SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 453

MOBILNA APLIKACIJA ZA ANALIZU OSOBNE POTROŠNJE NA TEMELJU SLIKA MALOPRODAJNIH RAČUNA

Petar Miličević

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 453

MOBILNA APLIKACIJA ZA ANALIZU OSOBNE POTROŠNJE NA TEMELJU SLIKA MALOPRODAJNIH RAČUNA

Petar Miličević

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

DIPLOMSKI ZADATAK br. 453

Pristupnik:	**Petar Miličević (0036515023)**
Studij:	Računarstvo
Profil:	Programsko inženjerstvo i informacijski sustavi
Mentor:	izv. prof. dr. sc. Alan Jović

Zadatak: **Mobilna aplikacija za analizu osobne potrošnje na temelju slika maloprodajnih računa**

Opis zadatka:

Analiza osobne potrošnje putem praćenja maloprodajnih računa u praksi se svodi na praćenje samo skupih stavki ili ukupnih iznosa računa. Za automatizaciju praćenja potrošnje putem maloprodajnih računa nije dostupno mnogo besplatnih rješenja. U ovom diplomskom radu potrebno je razviti mobilnu aplikaciju koja će poslužiti za analizu podataka osobne potrošnje s maloprodajnih računa. Aplikacija treba omogućiti slikanje maloprodajnog računa, lokalizaciju dijela računa sa stavkama, optičko prepoznavanje znakova radi utvrđivanja teksta pojedinih stavki, pohranu odrađenog računa u bazu podataka, korisničko sučelje za pregled ranije uslikanih računa te analizu podataka s računa. U analizi podataka potrebno je omogućiti pregled potrošnje u zadnjih N dana po kategorijama (i po mogućnosti podkategorijama) stavki, datumima i iznosima. Za kategoriziranje skeniranih računa potrebno je prevesti račun koristeći neki od modela za prevođenje te naučiti i implementirati odgovarajući model strojnog učenja koji će na odgovarajućem skupu podataka naučiti kategoriju (i po mogućnosti podkategoriju) stavke. Za prevođenje stavki i učenje kategorija treba razmotriti postojeće, prednaučene modela strojnog učenja te ih po potrebi fino podesiti za primjenu na stavkama maloprodajnih računa slikanih u Hrvatskoj.

Rok za predaju rada: 28. lipnja 2024.

Sadržaj

1. Uvod

Zbog sve većih troškova života praćenje osobnih financija sve se češće spominje kao bitan faktor u postizanju financijskog blagostanja pojedinca. To je naročito bitno za mlađu populaciju koja tek stupa u samostalan život i postepeno stječe navike upravljanja i raspolaganja osobnim financijama. Tradicionalno, analiza osobne potrošnje svodi se na ručno praćenje skupih stavki ili ukupnih iznosa na maloprodajnim računima, no takav pristup je često nepraktičan, zahtijeva puno vremena i sklon je greškama. Razvojem tehnologije pojavila su se rješenja koja omogućuju automatizirano praćenje potrošnje putem mobilnih aplikacija. Nažalost, većina dostupnih rješenja nisu besplatna i rijetko su prilagođena korisnicima u Hrvatskoj.

Cilj ovog diplomskog rada je razviti mobilnu aplikaciju koja će omogućiti jednostavno praćenje osobne potrošnje putem slikanja maloprodajnih računa. Aplikacija će koristiti napredne tehnike obrade prirodnog jezika (engl. *natural language processing - NLP*) i strojno učenje za lokalizaciju dijelova računa sa stavkama, optičko prepoznavanje znakova (engl. *optical character recognition - OCR*) za utvrđivanje teksta, te kategorizaciju stavki u svrhu analize potrošnje. Na taj način, korisnici će moći automatski pratiti svoje troškove i dobiti uvid u strukturu potrošnje bez potrebe za ručnim unosom podataka. Razvijena aplikacija omogućit će korisnicima slikanje maloprodajnih računa te automatsko prepoznavanje i klasifikaciju kupljenih stavki te kasniji pregled potrošnje po kategorijama, datumima i iznosima u odabranom periodu.

Implementacija ovakvog sustava zahtijeva nekoliko ključnih koraka, uključujući razvoj korisničkog sučelja, izgradnju poslužiteljske strane, integraciju tehnologija OCR i NLP te dizajn baze podataka za pohranu i analizu podataka. Kroz ovaj rad, detaljno će biti obrađen svaki od ovih koraka, pružajući uvid u tehnologije i alate koji su korišteni u njegovoj implemetaciji.

Neke od aplikacija sa sličnim funkcionalnostima su "Expensify", "Receipt Bank" i "Smart Receipts". "Expensify" je popularna aplikacija koja omogućuje korisnicima skeniranje računa i automatsko izdvajanje podataka, no zahtijeva pretplatu za korištenje naprednih značajki. "Receipt Bank" se fokusira na digitalizaciju poslovnih računa i integraciju s računovodstvenim programima, što je više usmjereno prema poslovnim nego individualnim korisnicima. "Smart Receipts" nudi slične osnovne funkcionalnosti, ali je manje prilagođen potrebama korisnika u Hrvatskoj (prepoznavanje stavki računa izdanih na hrvatskom jeziku).

2. Obrada prirodnog jezika

Obrada prirodnog jezika (NLP) je interdisciplinarno područje računarske znanosti, umjetne inteligencije i lingvistike koje se bavi interakcijom između računala i ljudskog jezika. NLP omogućuje računalima da razumiju, interpretiraju i generiraju ljudski jezik na način koristan čovjeku. Umjesto razvijanja striktno teorijskih okvira, NLP je fokusiran na izgradnju tehnologija za rješavanje širokog spektra korisnih zadataka, stoga ova tehnologija ima široku primjenu, od prepoznavanja govora i automatskog prevođenja do analize sentimenta i chatbotova [1]. Posljednje su godine donijele revoluciju u sposobnosti računala da razumiju ljudske i programske jezike, pa čak i biološke i kemijske sekvence, poput DNK i proteinske strukture, koje nalikuju jeziku. Najnoviji modeli umjetne inteligencije nalaze svoje mjesto u ovim područjima kako bi analizirali značenje i kontekst unesenog teksta i generirali smislene rezultate. Tako NLP postepeno postaje sastavni dio suvremenog života i nalazi svoje primjene u širokom spektru ljudske djelatnosti poput maloprodaje (chatbotovi korisničke službe) i medicine (tumačenje ili sažimanje elektroničkih zdravstvenih kartona). Agenti za razgovor kao što su Amazonova Alexa i Appleova Siri koriste NLP za slušanje korisničkih upita i pronalaženje odgovora. Viša razina takvih agenata — kao što je GPT-4, koji je nedavno otvoren za komercijalnu upotrebu — može generirati sofisticiranu prozu o širokom spektru tema, kao i moćne chatbotove koji su sposobni voditi koherentne razgovore. Google također koristi NLP kako bi poboljšao rezultate svoje tražilice, a društvene mreže poput Facebooka koriste ga za otkrivanje i filtriranje govora mržnje [2].

U nastavku su navedeni neki od najčešćih zadataka u obradi prirodnog jezika:

- **Klasifikacija cijelih rečenica**
    - **Analiza sentimenta (engl. *Sentiment analysis*)** – određivanje emocionalnog tona rečenice ili dokumenta. Sentiment može biti pozitivan, negativan ili neutralan
    - **Detekcija neželjene e-pošte (engl. *Spam detection*)** – odrediti je li e-mail poruka neželjena (engl. *spam*) ili legitimna (engl. *non-spam*)
    - **Određivanje gramatičke ispravnosti** – procjena ispravnosti rečenice u skladu s gramatičkim pravilima određenog jezika
    - **Određivanje logičke povezanosti rečenica** – utvrđivanje postojanja logičke povezanost između dviju rečenica, tj. smislenog nadovezivanja jedne na drugu
- **Klasifikacija svake riječi u rečenici**
    - **Identificiranje gramatičkih komponenti rečenice** – podrazumijeva određivanje gramatičke uloge svake riječi u rečenici, kao što su imenica, glagol, pridjev i sl.
    - **Identificiranje imenovanih entiteta** – cilj je prepoznati i klasificirati specifične entitete unutar teksta, kao što su osobe, lokacije, organizacije itd.
- **Generiranje teksta**
    - **Dovršavanje zadanog teksta** – uključuje nastavak teksta na temelju zadanog početka. Model predviđa i generira nastavak koji je smisaono povezan s početnim tekstom
    - **Popunjavanje praznina u tekstu koji sadrži maskirane riječi** – podrazumijeva popunjavanje praznih mjesta u tekstu gdje su riječi izostavljene ili maskirane. Model predviđa najprikladnije riječi koje nedostaju
- **Ekstrakcija odgovora iz teksta** – uključuje davanje odgovora na postavljeno pitanje koristeći informacije dostupne u zadanom kontekstu. Model analizira teks-

tualni kontekst i izdvaja relevantan dio koji sadrži odgovor

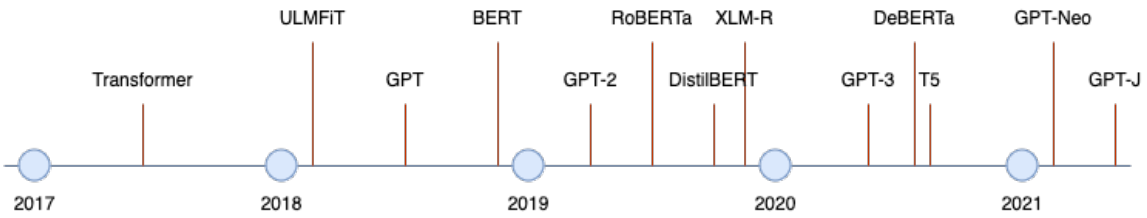
- **Generiranje nove rečenice iz ulaznog teksta**
  - **Prevođenje teksta na drugi jezik** – prevođenje teksta s jednog jezika na drugi jezik. Model analizira ulazni tekst na jednom jeziku i generira ekvivalentan tekst na drugom jeziku
  - **Sumiranje teksta** – sažimanje informacija iz ulaznog teksta u kraći tekst koji zadržava samo bitne informacije. Sumiranje može biti ekstraktivno (izdvajanje rečenica iz originalnog teksta) ili abstraktivno (generiranje novih rečenica koje sažimaju informacije)

S druge strane, NLP nije ograničen samo na pisani tekst. Također se bavi složenim izazovima u prepoznavanju govora i računalnim vidom, kao što je generiranje prijepisa audio uzoraka ili opisa slika [2].

NLP je izazovno područje jer računala ne obrađuju informacije na isti način kao ljudi. Na primjer, kada pročitamo rečenicu "Gladan sam", lako možemo razumjeti njezino značenje. Slično tome, s obzirom na dvije rečenice kao što su "gladan sam" i "tužan sam", lako možemo odrediti koliko su slične. Za modele strojnog učenja takvi su zadaci teži. Tekst treba obraditi na način koji modelu omogućuje učenje iz njega. Budući da je jezik složen, moramo pažljivo razmisliti o tome kako se ova obrada mora izvršiti [3]. Provedeno je mnogo istraživanja o tome kako predstaviti tekst, a neke metode bit će prikazane u nastavku.

2.1. Transformeri

Objavom rada *Attention Is All You Need* [4] 2017. godine, Googleov istraživački tim predložio je novu arhitekturu neuronskih mreža nazvanu Transformer koja je označila revoluciju u svijetu obrade prirodnog jezika. Ta arhitektura nadmašila je do tad preferirane rekurentne neuronske mreže (engl. *Recurrent neural networks*) u zadacima strojnog prevođenja. Paralelno s tim, metoda prijenosnog učenja (engl. *Transfer learning*) ULMFiT (engl. *Universal Language Model Fine-tuning*) pokazala je da se učenjem LSTM (engl. *Long short-term memory networks*) mreža na velikim i raznolikim korpusima mogu postići vrhunski klasifikatori teksta s malo označenih podataka. Ovi napretci kasnije su doveli do razvoja poznatih modela kao što su GPT i BERT, koji su kombiniranjem arhitekture Transformera s nenadziranim učenjem (engl. *Unsupervised learning*) postigli izvanredne rezultate u zadacima obrade prirodnog jezika [5]. Vremenski razvoj tih i ostalih značajnijih transformera prikazan je na slici 2.1.

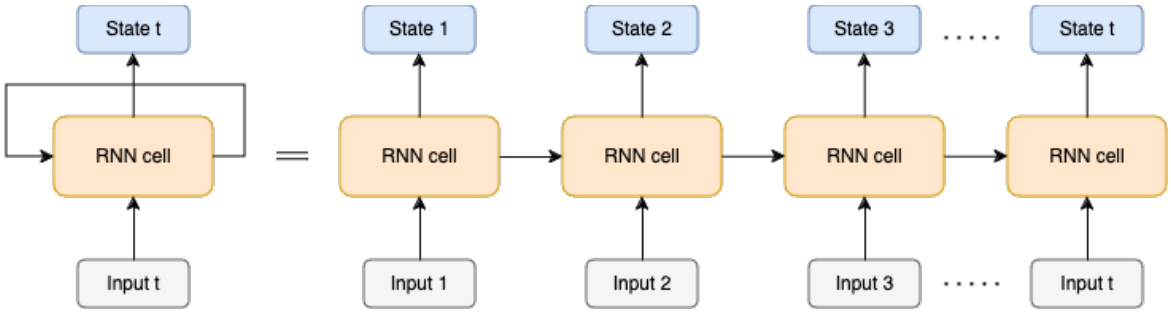


Slika 2.1. Razvoj transformera

2.1.1. Arhitektura koder-dekoder

Prije uvođenja transformera, rekurentne arhitekture (engl. *recurrent neural networks - RNN*) poput LSTM-a postizale su vrhunske rezultate u obradi prirodnog jezika. Specifičnost te arhitekture je povratna petlja koja omogućava prijenos informacija s jednog koraka na drugi, što ju čini idealnom za modeliranje sekvencijalnih podataka poput teksta. Kao što je prikazano na lijevoj strani slike 2.2., RNN prima neki ulaz (npr. riječ ili znak), propušta ga kroz mrežu i daje izlazni vektor (skriveno stanje). Istovremeno, model si prosljeđuje dio informacija natrag kroz povratnu petlju, kako bi ih mogao koristiti u sljedećem koraku, što je vidljivo na desnoj strani slike 2.2. To omogućuje RNN-u praćenje informacija iz prethodnih koraka i njihovo korištenje za predikcije izlaza [5].

