u javnim testiranjima poput *TruthfulQA*. Tehnike *RLHF* koje su korišćene u GPT-3.5 takođe su ugrađene u GPT-4. *OpenAI* aktivno nastoji da unapredi GPT-4 na osnovu povratnih informacija dobijenih od *ChatGPT* i drugih izvora.




*Slika 12. Napredak GPT modela na MBE testiranju*

Noviji GPT modeli (3.5 i 4) su testirani na zadacima koji zahtevaju zaključivanje i stručno znanje. Modeli su testirani na brojnim ispitima koji su poznati po svojoj izazovnosti. Jedan takav ispit na kojem su upoređivani GPT-3 (ada, babbage, curie, davinci), GPT-3.5, *ChatGPT* i GPT-4 je MBE ispit. Sa grafa možemo videti kontinuirano poboljšanje rezultata, pri čemu je GPT-4 čak nadmašio prosečan rezultat studenata.

## 3.6 Prilagođavanje GPT modela za analizu sentimenata

Analiza sentimenata je proces automatskog određivanja i razumevanja sentimenta ili emocionalnog tonaliteta izraza ili teksta. Ova analiza se često primenjuje na tekstualne podatke kako bi se odredilo da li je sentiment pozitivan, negativan ili neutralan. Cilj analize sentimenata je da se kvantifikuje ili kategorizuje emocionalni stav ili reakcija izražena u tekstu, kao što su mišljenja, ocene, komentari ili povratne informacije korisnika.

Kako znamo da se radi o analizi sentimenata, odabir modela za fino podešavanje treba da bude u skladu sa tim, obzirom da ovaj zadatak u svojoj osnovi ima klasifikaciju sentimenata, treba odabrati model koji je pogodan za takvu klasifikaciju, obzirom da na internetu ima dosta već fino podešenih modela, dobar odabir bi podrazumevao neki model koji je već podesen, a zatim nadograditi taj model dodatnim podešavanjem specifičnim za podatke koji su u pitanju.

Ima puno dostupnih modela koji su modifikacija na osnovni model, tačnije pretrenirani i fino podešeni na osnovu GPT modela za određene namene, sajt koji pruža informacije o ovakvim modelima i istovremeno pruža potpuno besplatno korišćenje svih dostupnih modela jeste https://huggingface.co/, ovde se mogu naći različiti modeli, za različite namene, podešeni za razne zadatke. Međutim, što se tiče GPT modela, on je na ovom sajtu dostupan samo do serije 2, svi noviji modeli se isključivo mogu naći na sajtu https://openai.com/, gde se za svaki token koji se koristi, ovom sajtu placa određena provizija, trenutno postoji četiri dostupna modela, iz GPT porodice na ovom sajtu koji se

mogu koristiti i specijalizovani su za fino podešavanje i to su: *ada*, *babbage*, *curie* I *davinci*, gradacijski poređani po ceni koja se plaća za svaki od njih, a cenovnik izgleda ovako:

| Model | Training | Usage |
|---|---|---|
| Ada | $0.0004 / 1K tokens | $0.0016 / 1K tokens |
| Babbage | $0.0006 / 1K tokens | $0.0024 / 1K tokens |
| Curie | $0.0030 / 1K tokens | $0.0120 / 1K tokens |
| Davinci | $0.0300 / 1K tokens | $0.1200 / 1K tokens |

*Slika 13. Cenovnik GPT modela specijalizovanih za fino podešavanje*

Sa slike se može videti da su cene formirane u odnosu na broj tokena koji se koristi, uzmimo u obzir, da ako koristimo tokenizaciju na nivou reci, sto je slučaj kod ovih modela, i da prosecan skup podataka za analizu sentimenata ima 3000 instanci, za taj broj instanci prosecan broj tokena je 100000, zaključujemo da bi cena za jedno fino podešavanje za prosecan skup podataka za analizu sentimenata kada je u pitanju preneseno učenje, kosta 0.16, 0.24, 1.20, 12.0 dolara, respektivno za gorepomenute modele. Sto je za potrebe istrazivanja preskupo, uzevsi u obzir da bi se neki modeli trebali iskoristiti vise puta, zbog samog ispitivanja efikasnosti hiperparametara.