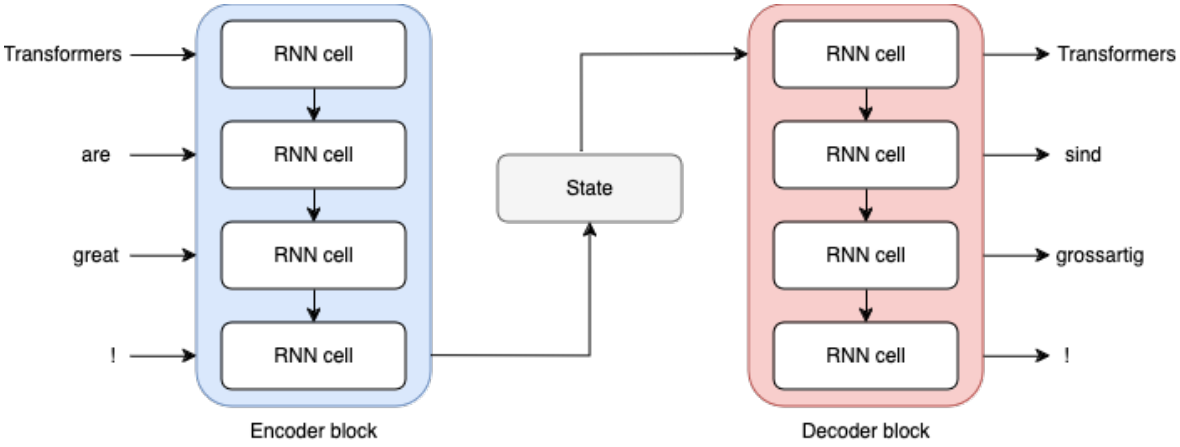
Iako je pojava transformera označila novo razdoblje u obradi prirodnog jezika, spo-



Slika 2.2. Način rada rekurentne neuronske mreže

menute arhitekture su i dalje često korištene za zadatke u NLP-u. Područje u kojem je RNN igrao važnu ulogu je razvoj sustava za strojno prevođenje, gdje je cilj preslikati sekvencu riječi s jednog jezika na drugi. Ovakav se zadatak obično rješava pomoću arhitekture koder-dekoder ili sekvenca-sekvenca, koja je dobro prilagođena situacijama gdje su i ulaz i izlaz sekvence proizvoljne duljine. Zadatak enkodera je kodirati informacije iz ulazne sekvence u određenu numeričku reprezentaciju koja se često naziva posljednje skriveno stanje. To se stanje zatim prosljeđuje dekoderu, koji generira izlaznu sekvencu [5].

Općenito, komponente enkodera i dekodera mogu biti bilo koja vrsta neuronske mreže koja može modelirati sekvence. To je na slici 2.3. prikazano za par RNN-ova, gdje je rečenica na engleskom "Transformers are great!" kodirana kao skriveni vektor koji se potom dekodira kako bi se generirao njemački prijevod "Transformer sind grossartig!" Ulazne riječi se sekvencijalno prosljeđuju kroz enkoder, a izlazne riječi se generiraju jedna po jedna, od vrha prema dnu [5].



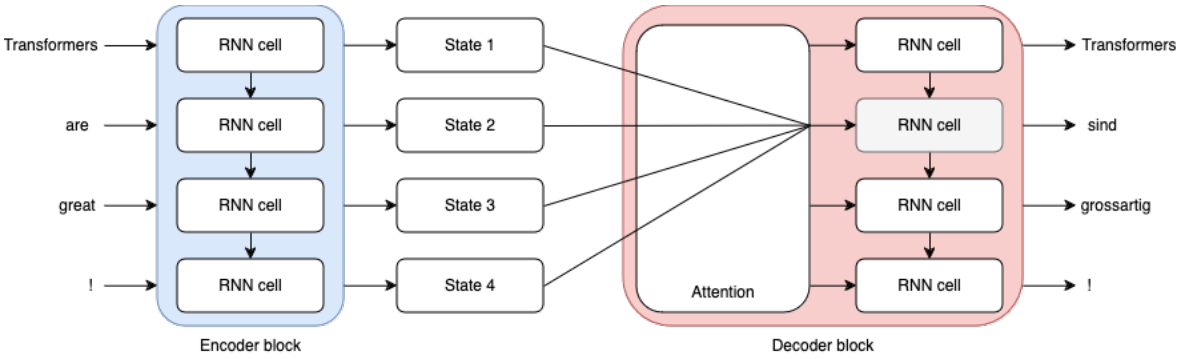
Slika 2.3. Kodersko-dekoderska arhitektura RNN-a

Iako jednostavna, slabost ove arhitekture je da konačno skriveno stanje enkodera

stvara usko grlo: ono mora predstavljati značenje cijele ulazne sekvence jer je to sve što dekoder ima na raspolaganju prilikom generiranja izlaza. To je posebno izazovno za duge sekvence, gdje informacije s početka sekvence mogu biti izgubljene u procesu komprimiranja u jednu, fiksnu reprezentaciju. Dopuštanjem dekoderu pristup svim skrivenim stanjima enkodera moguće je doskočiti ovom problemu. Opći mehanizam za to naziva se pažnja (engl. *Attention*) i ključna je komponenta mnogih modernih arhitektura neuronskih mreža [5].

2.1.2. Mehanizam pažnje

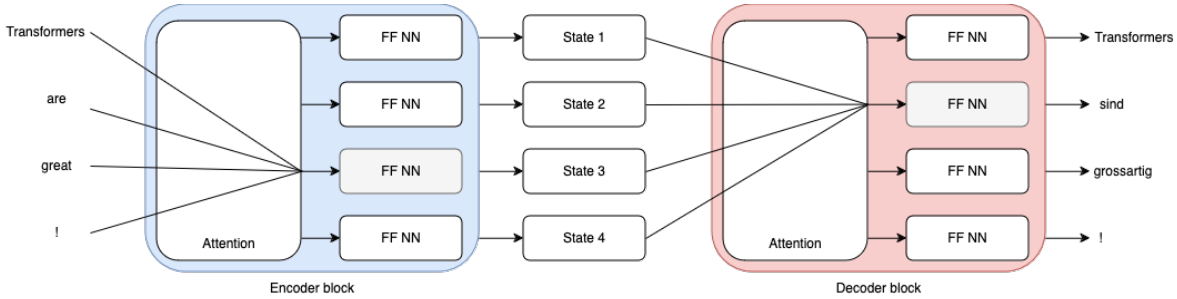
Glavna ideja iza mehanizma pažnje je omogućiti dekoderu pristup svim skrivenim stanjima enkodera umjesto jednog stanja za cijelu ulaznu sekvencu. Umjesto da dekoder koristi sva skrivena stanja odjednom, pažnja mu omogućuje da dodijeli različite težine svakom stanju u svakom koraku dekodiranja, čime se prioritiziraju najrelevantnija stanja [5]. Ovaj mehanizam je ilustriran u predviđanju trećeg tokena u izlaznoj sekvenci na slici 2.4.



Slika 2.4. Enkoder-dekoder RNN arhitektura s mehanizmom pažnje

Pažnja omogućuje modelima da nauče složena uparivanja između riječi u generiranom prijevodu i izvornom tekstu. Ipak, modeli poput LSTM-ova i dalje pate od ograničenja inherentno sekvencijalnih izračuna, koji se ne mogu paralelizirati. Ova ograničenja riješena su uvođenjem transformera, koji se oslanja isključivo na samo-pažnju (engl. *self attention*) i eliminira potrebu za rekurencijom [5].

U modelu transformera, enkoder i dekoder imaju svoje mehanizme samo-pažnje, koji imaju pristup svim skrivenim stanjima u istom sloju neuronske mreže. Izlazi iz ovih slojeva potom se prosljeđuju neuronskim mrežama s unaprijednim prosljeđivanjem (engl. *feed forward neural networks*), što čini arhitekturu osnovnog transformera (slika 2.5.). Ova arhitektura omogućuje brže učenje modela u usporedbi s rekurentnim modelima, što je dovelo do brojnih novih uspjeha u obradi prirodnog jezika [5].



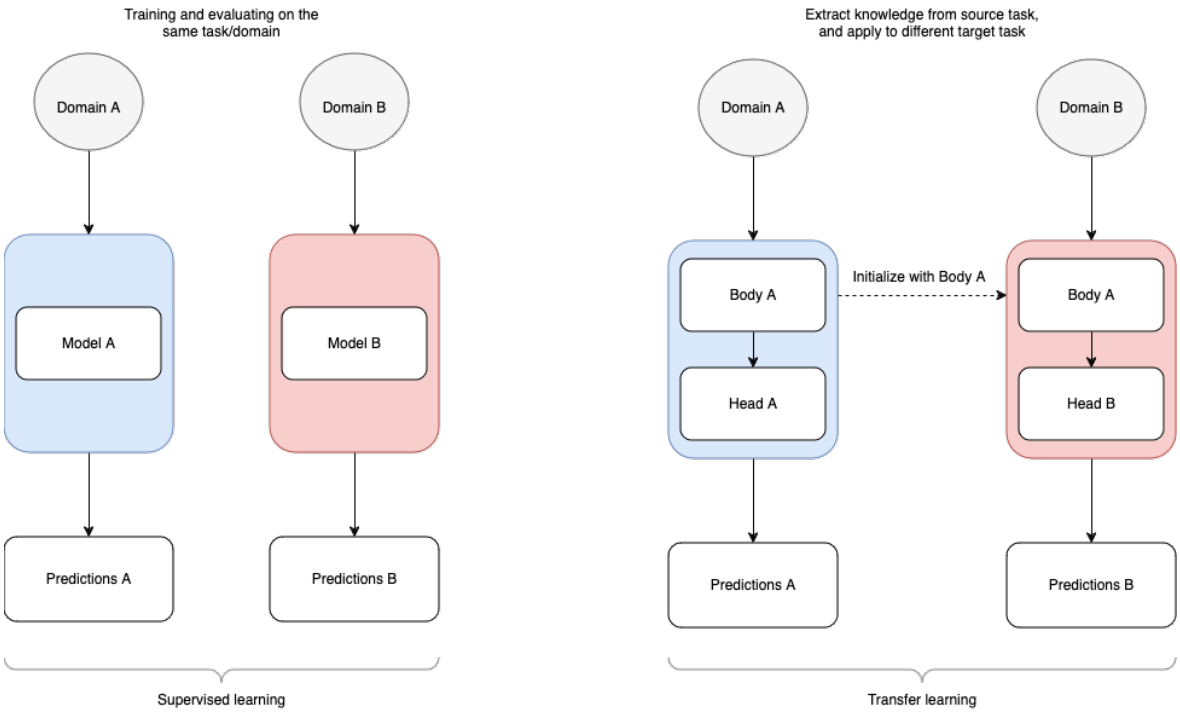
Slika 2.5. Arhitektura transformera

Prvi Transformer učen je od nule na velikom skupu parova rečenica na različitim jezicima. Nažalost, za mnoge praktične primjene obrade prirodnog jezika ne postoje veliki skupovi označenih podataka. Tom problemu je doskočeno tzv. prijenosnim učenjem (engl. *transfer learning*).

2.1.3. Prijenosno učenje

U današnje vrijeme, prijenosno učenje je uobičajena praksa u računalnom vidu, gdje se konvolucijske neuronske mreže poput ResNeta uče za jedan zadatak, a zatim fino podešavaju za drugi (engl. *fine tuning*), pri čemu mreža koristi znanje otprije stečeno na izvornom zadatku. Strukturalno, to podrazumijeva podjelu modela na tijelo i glavu, pri čemu je glava specifična za pojedini zadatak. U procesu učenja težine tijela podešavaju se općim značajkama izvorne domene, a kasnije se koriste za inicijalizaciju novog modela tijekom finog podešavanja. Ovaj pristup omogućuje gradnju modela koji se mogu učiti mnogo učinkovitije za specifične zadatke obrade prirodnog jezika s manje dostupnih označenih podataka [5]. Usporedba ovog (desno) i tradicionalnog nadziranog učenja (lijevo) prikazana je na slici 2.6.

U računalnom vidu, modeli se prvo uče na velikim skupovima podataka poput ImageNeta, koji sadrže milijune slika. Ovaj proces se zove predučenje, a njegova je svrha na-



Slika 2.6. Usporedba tradicionalnog nadziranog i prijenosnog učenja

učiti modele osnovnim značajkama slika, poput rubova ili boja. Prednaučeni modeli se zatim fino podešavaju za specifične zadatke poput klasifikacije vrsta cvijeća s relativno malim brojem označenih primjera. Fino podešeni modeli obično postižu veću točnost nego modeli učeni od nule na istim podacima.

Iako je prijenosno učenje postalo standardni pristup u računalnom vidu, dugo vremena nije bilo jasno što je analogni proces predučenja za obradu prirodnog jezika. Kao rezultat, aplikacije NLP-a obično su zahtijevale velike količine označenih podataka za postizanje usporedivih performansi. Međutim, 2017. i 2018. godine, nekoliko istraživačkih grupa je predložilo nove pristupe koji su konačno omogućili prijenosno učenje u NLP-u. Istraživači iz OpenAI-a postigli su snažne performanse na zadatku analize sentimenta koristeći značajke izvučene iz nenadgledanog predučenja. Nakon toga je ULMFiT uveo opći okvir za prilagodbu prednaučenih LSTM modela za razne zadatke [5], a uključuje tri glavna koraka navedena u nastavku:

- **Predučenje:** Početni cilj učenja je predviđanje sljedeće riječi na temelju prethodnih riječi (jezično modeliranje). Ovaj pristup ne zahtijeva označene podatke, što znači da može koristiti obilje dostupnog teksta iz različitih izvora (npr. Wikipedija).

- **Prilagodba domeni:** Nakon što je jezični model prednaučen na velikom skupu podataka, sljedeći korak je prilagodba specifičnom podatkovnom skupu (npr. baza filmskih recenzija IMDB). Ova faza također koristi jezično modeliranje, ali sada model mora predviđati sljedeću riječ u ciljanom skupu podataka.
- **Fino podešavanje:** U ovom koraku, jezični model se fino podešava s klasifikacijskim slojem za ciljani zadatak (npr. klasifikacija sentimenta filmskih recenzija).

U nastavku su opisani transformeri korišteni za potrebe implementacije aplikacije:

- **BERT** [6] - korišten za klasifikaciju stavki računa po kategorijama (*Beauty, Electronics, Fashion, Groceries, Home & Kitchen*)
- **Donut** [7] - korišten za parsiranje i izdvajanje podataka sa slika računa
- **Seamless** [8] - korišten za prevođenja skupa podataka (*eng.* u *hrv.*) za fino podešavanje modela BERT

2.1.4. BERT

BERT (engl. *Bidirectional Encoder Representations from Transformers*) je revolucionaran model koji su 2018. godine u radu *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding* [6] predstavili istraživači iz Googlea. Postigao je vrhunske rezultate u zadacima obrade prirodnog jezika (NLP), kao što su odgovaranje na pitanja (engl. *The Stanford Question Answering Dataset - SQuAD v1.1*) i jezična inferencija (engl. *The Multi-Genre Natural Language Inference - MNLI*). Ključna tehnička inovacija BERT-a je primjena dvosmjernog učenja Transformera u svrhu jezičnog modeliranja. Ovakav pristup bio je u suprotnosti s prethodnima, koji su obrađivali tekst slijeva nadesno ili su kombinirali obradu slijeva-nadesno i zdesna-nalijevo. Rezultati su pokazali da jezični model koji se uči na taj način ima dublji osjećaj za kontekst i tok jezika. U radu je detaljno opisana i tehnika nazvana *Maskirani jezični model* (engl. *Masked language model*) (MLM) koja omogućuje dvosmjerno učenje u modelima u kojima to prije nije bilo moguće [6].

Model je sagrađen na prethodno opisanoj arhitekturi Transformera i koristi mehanizam pažnje koji uči kontekstualne odnose između riječi ili podriječi u tekstu. Osnovna arhitektura Transformera uključuje dva već spomenuta mehanizma: enkoder koji ulaznoj sekvenci daje numeričku reprezentaciju i dekoder koji na temelju te reprezentacije generira određenu izlaznu sekvencu. Kako je cilj BERT-a generiranje jezičnog modela, potreban je samo enkoder koji čita cijeli niz riječi odjednom. Zato se BERT smatra dvosmjernim (engl. *bidirectionoal*) modelom, iako bi bilo točnije reći da je neusmjeren. Ova karakteristika omogućuje modelu da uči kontekst riječi na temelju njenog okruženja (lijevo i desno od riječi). U nastavku je opisan postupak obrade ulazne sekvence kod modela BERT [6].

Način rada

Proces počinje tokenizacijom ulaznog teksta, koji se zatim ugrađuje u vektore kombinirajući token, pozicijska i segmentna ugrađivanja. Ovi vektori predstavljaju ulaz u enkoder, koji koristi mehanizme samopažnje za razumijevanje konteksta svakog tokena. Konačno, izlaz je niz vektora, gdje svaki predstavlja token u ulaznom tekstu s pripadajućim kontekstnim informacijama. Ovaj mehanizam omogućuje generiranje vrlo kontekstu-

aliziranih prikaza teksta, stvarajući pogodnost za široku upotrebu u NLP-u [6].

1. **Tokenizacija**

   - **Ulazna sekvenca** - ulaz u model je tekst koji treba obraditi
   - **Tokenizacija** - ulazni tekst se razlaže na manje jedinice (tokene). Ovo uključuje dijeljenje teksta na riječi i podriječi koristeći već poznatu metodu, a jedna od najčešće korištenih je *WordPiece*. U njoj riječ "igranje" može biti podijeljeno na "igra" i "##nje" gdje "##" označava podriječ.
   - **Dodavanje posebnih tokena** - BERT koristi posebne tokene poput [CLS] (klasifikacijski token) na početku sekvence i [SEP] (token separator) za označavanje kraja sekvence ili odvajanje rečenica.

2. **Ugrađivanje u vektorsku reprezentaciju** (engl. *Embedding*)

   - **Ugrađivanje tokena** (engl. *Token embeddings*) - Tokeni se pretvaraju u svoje vektorske reprezentacije, pri čemu vektori enkapsuliraju značenje tokena u numeričkom obliku.
   - **Pozicijsko ugrađivanje** (engl. *Position embeddings*) - Kako transformeri inherentno ne razumiju redoslijed tokena, dodaju se pozicijski vektori kako bi se kodirala njihova pozicija u sekvenci. Ovo pomaže modelu da razumije redoslijed sekvence.
   - **Segmentno ugrađivanje** (engl. *Segment embeddings*) - Za zadatke koji uključuju parove rečenica, dodaju se segmentni vektori kako bi se označilo koji token pripada kojoj rečenici.

3. **Ulazni prikaz**

   - Konačni ulazni prikaz za svaki token dobiva se zbrajanjem njegovih tokenskih, pozicijskih i segmentnih ugrađivanja (engl. *embeddings*). Ovaj kompozit ugrađivanja zatim se unosi u model BERT.

4. **Obrada**

    - **Mehanizam samopažnje** (engl. *Self-Attention*) - BERT koristi mehanizam višestruke samopažnje (engl. *multi-head self-attention*) kako bi tokenima omogućio fokus na različite dijelove ulazne sekvence, uključujući samog sebe. Ovo pomaže u razumijevanju konteksta oko svakog tokena.

    - **Slijedne neuronske mreže** (engl. *Feed-Forward Networks - FFNN*) - Izlaz iz slojeva samopažnje prolazi kroz niz slijednih neuronskih mreža kako bi se generirali konačni prikazi tokena.

5. **Izlazni vektor**

    - **Niz vektora** - Izlaz modela je niz vektora, gdje svaki vektor odgovara tokenu u ulaznoj sekvenci, održavajući isti redoslijed.

    - **Veličina vektora (H)** - Veličina svakog vektora je fiksna i određena konfiguracijom modela (npr. BERT-base ima skrivenu veličinu od 768). Ovaj vektor obuhvaća kontekstualne informacije o tokenu.

Primjer načina rada za rečenicu "BERT je nevjerojatan"

1. **Tokenizacija**

    - **Ulazna rečenica** - "BERT je nevjerojatan"

    - **Dodavanje posebnih tokena** - token [CLS] na početku i [SEP] na kraju rečenice

    - **Tokeni** - ["[CLS]", "BERT", "je", "ne", "vjer", "oj", "atan", "[SEP]"]

2. **Ugrađivanje u vektorsku reprezentaciju** (engl. *Embedding*)

    - **Ugrađivanje tokena** (engl. *Token embeddings*) - pretvaranje tokena u vektore: ["v_cls", "v1", "v2", "v3", "v4", "v5", "v6", "v_sep"]

    - **Pozicijsko ugrađivanje** (engl. *Position embeddings*) - dodavanje pozicijskih vektora: ["p_cls", "p1", "p2", "p3", "p4", "p5", "p6", "p_sep"]

- **Segmentno ugrađivanje** (engl. *Segment embeddings*) - dodavanje segmentnih vektora: ["s_cls", "s1", "s2", "s3", "s4", "s5", "s6", "s_sep"]

3. **Ulazni prikaz**

   - Zbrajanje prethodnih ugrađivanja za svaki token - ["v_cls + p_cls + s_cls", "v1 + p1 + s1", "v2 + p2 + s2", "v3 + p3 + s3", "v4 + p4 + s4", "v5 + p5 + s5", "v6 + p6 + s6", "v_sep + p_sep + s_sep"]

4. **Obrada**

   - **Mehanizam Samopažnje** (engl. *Self-Attention*) - svaki token može se fokusirati na različite dijelove sekvence, uključujući i samog sebe, npr. token "[CLS]" može uzeti u obzir "BERT", "je", "ne", "vjer", "oj", "atan", "[SEP]" i obratno

   - **Slijedne Neuronske Mreže** (engl. *Feed-Forward Networks - FFNN*) - izlaz iz slojeva samopažnje prolazi kroz niz slijednih neuronskih mreža kako bi se generirali konačni prikazi vektora

5. **Izlazni vektor**

   - **Konačni vektor** - ["h_cls", "h1", "h2", "h3", "h4", "h5", "h6", "h_sep"], gdje svaki vektor sadrži kontekstualne informacije o odgovarajućem tokenu

Kod učenja jezičnih modela izražen je izazov definiranja cilja predikcije. Mnogi modeli predviđaju sljedeću riječ u sekvenci (npr. "Dijete je došlo kući iz ___"), gdje se inherentno ograničava učenje konteksta. Kako bi prevladao ovaj izazov, BERT koristi dvije strategije učenja:

Maskirano jezično modeliranje (engl. *Masked language modeling - MLM*)

Prije nego što se ulazna sekvenca proslijedi BERT-u, 15% riječi u svakoj sekvenci zamjenjuje se tokenom [MASK]. Model zatim pokušava predvidjeti originalnu vrijednost zamaskiranih riječi na temelju konteksta koji pružaju ostale riječi u sekvenci. Tehnički gledano, predviđanje izlaznih riječi zahtijeva:

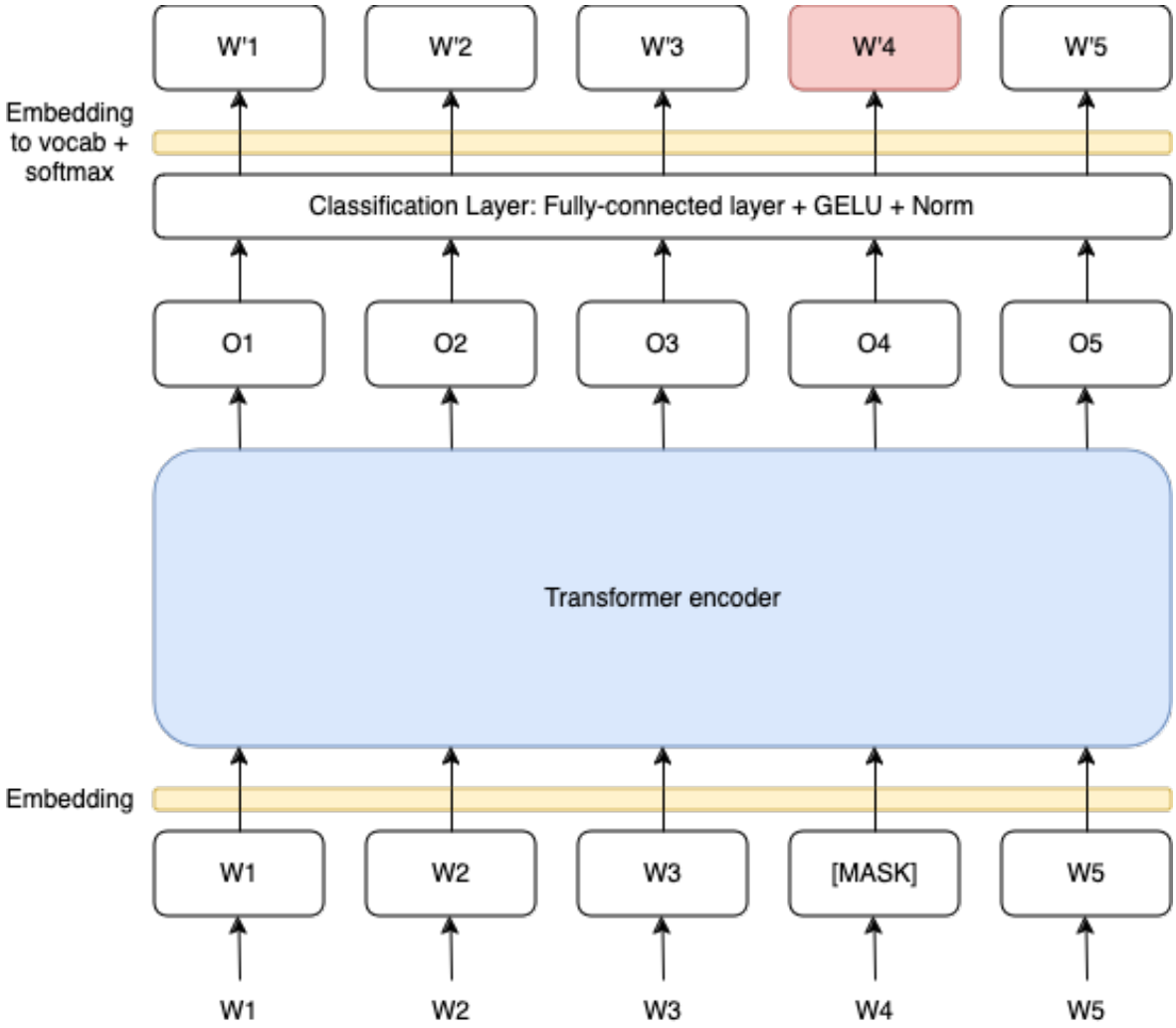
1. Dodavanje klasifikacijskog sloja na izlaz enkodera
2. Množenje vektora izlaza s ugnježđujućom matricom, transformirajući ih u dimenziju vokabulara
3. Izračunavanje vjerojatnosti svake riječi u vokabularu korištenjem funkcije *softmax*

Postupak maskiranog jezičnog modeliranja ulazne sekvence od pet riječi (w1, w2, w3, w4, w5) prikazan je na slici 2.7. BERT-ova funkcija gubitka uzima u obzir samo predviđanje maskiranih riječi i zanemaruje predviđanje onih nemaskiranih. Kao posljedica, model sporije konvergira u usporedbi sa smjernim modelima, što se u konačnici nadoknađuje povećanom sviješću o kontekstu [6].

Predviđanje sljedeće rečenice (engl. *Next sentence prediction - NSP*)

Model BERT u procesu učenja prima parove rečenica na ulazu i uči predviđati je li druga rečenica u paru sljedeća u originalnom dokumentu. Pritom 50% ulaza čine parovi u kojima je druga rečenica zaista sljedeća, dok se u preostalih 50% druga rečenica bira nasumično [6]. Da bi model pomogao razlikovati dvije rečenice u procesu učenja, ulaz se prije ulaska u model obrađuje na sljedeći način (slika 2.8.):

1. Token [CLS] umeće se na početak prve rečenice, a token [SEP] na kraj svake rečenice.
2. U svaki token dodaje se oznaka koja označava Rečenicu A ili Rečenicu B. Oznaka



Slika 2.7. Maskirano jezično modeliranje u modelu BERT

je slična konceptu ugrađenih oznaka za tokene, s vokabularom veličine 2.