Kako bi nad nekim skupom podataka mogla da se izvrsi analiza sentimenata, prvo treba tekstualne podatke urediti i dovesti ih u pogodan format za obradu. Zatim treba izvrsiti tokenizaciju teksta u skladu sa tokenizacijom modela koji se koristi za preneseno ucenje. Svaki model za preneseno ucenje ima svoj pretrenirani tokenizer i tekstualni podaci koji se koriste za fino podesavanje ovih modela za specifican zadatak se pre koriscenja prvo tokenizuju.

Ako bismo uzeli u obzir rad svih modela za klasifikaciju, na izlazima ovih modela, bez dodavanja dodatnih slojeva, dobijali bi se rezultati u varijansama, kako bi se ovi podaci preveli u razumljiv oblik, trebalo bi ih za početak prevesti uz pomoć *softmax* sloja, koji će na osnovu izlaza izračunati pripadnost klasi svake instance na izlazu, ovaj sloj se može podesiti u odnosu na broj klasa koje se nalaze u podacima, tako da će izlazne vrednosti poslednjeg skrivenog sloja neuronske mreže, nakon prolaska kroz ovaj sloj, biti jasne čoveku.

Međutim, softmax sloj, je svakako izlazni sloj normalne neuronske mreže, šta zapravo treba da se desi između poslednjeg skrivenog sloja pretreniranog modela i softmax sloja, kako bi se određeni model prilagodio za specifični zadatak i na posebne podatke?

Upravo to predstavlja fino podešavanje modela. Za početak, dodaju se određeni potpuno povezani slojevi *Dense layers,* na poslednji izlazni sloj, obzirom na to da ne znamo (zna se iz seme, ali se govori o uopštenom slučaju rešavanja problema) koliko neurona ima taj sloj, moramo prilagoditi izlaz tog sloja, na ulaz našeg, dodatog sloja, to se može uraditi na vise načina, jedan od najpopularnijih svakako jeste korišćenje mean funkcije za redukciju dimenzonalnosti izlaznog tenzora na oblik pogodan za *batch_size* dodatog *Dense* sloja, drugim recima, ovaj deo u svakom finom podešavanju predstavlja sponu između pretreniranog modela koji se koristi i dela koda koji fino podešava taj model za određeni

zadatak nad njegovim podacima. Batch_size se po pravilu određuje na osnovu količine podataka koja se obrađuje (~0.2 x instance_number), tj. na broj instanci, generalno, za velike količine podataka se koristi veći batch_size, obzirom na to da su skupovi podataka za preneseno učenje siromašni podacima, jasno je da će se za ovakve zadatke koristiti manji broj za batch_size. Naravno, svaki sloj sadrži aktivacionu funkciju.
Treba pomenuti i pojam *attention_mask* to je niz nula i jedinica koji se dodeljuje svakom modelu kako bi znao na koje podatke da obrati pažnju, a na koje ne. Ovaj vektor se uvodi uglavnom kada imamo *padding* (dopunjavanje sekvenci do određene dužine, kako bi sve sekvence bile iste dužine) podešen za podatke i služi da odvoji bitne tokene od nebitnih i na taj način ukaže modelu na tokene od interesa. Broj slojeva za fino podešavanje ne može biti fiksan i zahteva se istraživanje kako bi se odredio tačan broj, preporuka je početi od dva dodatna sloja sa *softmax* slojem dodatim na kraju. Nakon dodavanja slojeva, neuronska mreža se trenira klasično bez nekih razlika u odnosu na tipične neuronske mreže. Takodje, može se odabrati da li će tokom treninga moći da se trenira inicijalni model, postoji opcija i za zamrzavanje težina svih slojeva početnog modela, kako se iste ne bi menjale tokom procesa treninga. Inicijalna podešavanja jesu da je modelu omogućeno treniranje i da se sve težine neurona zamrzavaju.