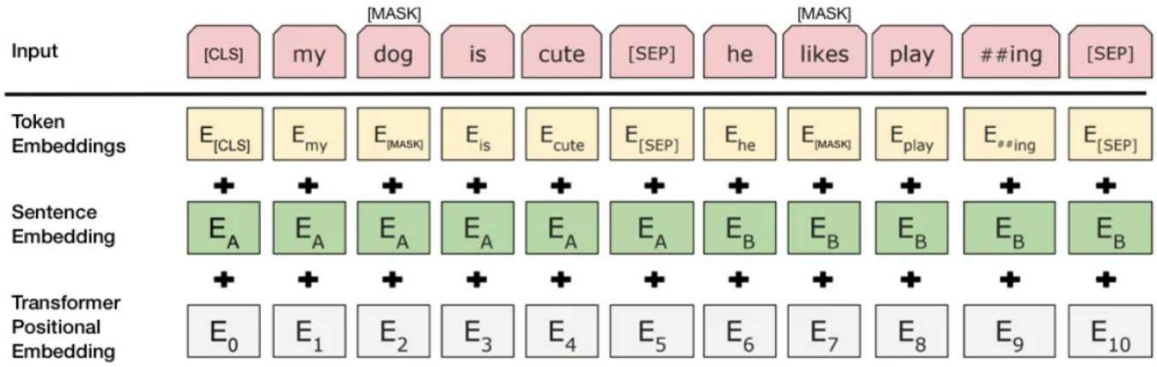
3. U svaki token se dodaje i pozicijska ugrađena oznaka koja označava njegov položaj u sekvenci.

Da bi model predvidio je li druga rečenica povezana s prvom, provode se sljedeći koraci:

1. Cijeli ulazni niz prolazi kroz Transformer.

2. Izlaz tokena [CLS] transformira se u vektor oblika 2x1, koristeći jednostavan sloj klasifikacije (naučene matrice težina i pomaka).

3. Izračunavanje vjerojatnosti da je druga rečenica sljedeća (*IsNextSequence*) korištenjem funkcije *softmax*.



Slika 2.8. Obrada ulaza u modelu BERT za potrebe NSP-a [6]

Prilikom učenja modela BERT, maskirano jezično modeliranje (MLM) i predviđanje sljedeće rečenice (NSP) provode se zajedno, u cilju minimizacije kombinirane funkcije gubitka oba pristupa.

Fino podešavanje

Kako bi se model BERT mogao koristiti za neki specifični zadatak u obradi prirodnog jezika, potrebno ga je fino podesiti korištenjem relevantnih podataka. Model ima svoju primjenu u širokom spektru jezičnih zadataka, uz dodavanje sloja na osnovni model [6]:

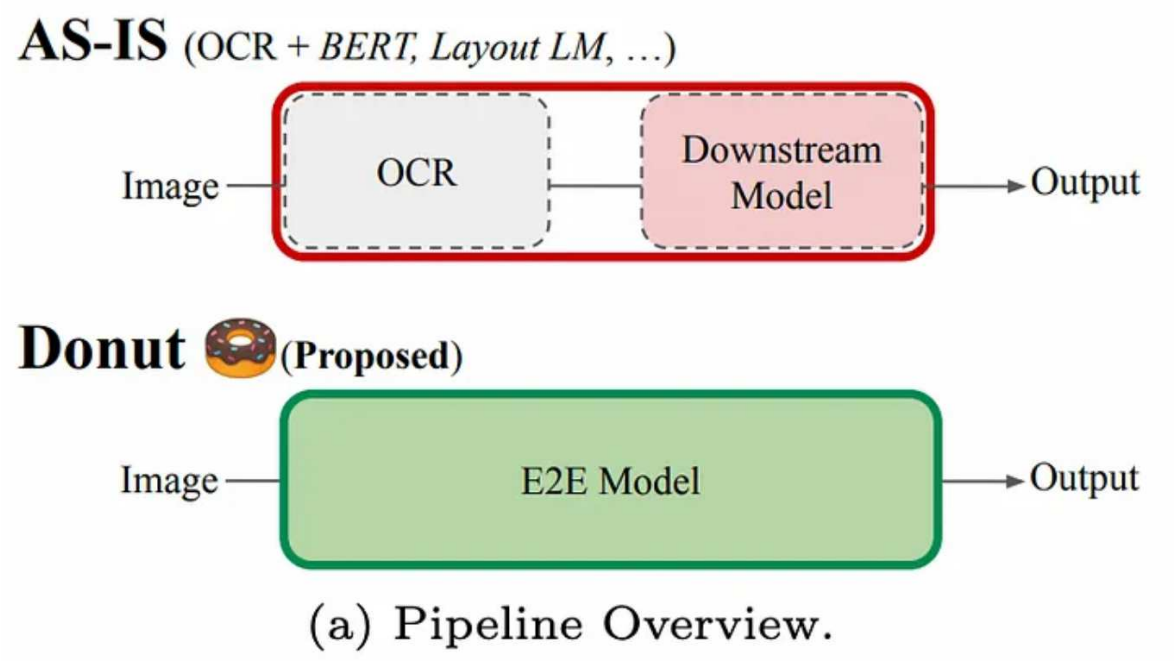
1. Za klasifikacijske zadatke poput analize sentimenta koristi se klasifikacijski sloj na izlazu Transformera za token [CLS], slično kao u postupku predviđanja sljedeće rečenice (NSP).

2. U zadacima odgovaranja na pitanja (npr. SQuAD v1.1), model prima pitanje vezano uz tekstualnu sekvencu i treba označiti odgovor u sekvenci. Koristeći BERT, takav model se može naučiti dodavanjem dva dodatna vektora koji označavaju početak i kraj odgovora.

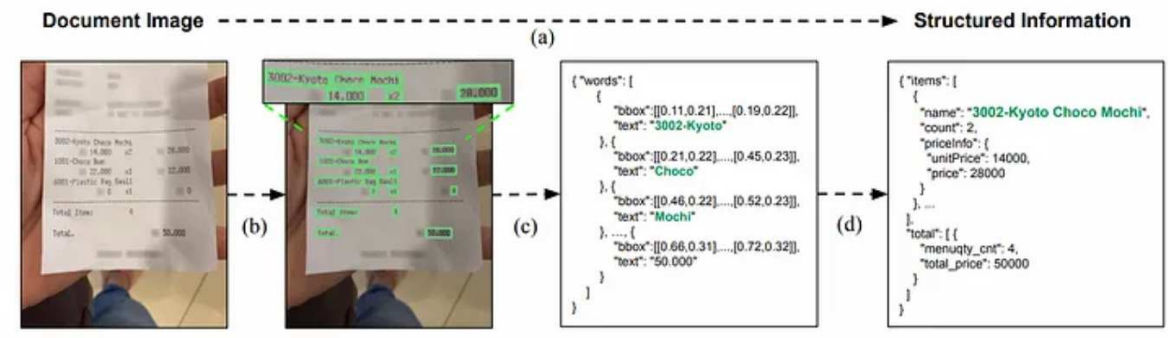
3. U prepoznavanju imenovanih entiteta (engl. *Named entity recognition - NER*), model prima tekstualnu sekvencu i treba označiti različite tipove entiteta (osoba, organizacija, datum itd.) koji se pojavljuju u tekstu. Koristeći BERT, model NER se može učiti prosljeđivanjem izlaznog vektora svakog tokena u klasifikacijski sloj koji predviđa NER-ovu oznaku.

2.1.5. Donut

Razumijevanje slikanih dokumenata poput računa predstavlja veliki izazov. Tradicionalne metode koriste tehnologiju OCR (engl. *Optical Character Recognition*) za čitanje teksta (slika 2.10), što je nerijetko računski zahtjevno, nefleksibilno i sklono pogreškama. Kako bi se doskočilo tom problemu 2021. godine objavljen je rad *OCR-free Document Understanding Transformer* [7] u kojem je predstavljen novi model *Donut* (engl. *Document Understanding Transformer*) koji revolucionarno pristupa ovom problemu. Donut izravno analizira slike dokumenata, eliminirajući potrebu za OCR-om, što rezultira poboljšanom brzinom i točnošću. Usporedba arhitekture tradicionalnog modela za parsiranje dokumenata i Donuta prikazana je na slici 2.9. Donut koristi napredni Transformer enkoder i dekoder, uz posebne tehnike predučenja koje modelu omogućuju duboko razumijevanje strukture i svojstava dokumenata. Učeći se na kombinaciji stvarnih i sintetičkih podataka, Donut postiže vrhunske rezultate na raznim zadacima parsiranja dokumenata, postavljajući nove standarde u području obrade prirodnog jezika i razumijevanja dokumenata.



Slika 2.9. Tradicionalni način parsiranja dokumenata (OCR) vs. Donut [7]



Slika 2.10. Tradicionalni način parsiranja dokumenata pomoću OCR-a [7]

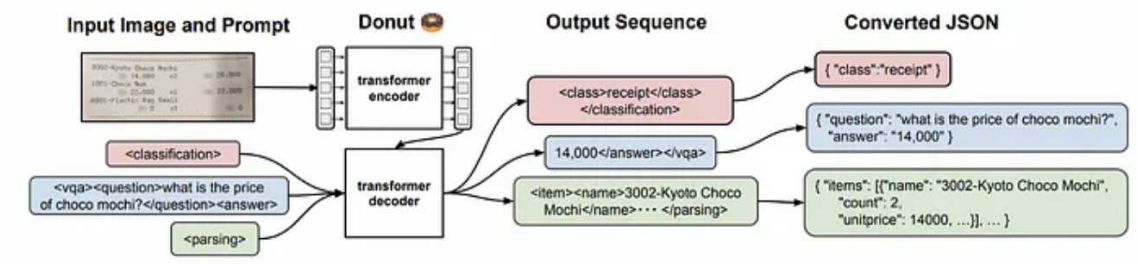
Generiranje podataka za učenje

Korištenje sintetičkog generatora dokumenata (engl. *Synthetic Document Generator - SDG*) za generiranje podataka korištenih u procesu predučenja postaje sve popularnije u zadacima razumijevanja dokumenata. Podaci generirani SDG-om obično su u obliku strukture XML koja sadrži informacije o dokumentu, poput slike, tekstualnih segmenata i pripadajućih okvira. SDG koristi slike iz ImageNeta za popunjavanje pozadine dokumenta i tekstove s Wikipedije za tekstualne segmente.

Korištenje SDG-a za generiranje podataka za pred-treniranje modela ima brojne prednosti. Može smanjiti komputacijske troškove i memorijske zahtjeve povezane s tradicionalnim metodama OCR-a za razumijevanja dokumenata. S druge strane, može povećati točnost modela pružajući smislenije sekvence ulaznih podataka. Konačno, omogućuje modelu veću fleksibilnost u radu s različitim vrstama dokumenata i jezicima [7].

Predučenje

Predučenje je ključni korak jer omogućuje modelu da nauči strukturu skupa podataka i odnose među različitim elementima podataka. Donut je cjeloviti (engl. *end-to-end*) model koji na ulazu prima sliku i na temelju zadanog upita daje odgovarajuće rezultate. Za predučenje modela korišten je spomenuti sintetički generator dokumenata (SDG) kojim je generirano pola milijuna uzoraka po jeziku za kineski, japanski, korejski i engleski jezik. Osim toga, korišten je i skup podataka IIT CDIP (engl. *Illinois Institute of Technology Complex Document Information Processing dataset*) koji sadrži jedanaest milijuna skeniranih slika dokumenata na engleskom [7].



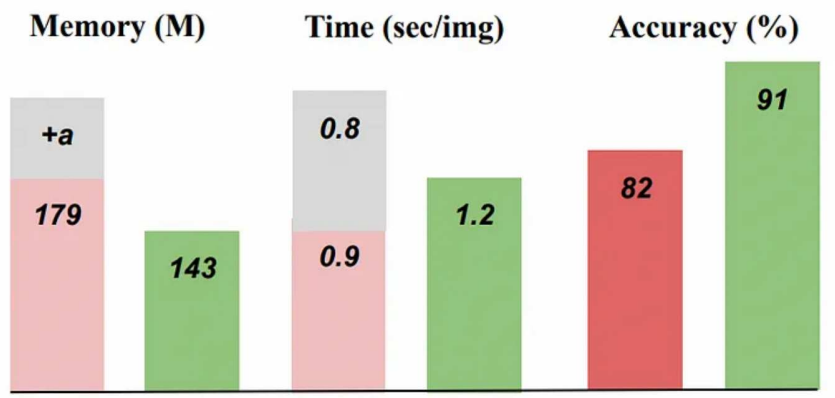
Slika 2.11. Ulazi i mogući izlazi Donuta [7]

Na slici 2.11. prikazani su mogući ulazi i pripadajući izlazi iz modela, od čega je jedan dio ulaza uvijek slika dokumenta, dok drugi dio čini upit. Na izlazu modela u svakom slučaju dobivamo JSON, koji je različito strukturiran ovisno o postavljenom upitu:

1. **Klasifikacija** - određivanje klase dokumenta (npr. račun)
2. **Odgovaranje na pitanja** - odgovaranje na postavljeno pitanje vezano za podatke sadržane na dokumentu (npr. Kolika je cijena nekog proizvoda?)
3. **Parsiranje** - parsiranje cijelog dokumenta i strukturirano odvajanje podataka (odvajanje pojedinih stavki, prepoznavanje ukupnog iznosa, datum)

Rezultati i performanse modela

Na različitim zadacima razumijevanja dokumenata predloženi model nadmašio je dotadašnje VDU (engl. *Visual Document Understanding*) modele bazirane na OCR-u u memoriji, vremenskoj učinkovitosti i točnosti (slika 2.12.). Ova postignuća otvorila su nove mogućnosti za daljnji razvoj i istraživanje tehnologija za razumijevanje dokumenata [7].



Slika 2.12. Usporedba performansi modela OCR (sivo + crveno) i Donut (zeleno) [7]

2.1.6. Seamless

SeamlessM4T (engl. *Massively Multilingual and Multimodal Machine Translation model*) je Metin (Meta Platforms, Inc.) multimodalni model sposoban za prevođenje i transkripciju jezika. Ovaj model podržava pretvaranje govora u tekst, kao i prevođenje govora u govor, čime efikasno eliminira potrebu za zasebnim modelima u različitim zadacima prevođenja. SeamlessM4T može obraditi do 100 jezika za prevođenje teksta i 35 jezika za prevođenje govora.

Razvoj modela imao je u cilju olakšati globalnu komunikaciju nudeći gotovo instantno prevođenje koje se može koristiti za potrebe različitih sustava za komunikaciju. Integrira se s postojećim platformama Mete kao što su Facebook, Instagram, WhatsApp i Messenger, unaprjeđujući korisničko iskustvo omogućavanjem olakšanih interakcija unatoč jezičnim barijerama.

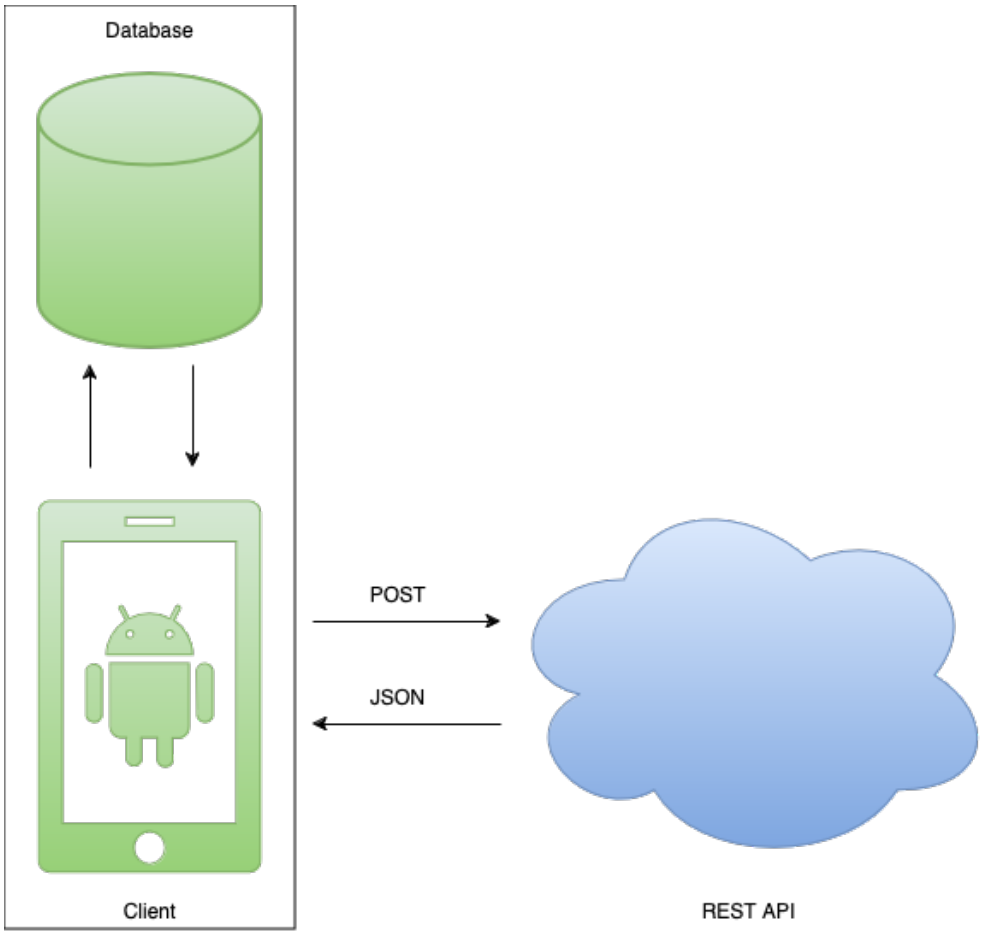
Jedna od značajnih prednosti ovog modela je njegova sposobnost rukovanja bučnim okruženjima u usporedbi s njegovim prethodnicima, osiguravajući smislene prijevode čak i u manje idealnim uvjetima. To je posebno korisno za korisnike mobilnih uređaja i one koji se nalaze u dinamičnim, stvarnim situacijama. Dodatno, Meta je objavila model i svoj skup podataka za učenje pod otvorenom licencom, u cilju poticanja daljnjih istraživanja i razvoja u području prevođenja potpomognutog umjetnom inteligencijom. Ovaj korak dio je Metine šire strategije korištenja umjetne inteligencija za stvaranje inkluzivnih i pristupačnih rješenja [8].

3. Izgradnja aplikacije

Izgradnja aplikacije obuhvaća izgradnju klijentske i poslužiteljske strane. Klijentska strana pruža korisničko sučelje za slikanje računa i uvid u pripadajuću vremensku analitiku potrošnje. S druge strane, poslužiteljska strana klijentskoj izlaže REST-ovu uslugu koji pruža sučelje za obradu poslane slike korištenjem već opisanih transformerskih modela (Donut i BERT). Pomoću Donuta obavlja se parsiranje slika računa (izdvajanje pojedinih stavki i dodatnih informacija o računu) koje kao rezultat daje JSON sa svim iščitanim podacima. Jedna od stavki JSON-a predstavlja popis stavki računa u obliku ugrađenih objekata JSON. Potom se za svaku od stavki obavlja klasifikacija pomoću fino podešenog modela BERT. Svaka stavka tako dobiva dodatno polje koje označava kategoriju, koja će kasnije biti ključna u vremenskoj analizi potrošnje. Konačno, svi podaci o skeniranim računima i pripadajućim stavkama spremaju se u bazu podataka na mobilnom uređaju. Tako klijentska strana kod prikaza otprije skeniranih računa ne ovisi o dostupnosti internetske veze i pruža bolje korisničko iskustvo. Detaljniji opis razvijenih funkcionalnosti dan je u nastavku.

3.1. Arhitektura sustava

Sustav je oblikovan u arhitekturalnom stilu REST i ima tri glavne komponente: klijentsku stranu (engl. *frontend*), poslužiteljsku stranu (engl. *backend*) i bazu podataka (engl. *database*). Skica arhitekture sustava prikazana je na slici 3.1., gdje je baza podataka smještena na mobilnom uređaju pa je omeđena pravokutnikom s klijentskom stranom.



Slika 3.1. Arhitektura sustava zasnovana na REST-u

REST (engl. *Representational State Transfer*) je arhitekturalni stil za oblikovanje usluga na webu. Kada klijent pošalje zahtjev prema REST-ovom poslužitelju, poslužitelj ovisno o korištenoj metodi HTTP-a (GET, POST, PUT, DELETE) obavlja određenu akciju nad podacima i vraća odgovarajući odgovor klijentu.

U razvijenoj aplikaciji, klijentsku stranu čini mobilni uređaj s pokrenutom klijentskom aplikacijom koja komunicira s poslužiteljskom stranom, tj. poslužiteljem zasnovanom na REST-u. Klijent s poslužiteljem komunicira protokolom HTTP, koristeći zahtjeve POST. HTTP je komunikacijski protokol koji omogućava dohvaćanje i slanje resursa na

webu. Podaci se dohvaćaju zahtjevima GET, a šalju se koristeći zahtjeve POST. Podaci koji se šalju su u formatu JSON. JSON (engl. *JavaScript Object Notation*) je standardni format razmjene i pohrane podataka u obliku parova ključ-vrijednost (engl. *key-value pairs*). Konačni JSON koji se formira kao rezultat korisničkog odabira slike računa i naziva trgovine, šalje se kao dio tijela (engl. *body*) HTTP-ovog zahtjeva POST poslužiteljskoj strani na obradu. Nakon obrade i dobivanja odgovora (engl. *response*) sa poslužiteljske strane, obrađeni podaci o računu i pripadajućim stavkama spremaju se u bazu podataka Realm na uređaju. Realm je NoSQL baza podataka dizajnirana za mobilne uređaje. Umjesto standardnog relacijskog modela poput onog u SQL bazama podataka, koristi objektno-orijentirani model koji omogućava brže operacije i jednostavnije upravljanje podacima.

Poslužiteljsku stranu čini pokrenuti REST poslužitelj. Poslužitelj prima HTTP-ove zahtjeve POST klijentske strane i obrađuje poslanu sliku računa korištenjem Transformera Donut i BERT. Korištenjem rezultata obrade formira se konačni JSON koji se u tijelu HTTP-ovog POST odgovora vraća klijentskoj strani.

3.2. Korištene tehnologije i alati

Klijentska aplikacija izgrađena je za operacijski sustav Android, uz korištenje programskog jezika Kotlin. Android je operacijski sustav temeljen na Linuxovoj jezgri (engl. *Linux kernel*). Razvio ga je Google, a namijenjen je uređajima kao što su pametni telefoni (engl. *smartphone*), tableti i pametni televizori (engl. *smart TV*). Android omogućava razvoj aplikacija u programskom jeziku Java, ali i u suvremenim jezicima kao što je Kotlin, koristeći Androidove razvojne alate (engl. *Android Software Development Kit - Android SDK*). Androidov ekosustav ističe se svojom otvorenom platformom, širokom podrškom za aplikacije iz Google Playa te integracijom s brojnim Googleovim uslugama i servisima.

Kotlin je moderni programski jezik koji je razvio JetBrains, koji se za potrebe razvoja Android aplikacija koristi od 2017. godine. Zahvaljujući svojoj jednostavnosti, brzini razvoja i smanjenju količine *boilerplate* koda u usporedbi s Javom, Google ga je 2019. proglasio preferiranim jezikom za razvoj Android aplikacija. Ističe se svojom izražajnom sintaksom, visokom sigurnošću tipova, podrškom za funkcionalno i objektno-

orijentirano programiranje te interoperabilnošću s postojećim Javinim kodom.

U cilju postizanja nesmetanog korisničkog iskustva i bolje organiziranosti koda, kao i buduće nadogradivosti, u ovoj aplikaciji korištene su suvremene biblioteke i arhitekturalni obrasci razvoja. Biblioteke AndroidXa: Core KTX i Lifecycle KTX omogućavaju integraciju s Kotlinom i sofisticirano upravljanje životnim ciklusom aplikacije. Biblioteka Jetpack Compose omogućuje deklarativni pristup razvoju korisničkog sučelja koji dramatično smanjuje kompleksnost i količinu koda potrebnog za kreiranje dinamičkih i interaktivnih elemenata korisničkog sučelja. Navigation Compose je biblioteka koja je korištena kako bi dodatno olakšala implementaciju navigacije, a predstavlja potporu navigaciji u aplikacijama napisanima u Composeu.

Umetanje ovisnosti (engl. *dependency injection*) postignuto je korištenjem biblioteke Hilt, koja radikalno pojednostavljuje upravljanje ovisnostima u aplikaciji te poboljšava modularnost, ispitljivost i održavanje koda. Biblioteke Retrofit i OkHttp korištene su za mrežnu komunikaciju i integraciju aplikacije s REST API-jem na poslužiteljskoj strani. Odgovori u obliko JSON-a koji stižu s poslužiteljske strane obrađuju se pomoću Gsona, koji olakšava serijalizaciju i deserijalizaciju podataka između Kotlinovih objekata i JSON-a.

Za lokalnu pohranu podataka na mobilnom uređaju odabrana je baza podataka Realm, nudeći pouzdan sustav tuoa NoSQL optimiziran za mobilne uređaje. Baza ima potporu za transakcije omogućavajući optimizirane upite te upravljanje podacima u obliku Kotlin objekata.

Na klijentskoj strani su u cilju pojednostaljavanja razvoja korištene dodatne biblioteke. Android Image Cropper koristi se kod izrezivanja slika prije slanja na obradu poslužiteljskoj strani, a Charts pruža vizualizacije podataka u obliku grafova, korištenih u dijelu sučelja koji prikazuje vremensku analizu potrošnje.

Poslužiteljska strana kombinira tehnologije i biblioteke koje omogućuju obradu slika računa i popratnu klasifikaciju pripadajućih stavki. Izlaganje REST servisa za obradu slika računa ostvareno je korištenjem biblioteke FastAPI. FastAPI je moderni Python web razvojni okvir (engl. *framework*) koji omogućuje jednostavno izlaganje REST servisa.

DonutProcessor iz Transformers biblioteke pruža implementaciju Transformera Donut, korištenog za parsiranje slika računa radi izdvajanja stavki računa i popratnih informacija o računu. Nakon inicijalne obrade i izdvajanja stavki provodi se klasifikacija pojedinih stavki korištenjem Transformer modela BERT, za što je iskorištena Torch biblioteka. Korištenje Torch-a bilo je ključno za fino podešavanje modela BERT, koji se na taj način prilagodio zadatku klasifikacije stavki računa na hrvatskom jeziku. Model se tako koristi za dodjelu kategorije svakoj stavki računa, što omogućava kasniju analizu potrošnje na klijentskoj strani.

Za fino podešavanje transformera korišteno je više Python biblioteka. Pandas je bio ključan za manipulaciju i analizu podataka, omogućujući jednostavno rukovanje podacima u obliku podatkovnih okvira (engl. *dataframes*). Kao osnovni okvir za duboko učenje korišten je Pytorch, koji pruža jednostavno sučelje za definiranje i učenje neuronskih mreža. Pytorch klase Dataset i Dataloader iskorištene su za jednostavno učitavanje i obradu podataka u grupama (engl. *batch*), što je neophodno za kontrolirano učenje modela BERT u petlji učenja. Konačno, biblioteka Transformers pruža prednaučene modele transformera te alate za njihovu prilagodbu i fino podešavanje.