# 4 Praktični deo projekta

Ideja koja stoji iza praktičnog dela jeste fino podešavanje GPT modela kako bi se realizovalo preneseno učenje između već istreniranog modela i skupa podataka koji nema dovoljan broj instanci za samostalno formiranje kvalitetnog modela dubokog učenja. Fokus će biti na analizi sentimenata tvitova korisnika, tačnije klasifikaciji sentimenata korisnika, sa ciljem prepoznavanja negativnih i pozitivnih komentara.
Biblioteke koje su korišćene u projektu su: *TensorFlow*, *matplotlib*, *pandas*, *seaborn*, *sklearn*… Korišćena je verzija python-a 3.11.1.

Skup podataka koji je korišćen se zasniva na tvitovima u kojima je izražavano mišljenje korisnika ove društvene mreže. Ima dve kolone od kojih jedna predstavlja lematizovane tvitove, dok druga predstavlja sentiment koji je automatski izgenerisan od strane VADER_sentiment algoritma za analizu sentimenta koji se koristi za određivanje sentimenta (pozitivnog, negativnog ili neutralnog) u tekstualnim podacima.
Algoritam VADER_sentiment se oslanja na rečnik sa unapred definisanim sentimentnim vrednostima za veliki broj reči, kao i na heuristike za analizu sentimenta. Koristi se za analizu sentimenta u kratkim tekstualnim porukama, kao što su tvitovi ili komentari na društvenim mrežama.

## 4.1 Priprema podataka i modela za fino podešavanje

Preprocesiranje podataka podrazumeva pripremu podataka za prilagođenje modela. Obzirom na to da je ideja da se izvrši klasifikacija komentara na pozitivne ili negativne, izdvojene su instance sa strogo pozitivnim i negativnim kontekstima. Skup podataka je izbalansiran preventivno, iako kod neuronskih mreža balansiranje ne igra značajnu ulogu, u nastavku će biti prikazani rezultati sa balansiranim i nebalansiranim skupom podataka, radi prikaza uticaja balansiranja na podatke. Podaci su podeljeni, po propozicijama dubokog učenja, na *train*, *test* i kasnije u projektu value podskupove.

```
X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y)
```

Treba napomenuti i da je korišćen parametar *stratify train_test_split* funkcije, kako bi se izbalansirala podeljenost pozitivnih i negativnih sentimenata u setovima podataka.

Kao sto je u teorijskom delu naglašeno, podaci pre treniranja, moraju da se prilagode obliku koji je pogodan za određeni model.

```
tokenizer = GPT2Tokenizer.from_pretrained("gpt2",
    pad_token=PAD_TOKEN,
    eos_token=EOS_TOKEN,
    max_length=MAX_LENGTH,
    is_split_into_words=True)
```

Obzirom na to da je GPT transformers model, za rad sa ovim modelom je bilo potrebno preuzeti i instalirati *transformers* biblioteku koja omogućava rad sa GPT2 tokenima. Kao sto se može videti iz navedenog, preuzimanje tokenizera se vrši veoma lako, samo se navede ime modela sa *HuggingFace* sajta, dodatni parametri koji su dodati u tokenizer jesu *pad_token*, koji sluzi za dodavanje praznih stringova kako bi se izjednačile dužine sekvenci, *eos_token* koji označava kraj sekvence, *max_length* parametar koji označava maksimalnu dužinu sekvence, gde se *padding* token koristi za dopunjavanje do maksimuma. I *split_into_words* parametar koji je postavljen na True, sto znaci da tokenizer očekuje sekvence reči kao ulaze i njegov zadatak je da svaku tokenizuje posebno.

```
X_train_ = [tokenizer(str(x), return_tensors='tf',
                      max_length=MAX_LENGTH,
                      truncation=True, pad_to_max_length=True,
                      add_special_tokens=True)['input_ids']
                      for x in X_train]
```

Tokenizacija setova podataka, parametri su sledeći, *return_tensors='tf'* parametar koji govori tokenizeru da vrati izlaz kao *TensorFlow* tenzor, jer nam je plan da radimo sa modelom koji zahteva ovakav tip izlaza, *truncation* govori tokenizeru da odseče sekvence ako su duže od maksimalne dužine, *pad_to_max_length* dopunjava kraće sekvence do maksimalne dužine, *add_special_tokens* dodaje gorenavedene specijalne tokene u podatke. Na kraju, ['input_ids'] izdvaja ID-jeve unosa iz izlaza tokenizera, što je ono što trebamo

proslediti modelu. Tokenizer može da vrati i druge informacije, poput *attention* maski, ali u ovom slučaju, izdvajamo samo ID-jeve unosa.

```
X_train_in = tf.squeeze(tf.convert_to_tensor(X_train_), axis=1)
```

Ova linija koda prvo konvertuje X_train_ u tenzor, a zatim koristi funkciju squeeze da ukloni sve dimenzije veličine 1 na poziciji 1. Rezultat je tenzor koji ima jednu manju dimenziju.Na primer, ako je oblik X_train_ bio (n, 1, m), gde su n i m bilo koji brojevi, posle primene tf.squeeze oblik će postati (n, m). Ova operacija je korisna za uklanjanje nepotrebnih dimenzija i čini dalju obradu podataka efikasnijom.

```
X_train_mask_ = [tokenizer(str(x), return_tensors='tf',
                    max_length=MAX_LENGTH,
                    truncation=True, pad_to_max_length=True,
                    add_special_tokens=True)["attention_mask"]
                    for x in X_train]

X_train_mask = tf.squeeze(tf.convert_to_tensor(X_train_mask_), axis=1)
```

Ove dve linije koda izvlače *attention* masku, naravno, sve ovo je urađeno i za test i validacione podatke, nakon ovih par linija koda imamo sve sto nam je potrebno za korišćenje GPT modela.

```
def map_sentiment(value):
  if value == 'Negative':
    return 0
  if value == 'Positive':
    return 1

y_train_ = y_train.map(map_sentiment)
y_test_ = y_test.map(map_sentiment)
```

Prevođenje izlaznih podataka u pogodan oblik.

Sledeća stavka jeste odabir i učitavanje GPT modela koji trebamo da fino podesimo.

Na sajtu HuggingFace se mogu pronaći različite varijacije GPT modela.


*Slika 14. Varijante GPT modela*

Kao sto se može videti na slici 23., u ovom trenutku postoji 6,084 verzija gpt modela, zbog toga treba dobro proučiti dostupne modele pre odabira pravog modela za određeni zadatak.

U radu je primenjen klasični “gpt2” model, koji je najviša verzija besplatno dostupna na internetu iz te porodice.

Učitavanje modela će biti objašnjeno na primeru *gpt2.*

```
model = TFGPT2Model.from_pretrained("gpt2", use_cache=False,
        pad_token_id=tokenizer.pad_token_id,
        eos_token_id=tokenizer.eos_token_id)
model.training = True
```

TensorFLow omogućava učitavanje pretreniranih modela porodice GPT2 modela, parametri funkcije *from_pretrained* su model, koji je u ovom slučaju gpt2 model prilagođen za srpski jezik, zatim *use_cache* parametar koji olakšava ponovno korišćenje ovog modela čuvajući neke informacije u kešu, ovaj parametar je podešen na False zbog toga sto kasnije u kodu čuvamo model na lokalnoj mašini. *Pad_token* i *eos_token* su preuzeti sa tokenizera, jer moraju da se poklapaju. Nakon sto je model preuzet, čuva se u računaru, radi uštede vremena i interneta.

```
model = TFGPT2Model.from_pretrained("gpt2")
```

Zbog postojanja vise modela, potrebna je linija za specifikaciju učitavanja trenutno traženog modela u kernel.

```
model.training = True

model.resize_token_embeddings(len(tokenizer))

for layer in model.layers:
    layer.trainable = False

model.summary()
```

Priprema modela za fino podešavanje i dalji trening. Omogućavanje treniranja - Ova linija postavlja model u režim treninga. Ovo omogućava da se u modelu aktiviraju specifične funkcionalnosti koje su korisne samo tokom faze treninga, kao što su dropout i batch normalization. Zatim promena dužine - Ova linija menja veličinu sloja za ugrađivanje tokena (token embedding layer) u modelu tako da odgovara broju tokena u tokenizatoru. Ovo je neophodno jer se broj tokena u tokenizatoru može promeniti kada dodajemo nove tokene, a model mora da prati ove promene tako što će prilagoditi veličinu svog sloja za ugrađivanje tokena. Linija u kojoj se “zamrzavaju” slojevi modela - Ova linija prolazi kroz svaki sloj u modelu i postavlja njegov atribut *trainable* na *False*. Ovo znači da ovi slojevi neće biti ažurirani tokom faze treninga, što je korisno kada želimo da "zamrznemo" neke slojeve modela dok treniramo druge slojeve.