Programski alati korišteni za razvoj aplikacije su:

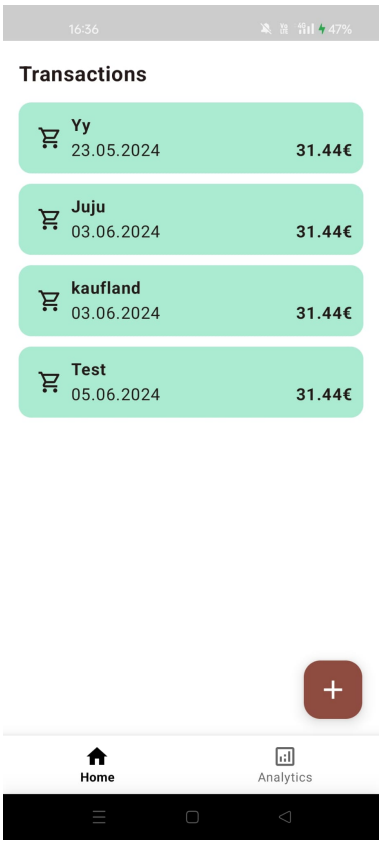
- **Android Studio** - klijentska strana
- **Visual Studio Code** - poslužiteljska strana
- **Google Colab** - fino podešavanje modela BERT

3.3. Klijentska strana

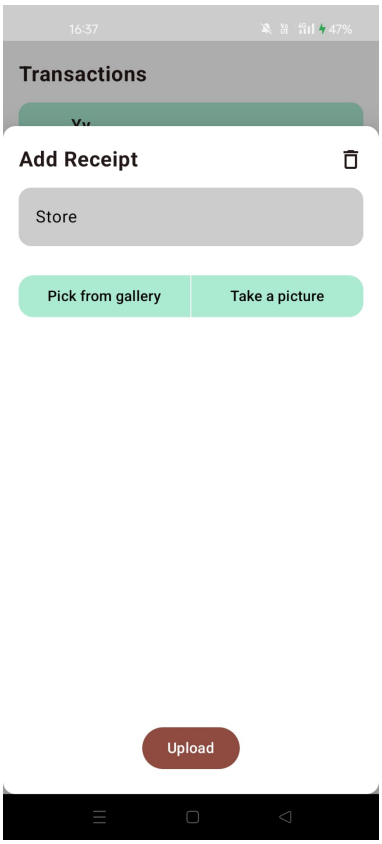
Klijentsku stranu sustava čini mobilna aplikacija za Android, koja pruža intuitivno korisničko sučelje i ostvaruje korisnu primjenu Transformera Donut i BERT. Za razliku od standardnog pristupa u razvoju Android aplikacija, u Jetpack Compose-u svaki ekran zapravo je funkcija (anotirana s @Compoable) koja se iscrtava na ekranu ovisno o trenutnoj ruti (engl. *route*). Dodatno, svaka takva funkcija može pozivati ostale *Composable* funkcije i na taj način gradi stablastu strukturu korisničkog sučelja (engl. *composable tree*). Pojedini dijelovi stabla ponovno se iscrtavaju ako se podaci o kojima ovise mijenjaju, što omogućuje usklađenost podataka i korisničkog sučelja u stvarnom vremenu. Ovaj koncept ključan je za tzv. deklarativnu paradigmu izrade korisničkog sučelja i predstavlja glavnu razliku u odnosu na standardnu imperativnu paradigmu. Podaci o kojima composable funkcije ovise izloženi su u pripadajućim ViewModel-ima kroz Kotlin Flow, koji omogućuje emitiranje i prikupljanje podataka tijekom vremena, po principu oblikovnog obrasca promatrač (engl. *observer*). View modeli upravljaju stanjem pripadajućih ekrana i poveznica su između korisničkog sučelja i poslovne logike aplikacije. Dio s poslovnom logikom enkapsulira metode za mrežnu i lokalnu manipulaciju podacima, a radi preglednosti odijeljen je u slojeve repozitorija (engl. *repository*) i izvora podataka (engl. *data source*). Repozitorij služi kao poveznica ViewModel-a i izvora podataka, koji direktno komunicira s funkcijama za lokalni (baza podataka) i mrežni pristup podacima (Retrofit API). Razvijeni ekrani s pripadajućim funkcionalnostima prikazani su u nastavku.

3.3.1. Funkcionalnosti

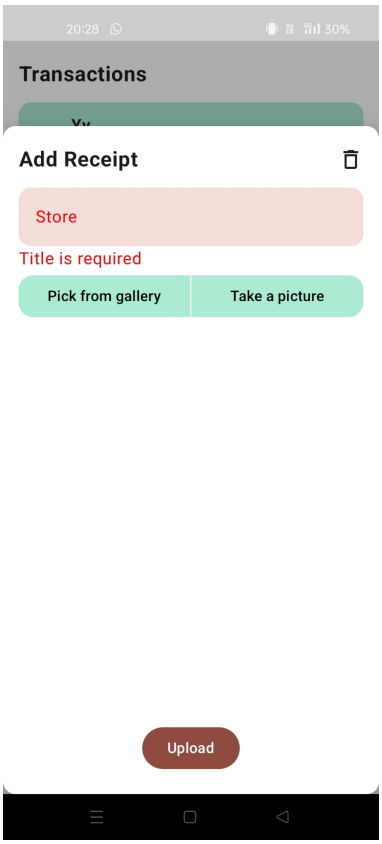
Prilikom pokretanja aplikacije otvara se ekran Home (slika 3.2.) na kojem su u listi prikazani svi skenirani računi. Svaki element liste sadrži ključne podatke o računu: naziv prodajnog mjesta, datum i ukupan iznos u eurima. Kako je korišteni model BERT fino podešen za rad s artiklima na hrvatskom jeziku, pretpostavka je da će uslikani računi biti na hrvatskom, pa prikaz svih iznosa u aplikaciji ima postfix €. U donjem desnom kutu ekrana nalazi se plutajući akcijski gumb (engl. *floating action button*) koji na klik otvara formu za dodavanje novog računa prikazanu na slici 3.3. Forma očekuje unos dvaju podataka: naziv prodajnog mjesta i sliku računa. Ovdje nisu dodana neka specifična ograničenja glede formata podataka, tako naziv prodajnog mjesta može biti bilo kakav znakovni niz (engl. *string*), a slika računa mora biti u nekom od formata za pohranu slika (npr. png, jpg, jpeg). Jedino postavljeno ograničenje za uspješan unos forme je unos svih podataka, što znači da slanje podataka na poslužitelj neće biti moguće ako je neko od polja prazno (slika 3.4.).



Slika 3.2. Ekran Home

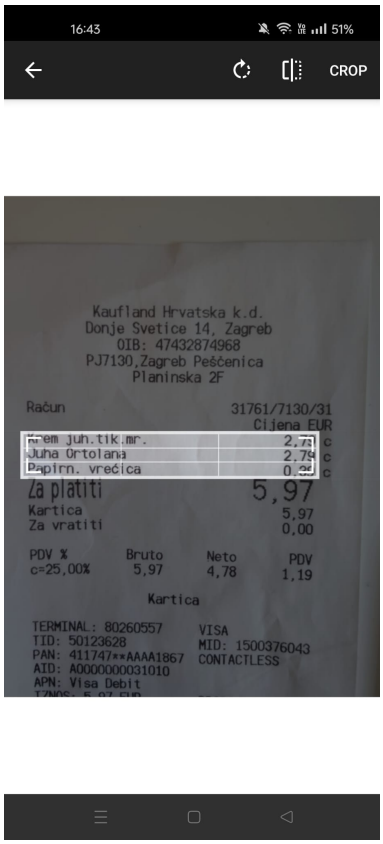


Slika 3.3. Forma za dodavanje računa

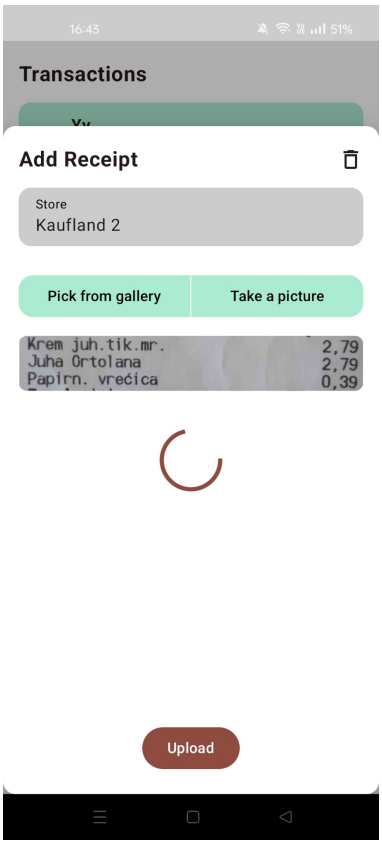


Slika 3.4. Neuspjelo slanje podataka na poslužitelj

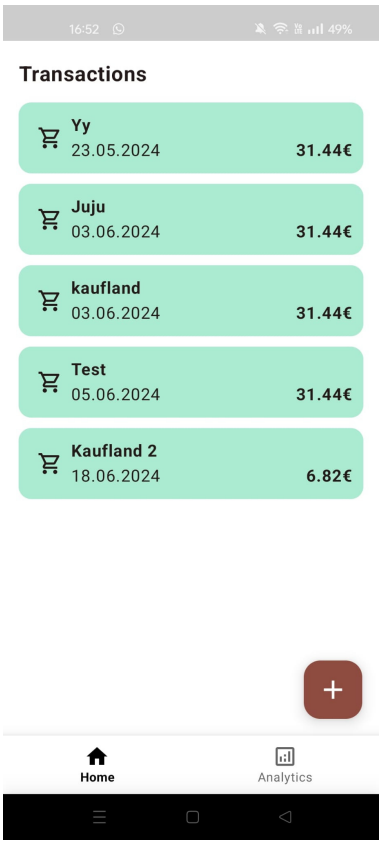
Pri odabiru slike računa moguće je odabrati sliku iz galerije (opcija *Pick from gallery*) ili korištenjem kamere uslikati željeni račun (opcija *Take a picture*). Jednom kada je slika računa spremna, prikazuje se ekran za izrezivanje (slika 3.5.), gdje je korisniku omogućeno izrezivanje dijela računa s prikazom stavki. Dodatno, ovdje je sliku moguće zakrenuti (engl. *rotate*) i zrcaliti (engl. *mirror*). Ako su svi traženi podaci uneseni u formu, klikom na gumb *Upload* omogućeno je njihovo slanje na poslužitelj. Tijekom obrade podataka korisniku se prikazuje kružna traka napretka (engl. *circular progress indicator*) (slika 3.6.). Ako se tijekom obrade podataka dogodila greška, ona će se ispisati korisniku iznad *Upload* gumba. Konačno, nakon uspješne obrade podataka na poslužitelju forma se zatvara, a novododani račun prikazan je na Home ekranu (slika 3.7.).



Slika 3.5. Izrezivanje dijela računa sa stavkama



Slika 3.6. Slanje podataka na poslužitelj



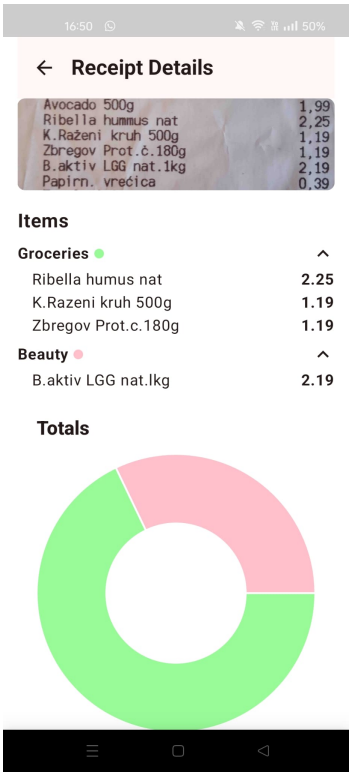
Slika 3.7. Dodani račun na Home ekranu

Klikom na pojedini račun u listi na ekranu Home otvara se ekran Details koji prikazuje detalje tog računa (slika 3.8.).

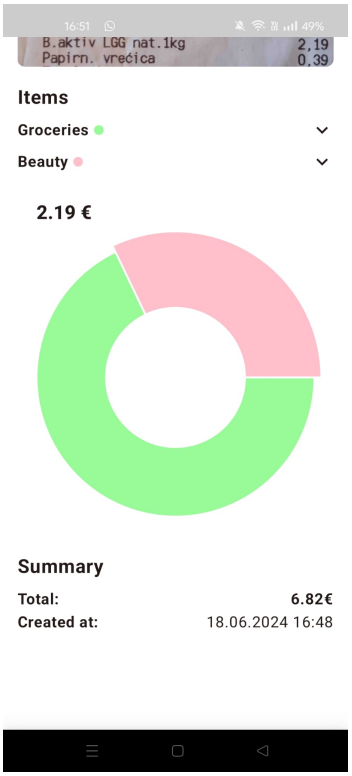
Na Details ekranu prikazana je slika računa i podaci dobiveni analizom istog na poslužitelju. Ti podaci obuhvaćaju:

- Stavke računa grupirane po kategorijama, pri čemu su moguće kategorije *Beauty, Electronics, Fashion, Groceries, Home & Kitchen*
- Grafički prikaz potrošnje po kategorijama
- Sažetak računa - datum i ukupni iznos

Ovdje je bitno napomenuti da svaka od pet predviđenih kategorija proizvoda ima svoju boju, koja je naznačena uz prikaz proizvoda pojedine kategorije. Boja je jednoznačna za svaku kategoriju i koristi se uniformno kroz cijelu aplikaciju. Konkretno, ovdje su na kružnom grafu (engl. *donut chart*) bojama prikazani udjeli potrošnje po kategoriji za promatrani račun. Klikom na strelice desno od naslova pojedine kategorije moguće je sakriti prikaz skeniranih proizvoda u toj kategoriji, kako bi se ostavilo prostora za istovremeni prikaz kružnog grafa i legendi pripadajućih boja. Klikom na različite boje grafa prikazuju se ukupni iznosi po pripadajućim kategorijama (slika 3.9.).



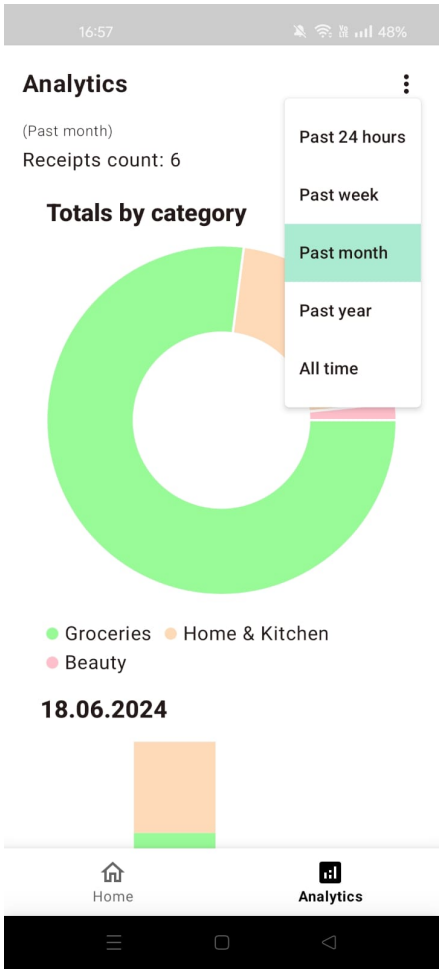
Slika 3.8. Ekran Details



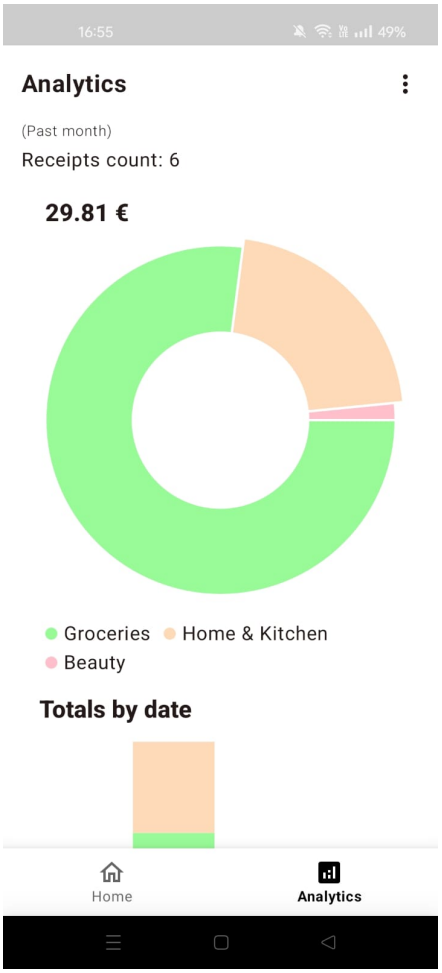
Slika 3.9. Analiza pojedinačnog računa

Klikom na tab *Analitycs* u donjem navigacijskom oknu (engl. *bottom navigation bar*) prelazi se na Analytics ekran (slika 3.10.). Na ovom ekranu prikazana je analiza potrošnje u odabranom periodu (protekli dan - *Past 24 hours*, tjedan - *Past week* , mjesec - *Past month*, godina - *Past year* i oduvijek - *All time*). Za odabrani period potrošnje prikazana je:

- Analiza potrošnje po kategorijama (slika 3.11.)
- Analiza potrošnje po datumima (slika 3.12.)
- Istaknute stavke (slika 3.13.)



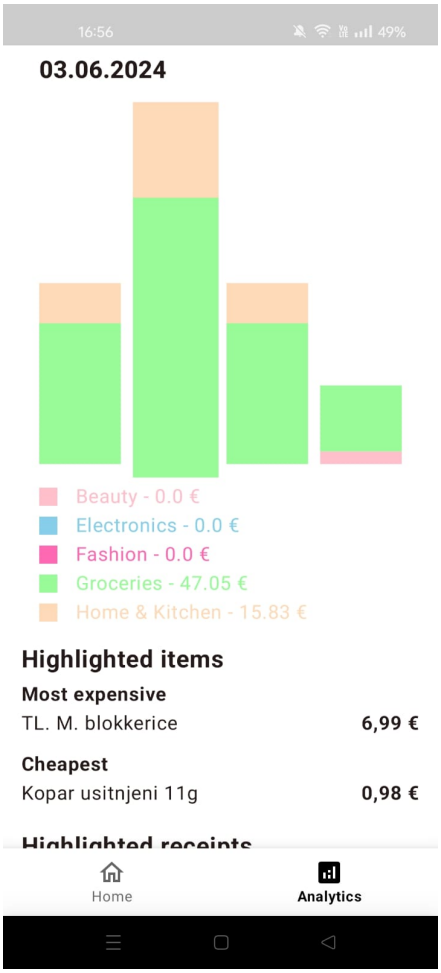
Slika 3.10. Ekran Analytics



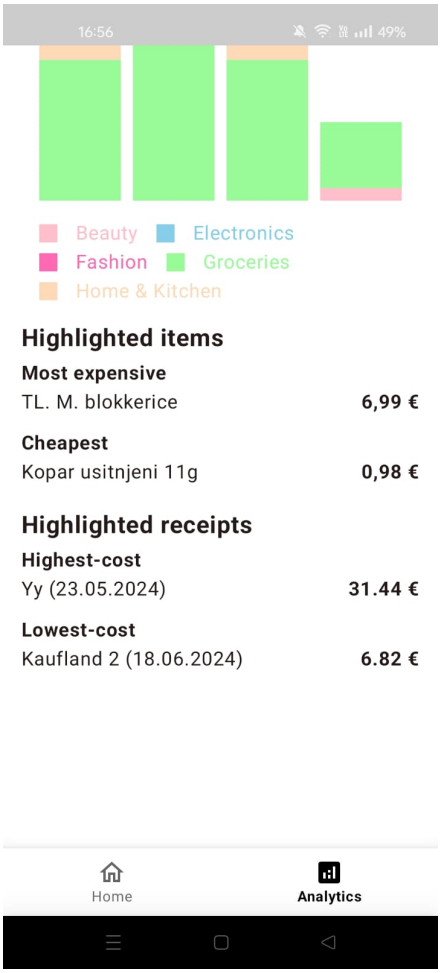
Slika 3.11. Potrošnje po kategorijama

Kod analize potrošnje po kategorijama, slično kao i na Details ekranu prikazana je ukupna potrošnja po kategorijama proizvoda u odabranom vremenskom periodu. Klikom na različite boje grafa prikazuju se ukupni iznosi po pripadajućim kategorijama.

Analiza potrošnje po datumima prikazuje stupčasti graf u kojem je svaki zabilježeni datum naznačen stupcem. Svaki stupac obojen je u omjerima potrošnje po kategorijama za taj datum, a klikom na stupac iznad grafa se prikazuje pripadajući datum, dok se ispod grafa naznačuju potrošnje po pojedinim kategorijama (slika 3.12.).



Slika 3.12. Potrošnje po datumima



Slika 3.13. Izdvojeni artikli i računi

Dio s istaknutim stavkama prikazuje najskuplje i najjeftinije kupljene artikle, kao i najviše i najniže račune za odabrani vremenski period (slika 3.13.).

3.4. Poslužiteljska strana

Poslužiteljska strana izlaže sučelje za obradu slika računa i klasifikaciju pripadajućih stavki. Sva funkcionalnost poslužitelja sažeta je u jednu krajnju točku (engl. *endpoint*), kojoj se podaci iz forme u Androidovoj aplikaciji (slika 3.6.) šalju u tijelu (engl. *body*) HTTP-ovog zahtjeva POST. Dospjela slika računa zatim se prosljeđuje transformeru Donut koji parsiranjem slike izdvaja stavke i popratne informacije o računu. Nakon inicijalne obrade, klasifikacija pojedinih stavki provodi se pomoću transformera BERT, prilagođenog zadatku klasifikacije proizvoda na hrvatskom jeziku. Ovaj pristup omogućuje detaljnu analizu potrošnje na klijentskoj strani, unapređujući funkcionalnost sustava za obradu računa. Postupak integracije navedenih modela dan je u nastavku.

3.4.1. Integracija Transformera Donut

Integracija Transformera Donut obuhvaća učitavanje modela "donut-base-finetuned-cord-v2" pomoću DonutProcessor i VisionEncoderDecoderModel klasa iz biblioteke Transformers. Inicijalna postavka modela omogućuje procesiranje slika računa, koje se prvo prilagođavaju potrebnim dimenzijama (224x224) prije obrade u Transformeru. Nakon dobivanja rezultata (skeniranih računa i pojedinačnih stavki) tekstualne sekvence dekodiraju se i čiste.

Konačno, svaka stavka računa služi kao ulaz u klasifikator BERT koji joj na temelju naziva dodjeljuje pripadajuću kategoriju. Proces integracije Transformera BERT dan je u nastavku.

3.4.2. Integracija Transformera BERT

Kako bi se Transformer BERT pripremio za zadatak klasifikacije proizvoda u kategorije, potrebno ga je fino podesiti odgovarajućim podacima. Fino podešavanje (engl. *fine tuning*) modela BERT predstavlja ključan proces u obradi prirodnog jezika koji omogućava prilagodbu prethodno naučenog jezičnog modela nekom specifičnom zadatku. Time se postiže veća točnost i generalizacija na novim podacima, a model se prilagođava specifičnim uvjetima ili domeni primjene te omogućava efikasniji proces učenja u usporedbi s učenjem modela ispočetka. Postupak finog podešavanja Transformera BERT dan je u nastavku.

Podaci

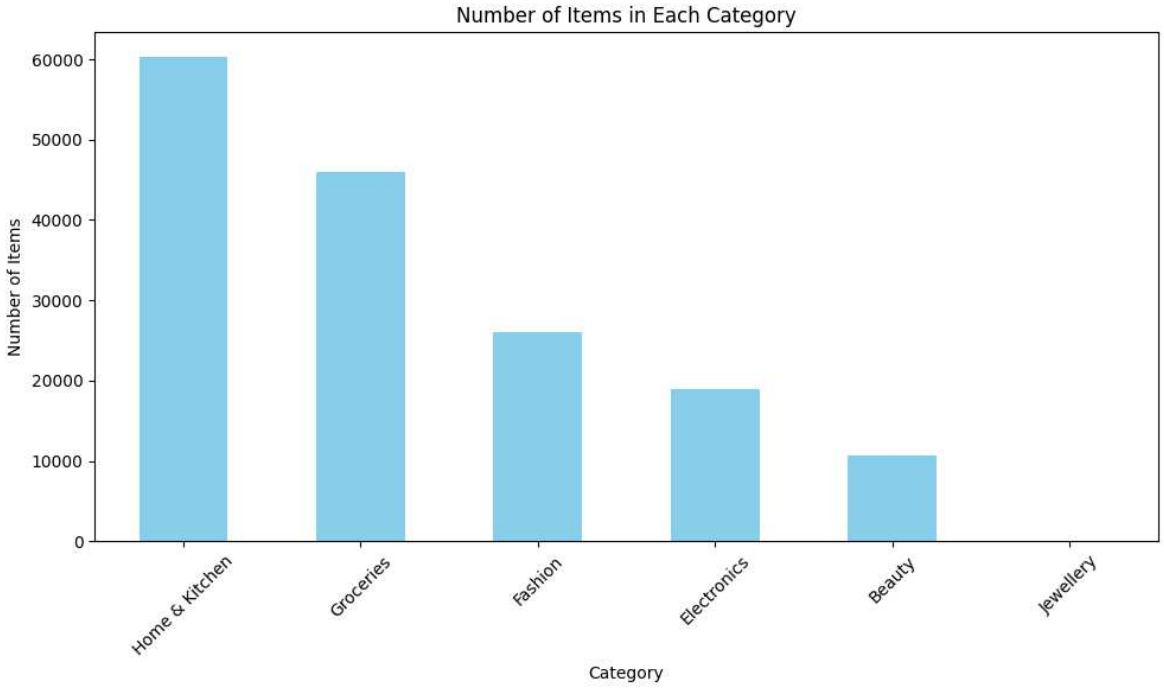
Kako bi mogao obavljati zadatak klasifikacije artikala u kategorije, Transformer BERT bilo je potrebno fino podesiti odgovarajućim podacima. Za ovaj zadatak odabran je skup podataka (engl. *dataset*) *Jio Mart Product Items* prikupljenih s web stranice JioMart. Skup podataka sadrži ukupno 158172 proizvoda i 5 stupaca (engl. *column*):

1. **category** - glavna kategorija proizvoda
2. **sub_category** - potkategorija proizvoda
3. **href** - link na proizvod
4. **items** - ime proizvoda
5. **price** - cijena proizvoda

U postupku finog podešavanja korišteni su stupci category i sub_category. Category ima 6 mogućih vrijednosti u cijelom skupu podataka, uz svaku kategoriju naznačen je i ukupan broj proizvoda te kategorije u skupu podataka:

1. **Groceries** - 46044
2. **Home & Kitchen** - 60335
3. **Fashion** - 26101
4. **Electronics** - 19022
5. **Beauty** - 10741
6. **Jewellery** - 70

Na slici 3.14. dan je grafički prikaz ovih brojki. Kako je skup podataka neujednačen, bili su potrebni određeni koraci pripreme podataka prije postupka finog učenja Transformera BERT, što je prikazano u potpoglavlju 3.4.2.



Slika 3.14. Broj proizvoda po kategorijama

Dodatan problem u promatranom skupu podataka je jezik. Imena svih proizvoda u *items* stupcu su na engleskom jeziku. Kako je zamišljeno da model radi klasifikaciju s imenima proizvoda na hrvatskom jeziku, prije postupka finog podešavanja bilo je potrebno prevesti nazive u stupcu *items* na hrvatski. U idućem potpoglavlju doskočeno je tom problemu. Ovdje treba napomenuti da je radi prilagodbe hrvatskom jeziku korišten nestandardni model BERT nazvan BERTić (engl. *A transformer language model for Bosnian, Croatian, Montenegrin and Serbian*). Model je učen u Zagrebu, na više od 8 milijardi tokena bosanskog, hrvatskog, crnogorskog i srpskog teksta. Naziv BERTić dolazi od nastavka -ić, tj. najčešćeg nastavka prezimena u zemljama gdje se ovi jezici govore.