`model.summary()` - Ova linija ispisuje sažetak modela. Sažetak uključuje broj slojeva u modelu, oblike izlaza svakog sloja, broj treniranih i netreniranih parametara, itd. Ovo je korisno za brzi pregled strukture i veličine modela.
Ovaj deo koda je tipičan korak u procesu prenesenog učenja, gde preuzimamo prethodno obučeni model (u ovom slučaju GPT2 model), prilagođavamo ga za novi zadatak (menjajući veličinu sloja za ugrađivanje tokena) i zatim treniramo samo deo modela (ostavljajući neke slojeve zamrznute).

```
  input = tf.keras.layers.Input(shape=(None,), dtype='int32')
mask = tf.keras.layers.Input(shape=(None,), dtype='int32')
x = model(input, attention_mask=mask)
```

Ovaj kod pravi novi ulazni sloj za `model.shape=(None,)` označava da sloj može prihvatiti nizove bilo koje dužine, a dtype='int32'označava da se očekuju celobrojni podaci (što je tipično za ID-jeve tokena). Sledeća linija, slično prethodnoj, ali se ovaj sloj koristi za *attention* maske koje pokazuju modelu koje delove unosa treba ignorisati prilikom procesiranja. Poslednja linija dodeljuje modelu id-jeve ulaza i *attention* maske i predstavlja poslednji korak pre finog podešavanja modela, priprema za povezivanje sa konkretnim zadatkom.

## 4.2 Fino podešavanje (eng. *fine-tuning*)

Ovaj deo koda primenjuje seriju transformacija na izlazu modela kako bi se dobio konačan izlaz, a zatim definiše novi model koji uključuje te transformacije. Ovo je ključni deo fine tuning procesa jer dodaje nove slojeve koji se treniraju na specifičnom zadatku, dok se originalni model (u ovom slučaju, GPT-2 model) koristi kao *feature extractor*.

```
#x = x.last_hidden_state[:, -1]
x = tf.reduce_mean(x.last_hidden_state, axis=1)
x = tf.keras.layers.Dense(16, activation='relu')(x)
x = tf.keras.layers.Dropout(0.3)(x)
output = tf.keras.layers.Dense(2, activation='softmax')(x)

clf = tf.keras.Model([input, mask], output)
```

`#x = x.last_hidden_state[:, -1]`: Ova linija je zakomentarisana, ali ako se otkomentariše, ona će selektovati poslednji izlaz svake sekvence iz modela. Ovo je korisno za zadatke gde je važan kontekst cele sekvence, kao što je analiza sentimenata.

`x = tf.reduce_mean(x.last_hidden_state, axis=1)`: Ova linija izračunava prosek svih izlaza u sekvenci, što daje jedan izlaz po sekvenci. Ovo je alternativa prethodnoj liniji: umesto da gledamo samo poslednji izlaz, ovde uzimamo u obzir sve izlaze.

`x = tf.keras.layers.Dense(16, activation='relu')(x)`: Ova linija dodaje potpuno povezan (dense) sloj sa 16 jedinica. Potpuno povezani sloj je sloj neuronske mreže u kojem su sve jedinice povezane sa svim jedinicama prethodnog sloja. Aktivaciona funkcija ovog

sloja je ReLU (Rectified Linear Unit), koja je jedna od najčešće korišćenih aktivacionih funkcija u dubokom učenju.

Dalje su dodati dodatni slojevi za fino podešavanje, tokom istraživanja su isprobane razne kombinacije slojeva, gde je dropout sloj primenjivan nakon svakog sloja, na samom kraju, između, isprobane su različite duzine *batch*-eva, međutim, osnovna postavka, sa dva potpuno povezana sloja je nakon istraživanja dala najbolje rezultate.

`output = tf.keras.layers.Dense(2, activation='softmax')(x)`: Ova linija dodaje finalni potpuno povezani sloj sa 2 jedinice, što odgovara broju klasa u zadatku klasifikacije. Aktivaciona funkcija ovog sloja je *softmax*, koja se često koristi za zadatke klasifikacije jer pretvara izlaze modela u verovatnoće klasa. *Softmax* sloj je zaslužan za prevođenje verovatnoća u vrednosti razumljive čoveku.

`clf = tf.keras.Model([input, mask], output)`: Na kraju, ova linija koda definiše novi model koji uključuje sve prethodne transformacije. Tako da je završeno fino podešavanje modela i model je spreman za treniranje nad novim podacima sa novim zadatkom. Za svaki od modela izvršeno je specifično fino podešavanje kako bi se postigle željene performanse.

## 4.3 Treniranje modela

Podešavanje osnovnih hiperparametara:

```
base_learning_rate = 0.0005
optimizer=tf.keras.optimizers.Adam(learning_rate=base_learning_rate)
#loss=tf.keras.losses.BinaryCrossentropy()
loss=tf.keras.losses.SparseCategoricalCrossentropy()

clf.compile(optimizer=optimizer, loss=loss, metrics=['accuracy'])

callbacks = tf.keras.callbacks.EarlyStopping(
        monitor="val_accuracy", verbose=1, patience=10,
        restore_best_weights=True)
```

Nakon istraživanja ustanovljeno je da je najbolja početna stopa učenja 0.0005. Korišćen je *Adam* optimizer, tipični odabir kod modela dubokog učenja, takodje je pokusan rad sa *SGD* optimizerom, ali ovaj pristup nije dao značajne rezultate. Kada je u pitanju loss funkcija, obzirom na to da je u pitanju klasifikacija i radimo nadgledano učenje, gde se zna koji izlazni podaci treba da se dobiju na izlazu, ova funkcija se čini najpogodnijom, naravno, tu je mogla da se iskoristi binarna *crossentropy*, međutim zbog potreba projekta je ostavljena višeklasna verzija ove funkcije koja nema loš uticaj na binarnu klasifikaciju. Vrši se vraćanje na najbolje težine neurona nakon treninga.

```
history = clf.fit([X_train_in, X_train_mask],
                  y_train_in,
                  epochs=50,
                  batch_size=32,
                  validation_split=0.2,
```

```
                    callbacks=callbacks)
```

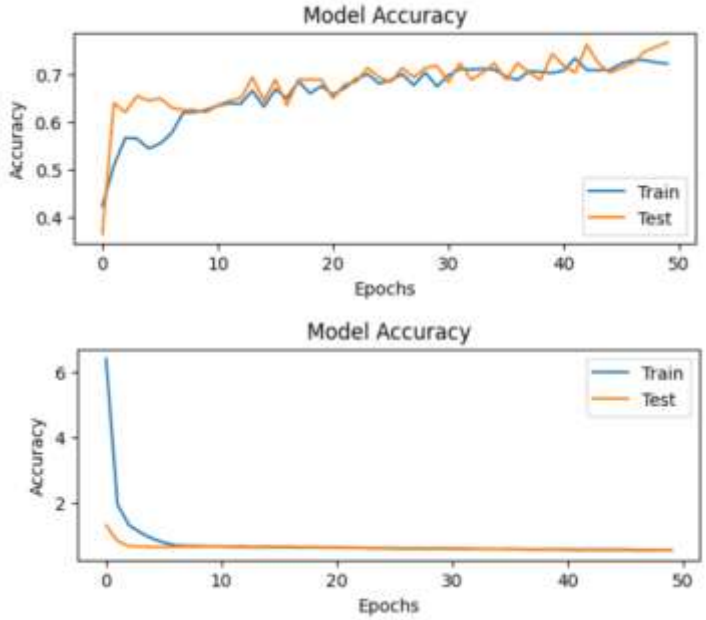
Linija koda u kojoj se model trenira, za broj epoha je postavljeno 50, jer je skup podataka koji je dostupan veoma mali i model se nakon nekog vremena prilagodi podacima za testiranje, dok se uočavaju razlike u testiranju nad validacionim i test skupom podataka. *Batch_size* je inicijalno setovan na 256, međutim, binarnim pomeranjem u desno je ustanovljeno da se najbolji rezultati dobijaju za vrednost *batch_size*=32.

## 4.4 Rezultati treninga i finog podešavanja (evaluacija)

U ovom eksperimentu, primenjen je model dubokog učenja koji je treniran nad skupom podataka za analizu sentimenta. Tokom treninga, model je prolazio kroz 50 epoha, pri čemu su najbolji rezultati postignuti u poslednjoj epohi. Ovo ukazuje na to da je model u trenutku poslednje epohe i dalje napredovao. Shodno ovome, pretpostavka je da bi se potencijalno dobili još bolji rezultati u slučaju kada bi se izvršavao veći broj epoha.

Tačnost (*val_accuracy*) dobijena u poslednjoj epohi od 0.7659 ukazuje na to da je model uspeo da tačno klasifikuje 76,59% tvitova na test skupu. Važan faktor pri evaluaciji modela je i gubitak (val_loss). U ovom slučaju, najniži gubitak od 0.5281 takođe je postignut u poslednjoj epohi. Ovaj rezultat ukazuje na to da je model uspešno minimizovao gubitak tokom treninga.

Grafici odnosa tačnosti (*accuracy*) i gubitka (*loss*) pružaju dodatni uvid u performanse modela. U ovom slučaju, grafici ne pokazuju znakove overfittinga, što je bio jedan od ciljeva. To znači da model nije preterano prilagođen trening skupu i da se dobro generalizuje na test skup. Na sledećoj slici se nalaze pomenuti grafici.




*Slika 15. Grafici za accuracy i loss*

Nakon treninga, model je evaluiran na posebnom test skupu podataka kako bi se proverila njegova sposobnost generalizacije. Dobijeni rezultati pokazuju da je model postigao tačnost od 0.7038 na test skupu.

Ključni izveštaj za evaluaciju je classification report koji pruža detaljnije informacije o performansama modela za svaku klasu. Prema izveštaju, preciznost (precision) i odziv (recall) su najviši za klasu 1 (pozitivni sentiment), dok su nešto niži za klasu 0 (negativni sentiment). F1-mera (f1-score) je mera koja kombinuje preciznost i odziv, a u ovom slučaju iznosi 0.76 za klasu 1 i 0.61 za klasu 0.

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 0.63 | 0.59 | 0.61 | 134 |
| 1 | 0.75 | 0.78 | 0.76 | 207 |
| accuracy | | | 0.70 | 341 |
| macro avg | 0.69 | 0.68 | 0.69 | 341 |
| weighted avg | 0.70 | 0.70 | 0.70 | 341 |

*Slika 16. Izveštaj klasifikacije*

Na kraju imamo matricu konfuzije. Matrica konfuzije pruža vizuelni prikaz tačno i netačno klasifikovanih instanci za svaku klasu. Na osnovu matrice konfuzije, možemo videti da je model uspeo da tačno klasifikuje veći broj instanci za klasu 1 u odnosu na klasu 0.


*Slika 17. Matrica konfuzije*

Nad skupom podataka za analizu sentimenta tvitova postignuta je delimično zadovoljavajuća tačnost i performanse. Iako tačnost od 0.7038 može biti poboljšana, posebno za klasu 0, F1-mera od 0.76 za klasu 1 ukazuje na dobro balansiranje preciznosti i odziva.

Važno je napomenuti da su ograničeni resursi uticali na odluku da se ne zadrže modeli koji su pokazivali preterano prilagođavanje (overfitting) ranije u eksperimentu. Ovo ukazuje na potrebu za dodatnim resursima kako bi se bolje istražili i optimizovali modeli koji bi mogli postići još bolje rezultate.

Dalje unapređenje ovog eksperimenta može uključivati veći skup podataka, dodatne slojeve ili neuronske mreže kako bi se model proširio, kao i primenu tehnika za balansiranje klasa radi bolje obrade podataka s nejednakom raspodelom sentimenta.

Uzimajući u obzir sve ove faktore, ova evaluacija pruža dobar početni uvid u performanse modela dubokog učenja za analizu sentimenta tvitova. Dalje optimizacije i prilagođavanja mogu dovesti do još boljih rezultata u budućnosti.

# 5 Zaključak

Preneseno učenje je veoma širok pojam, njegova primena se prostire u svim oblastima mašinskog/dubokog učenja i polako preuzima primat u treniranju novih modela, pogotovo u oblasti NLP-a, rešava mnoge probleme, od nedovoljnih hardverskih resursa, preko nedostatka trening podataka za neki zadatak, do olakšavanja personalizacije i specijalizacije aplikacija.

Osim doprinosa kada su u pitanju oblasti mašinskog učenja, preneseno učenje štedi vreme inženjerima i omogućava novim i ambicioznim ljudima da s lakoćom istreniraju neki model sa fokusom na specifičan zadatak bez opsežnog poznavanja neuronskih mreža.

Procesiranje prirodnih jezika je jedan od najvećih, ako ne i najveći “profiter”, pored računarskog vida, kada je u pitanju preneseno učenje, iz prostog razloga što se većina modela u NLP-u zasniva na jezičkim pravilima koja se mogu vrlo lako primenjivati na različite zadatke, ogromna je ušteda u vremenu treniranja, pripremi podataka, prilagođavanju novih zadataka na “temeljima” starih i već naučenih. Pa je shodno tome i najzanimljivija oblast za istraživanje kada je u pitanju preneseno učenje.

Transformers arhitektura, kao glavni pokretač cele priče o prenesenom učenju, svojim pristupom, omogućila je GPT modelima da stignu u sam vrh veštačke inteligencije i da u današnjici, običnom čoveku, budu prva asocijacija kada se pomene pojam veštačke inteligencije.

Nažalost, GPT-3+ verzije modela nisu besplatne za korišćenje, možda se u budućnosti to promeni, zbog takvog trenutnog stanja, modeli stariji od GPT-2 modela nisu praktično istraženi u ovom radu, ali je arhitektura koja se krije iza dostupnih modela dovoljan

pokazatelj sposobnosti, moći i efikasnosti GPT modela, koji su lako prilagodljivi za razne NLP probleme, od kojih je jedan analiza sentimenata i u praktičnom radu je prikazano upravo fino podešavanje GPT modela za ovaj problem.

Fino podešavanje, pretreniranje modela, kao osnovni pojmovi u prenesenom učenju su bliži inženjeru od transformers arhitekture. Predstavljaju ponovno treniranje već istreniranih modela i izmenu arhitekture neuronskih mreža kako bi se određeni model prilagodio novom zadatku i nakon toga predstavljao model za specifičnu namenu kada je u pitanju odabrani problem.

Praktični deo rada je namenjen analizi sentimenata tvitova, gde je uspesno podešen model za konkretan zadatak. U praktičnom delu rada je dokazano par tvrdnji koje su se provukle kroz teorijski deo, a to je da sa malom količinom podataka možemo dobiti sasvim korektan model za određeni zadatak, zatim smanjenje vremena izvršavanja, lako fino podešavanje, lako ispitivanje efikasnosti raznih modela, modeli čija izgradnja od nule bi zahtevala dane i nedelje treninga, su kreirani kroz nekoliko sati, takođe je dokazano i to da treniranje ovakve vrste modela ne treba da se fokusira na broj epoha nego na efikasnost modela iz razloga pretreniranja modela zbog male količine trening podataka.

Priča o prenesenom učenju je dobila svoj pun smisao nakon praktičnog dela rada, pokazana je perspektiva, efikasnost, kvalitet i ideja. Preneseno učenje je danas na uzlaznoj putanji, a u bliskoj budućnosti će biti osnova i početak mnogih ideja. S obzirom da je oblast veštačke inteligencije svakog dana sve popularnija i da svakodnevno napreduje, preneseno učenje je nešto što svaki inženjer iz ove oblasti mora imati na umu kako bi pratio korak vremena.

# 6 Literatura

[1] Y. Z. W. D. S. J. P. Qiang Yang, Transfer Learning, Peking: Springer, 2022.

[2] Y. C. Jindong Wang, Introduction to Transfer Learning - Algorithms and practice, Peking: Springer, 2022.

[3] A. Mehra, *A Deep Dive into GPT Models: Evolution & Performance Comparison,* p. 8, 2023.

[4] J. W. R. C. D. L. D. A. I. S. Alec Radford, „Language Models are Unsupervised Multitask Learners,“ 2020.