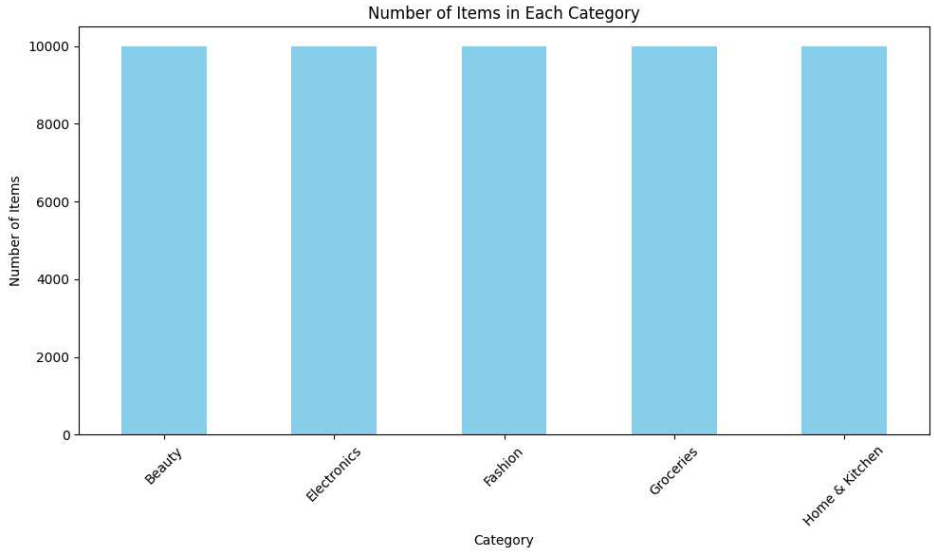
Radi dobivanja boljeg osjećaja o izgledu podataka, u nastavku je prikazano nekoliko primjera iz izvornog skupa podataka (slika 3.15.).

	category	sub_category	href	items	price
0	Groceries	Fruits & Vegetables	https://www.jiomart.com/c/groceries/fruits-veg...	Fresh Dates (Pack) (Approx 450 g - 500 g)	109.0
1	Groceries	Fruits & Vegetables	https://www.jiomart.com/c/groceries/fruits-veg...	Tender Coconut Cling Wrapped (1 pc) (Approx 90...	49.0
2	Groceries	Fruits & Vegetables	https://www.jiomart.com/c/groceries/fruits-veg...	Mosambi 1 kg	69.0
3	Groceries	Fruits & Vegetables	https://www.jiomart.com/c/groceries/fruits-veg...	Orange Imported 1 kg	125.0
4	Groceries	Fruits & Vegetables	https://www.jiomart.com/c/groceries/fruits-veg...	Banana Robusta 6 pcs (Box) (Approx 800 g - 110...	44.0

Slika 3.15. Podaci u izvornom skupu podataka

Priprema podataka

Priprema podataka sastoji se od nekoliko uzastopnih koraka. Zbog prethodno opisane neujednačenosti broja proizvoda po kategorijama (slika 3.14.) proizvodi kategorija *Jewellery* izbačeni su iz skupa podataka, zbog premalog broja primjera (70). Nadalje, kako je idući najmanji broj proizvoda u kategoriji *Beauty* (10741), broj proizvoda slučajnim je odabirom u svim kategorijama srezan na 10000. Na taj način umjetno je postignut ujednačen skup podataka koji sadrži pet kategorija: *Groceries*, *Home & Kitchen*, *Fashion*, *Electronics* i *Beauty*. Izgled skupa podataka nakon ovih koraka prikazan je na slici 3.16.



Slika 3.16. Broj proizvoda po kategorijama nakon obrade

Jednom kada je skup podataka balansiran, potrebno ga je prevesti. Prevođenje skupa podataka na hrvatski jezik odnosi se na stupac *items* koji sadrži imena pojedinih proizvoda. Za potrebe prevođenja korišten je prethodno opisani "facebook/seamless-m4t-v2-large" Transformer (potpoglavlje 2.1.6.). Prevedeni stupac *items* nazvan je *translated_items*. Konačan izgled obrađenog skupa podataka prikazan je na slici 3.17.,

	category	sub_category	href	items	price	translated_items
0	Beauty	Fragrances	https://www.jiomart.com/c/beauty/fragrances/wo...	Ajmal Viola EDP Fruity Floral Perfume And Sacr...	2250.0	Ajmal Viola EDP voćni cvjetni parfem i sveta l...
1	Fashion	Men	https://www.jiomart.com/c/fashion/men/clothing...	Ultra Marathon Cap with Mesh Panels	289.0	Ultra Marathon Cap s mrežnim panelima
2	Home & Kitchen	Stationery	https://www.jiomart.com/c/groceries/home-kitch...	Rigid Mailer Size 10 x 8 inch GSM : 430 Pack o...	795.0	Rigid Mailer veličina 10 x 8 inča GSM : 430 Pa...
3	Electronics	Home Appliances	https://www.jiomart.com/c/electronics/home-app...	LG 1.5 Ton 3 Star LS-H18VNXD Inverter Split AC	42990.0	LG 1.5 Ton 3 Star LS-H18VNXD Inverter Split AC
4	Electronics	Accessories	https://www.jiomart.com/c/electronics/accessor...	Saco Transparent Keyboard Skin For Dell Inspir...	383.0	Saco Prozirna Klavijatura Skin za Dell Inspiro...

Slika 3.17. Podaci u konačnom skupu podataka

Fino podešavanje modela

Kako bi predobrađeni skup podataka bio prilagođen za fino podešavanje Trasnformera BERT, potrebno je još nekoliko koraka. U skupu podataka ostavljeni su samo stupci bitni za BERT, tj. *translated_items* i *category*, pri čemu je stupac *translated_items* preimenovan u *product*. Nadalje, potrebno je kodirati stupac *category* kako bi se vrijednosti kategorija transformirale u numeričke. Novododani stupac s transformiranim numeričkim vrijednostima dodan je u skup podataka i nazvan *category_encoded* (slika 3.18.).

	product	category	category_encoded
0	Aamivi Multicolor Alphabet G 3D iluzija noćna ...	Home & Kitchen	4
1	Inalsa Impress 2100 Watt Indukcijska kuhinja s...	Electronics	1
2	Sandisk 128GB Cruzer Blade USB flash disk, SDC...	Electronics	1
3	Arus Multicolor Ganesha Shape Magic Night Lamp	Home & Kitchen	4
4	NEUTRON Najnoviji ljubavni kasualni kraljevski...	Fashion	2

Slika 3.18. Transformirane vrijednosti u skupu podataka

Nakon enkodiranja *category* stupca bilo je potrebno inicijalizirati klase CustomDataset i Dataloader iz biblioteke PyTorch. CustomDataset određuje kako se podaci obrađuju prije ulaska u model, dok je Dataloader zadužen za slanje podataka modelu u grupama (engl. *batches*) radi optimiziranja procesa učenja.

Klasa CustomDataset prilagođava skup podataka za fino podešavanje modela BERT. Konstruktor prima DataFrame i maksimalnu duljinu sekvence te zasebno izdvaja stupce *product* i *category_encoded*. Metoda *__len__* vraća broj zapisa u skupu podataka, a metoda *__getitem__* dohvaća proizvod i oznaku kategorije na danom indeksu, vraćajući ih kao Pythonov par (engl. *tuple*) s proizvodom i oznakom kategorije pretvorenom u *torch.Tensor*. Ova klasa omogućuje DataLoaderu da učitava podatke u grupama tijekom učenja modela, čime se osigurava kontrolirano učitavanje i obrada podataka, smanjuje opterećenje memorije i ubrzava proces učenja.

Podaci se zatim dijele na skupove za učenje, validaciju i testiranje. Prvo se određuje veličina skupa za učenje, uz očuvanje distribucije klasa, nakon čega se skup za validaciju dodatno dijeli na validacijski i testni skup uz očuvanje iste distribucije. Ovakva podjela omogućuje evaluaciju modela na neviđenim podacima. Nakon podjele, podatke se pre-

tvara u instance CustomDataset klase s maksimalnom duljinom sekvence MAX_LEN. DataLoader objekti zatim se koriste za efikasno učitavanje podataka u modele za trening i evaluaciju. Konačno, bilo je potrebno definirati model neuronske mreže koji sadrži model BERT i prilagođen je finom podešavanju u petlji učenja (engl. *training loop*) i kasnijoj klasifikaciji proizvoda.

Neuronska mreža definirana je klasom BERTClass, a uključuje model BERT, sloj Dropout za regularizaciju te linearni sloj za klasifikaciju. Metoda forward prima tokenizirani tekst na ulazu te za njega BERT generira dva izlaza, drugi izlaz (*pooled output*) prolazi kroz Dropout sloj prije nego što se proslijedi linearnom sloju.

Ulazni tekst se prije ulaska u forward metodu tokenizira korištenjem klase AutoTokenizer.from_pretrained("classla/bcsm-bertic"). Izlazi linearnog sloja koriste se za izračunavanje gubitka i procjenu točnosti predikcije modela. Kao funkcija gubitka korištena je funkcija križne entropije (engl. *cross entropy loss*).

Ovi koraci ključni su za stvaranje i pripremu prilagođenog modela BERT za fino podešavanje na specifičnom zadatku, uključujući konfiguraciju, definiciju slojeva i inicijalizaciju tokenizera potrebnog za obradu tekstualnih podataka na ulazu.

Konačno, definirana je funkciju za učenje koja koristi prethodno odvojene podatke (CustomDataset) za učenje kroz određeni broj epoha. Epoha određuje koliko puta cijeli skup podataka prolazi kroz mrežu. Svaka epoha započinje postavljanjem modela u način rada za učenje, nakon čega se prolazi kroz sve grupe podataka u skupu za učenje, tijekom čega Dataloader prenosi podatke modelu na temelju veličine serije. Za svaku seriju, model se optimira prema funkciji gubitka, koji se računa na temelju izlaza modela i stvarne kategorije proizvoda. Nakon svake epohe, model se validira na skupu za validaciju radi provjere vrijednosti gubitka. Ako se tijekom validacije postigne bolji rezultat nego prethodno, ažurira se najbolji model. Petlja učenja u konačnici vraća najbolji model, najbolju metriku i indeks najbolje epohe, osiguravajući tako korištenje modela s najboljim parametrima. Model je na skupu podataka za testiranje (10% podataka) postigao rezultate za funkciju gubitka od 0.05166 i mjeru F1 od 0.98538. Dobiveni parametri najboljeg modela spremaju se za kasniju upotrebu na poslužiteljskoj strani.

4. Rezultati i rasprava

U ovom radu je kombinacijom nekoliko Transformera (Seamless, Donut i BERTić) i mobilne aplikacije ostvarena praktična primjena više zadataka analize prirodnog jezika. Svaki od tih transformera igra važnu ulogu u procesu skeniranja slika računa, prepoznavanja stavki i klasifikacije tih stavki u odgovarajuće kategorije. U nastavku su komentirani rezultati rješenja navedenih zadataka obrade prirodnog jezika.

Prevođenje naziva proizvoda u skupu podataka

Za prevođenje naziva proizvoda u skupu podataka korišten je transformer *SeamlessM4T-Large V2*, koji pruža funkcionalnost prevođenja teksta u tekst (engl. *text-to-text translation - T2TT*) na 96 jezika, među kojima su engleski i hrvatski. Stoga je korišten bez dodatnog finog podešavanja hrvatskom jeziku. Model je predstavljen u radu *Seamless: Multilingual Expressive and Streaming Speech Translation* [8], gdje su navedene metrike koje ukazuju na njegove visoke performance u odnosu na dotadašnje vrhunske modele izravnog prevođenja.

	Flores (↑chrF)	
	X–eng *(n=95)*	eng–X *(n=95)*
NLLB-1.3B	59.3	48.2
NLLB-3.3B	60.6	49.6
SeamlessM4T-Medium	55.4	48.4
SeamlessM4T-Large	**60.8**	**50.9**
SeamlessM4T-Large v2	59.2	49.3

Slika 4.1. Evaluacija metrike chrF na skupu podataka Flores [8]

Slika 4.1. prikazuje tablicu usporedbe *SeamlessM4T-Large V2* sa ostalim SOTA mode-

lima korištenjem metrike chrF (engl. *CHaRacter-level F-score*) na skupu podataka Flores. Metrika chrF služi za procjenu strojnog prijevoda izračunavajući sličnost između izlaza strojnog i referentnog prijevoda pomoću n-grama znakova, a ne n-grama riječi. Metrike temeljene na n-gramima riječi posebno su problematične za jezike visoke morfologije. Usklađivanje nizova znakova kod chrF-a pomaže u prepoznavanju različitih oblika jedne riječi. Flores je referentni skup podataka za strojno prevođenje između engleskog i četiri jezika s malim resursima, nepalskog, sinhalskog, kmerskog i paštunskog, na temelju rečenica prevedenih s Wikipedije. Na slici 4.1. u lijevom stupcu su rezultati kod prevođenja jezika na engleski, a u desnom prevođenje s engleskog na druge jezike, gdje *n* označava ukupan broj testiranih jezika. Ovdje je vidljivo da *SeamlessM4T-Large* ima najbolje rezultate kad je u pitanju strojno prevođenje teksta u tekst, što znači da bi imalo smisla koristiti ga umjesto spomenutog modela *SeamlessM4T-Large v2*.

Obrada slika računa

Za obradu slika računa korišten je Transformer Donut. Model Donut nadmašuje obične OCR modele jer pruža cjelovito (engl. *end-to-end*) rješenje koje direktno pretvara slike dokumenata u strukturirane informacije (JSON), integrirajući razumijevanje konteksta i strukture dokumenta. Dok tradicionalni OCR modeli prvo prepoznaju tekst i zatim prolaze kroz dodatne korake obrade, Donut koristi napredne tehnike dubokog učenja za bolje prepoznavanje i ekstrakciju ključnih informacija, prilagođavajući se kompleksnim strukturama i višejezičnim dokumentima. Zbog toga postiže bolju točnost i preciznost, posebno za specifične zadatke poput skeniranja maloprodajnih računa. Slični modeli koji koriste OCR uključuju Tesseract, Google Cloud Vision OCR, Amazon Textract, i ABBYY FineReader, koji se fokusiraju na prepoznavanje teksta iz slika, ali ne integriraju nužno napredno razumijevanje dokumenta kao Donut. Model je predstavljen u radu *OCR-free Document Understanding Transformer* [7], u kojem su navedene i metrike koje ga uspoređuju s ostalim SOTA modelima u sklopu različitih zadataka obrade prirodnog jezika.

Na slici 4.2. prikazane su izmjerene metrike klasifikacije dokumenata na skupu podataka RVL-CDIP (engl. *Ryerson Vision Lab Complex Document Information Processing*). Skup podataka sastoji se od slika skeniranih dokumenata koji pripadaju 16 klasa kao što su pismo, obrazac, e-mail, životopis, dopis itd. Sadrži 320 000 slika za učenje, 40 000

	OCR	#Params	Time (ms)	Accuracy (%)
BERT	✓	110M + $\alpha^{\dagger}$	1392	89.81
RoBERTa	✓	125M + $\alpha^{\dagger}$	1392	90.06
LayoutLM	✓	113M + $\alpha^{\dagger}$	1396	91.78
LayoutLM (w/ image)	✓	160M + $\alpha^{\dagger}$	1426	94.42
LayoutLMv2	✓	200M + $\alpha^{\dagger}$	1489	95.25
Donut (Proposed)		143M	**752**	**95.30**

Slika 4.2. Metrike za klasifikaciju dokumenata na skupu podataka RVL-CDIP [7]

slika za validaciju i 40 000 slika za testiranje. Slike karakterizira niska kvaliteta, šum i niska rezolucija. Za svaki model naznačeno je koristi li standardnu tehnologiju OCR, broj parametara, vrijeme potrebno za klasifikaciju i postotak točnosti. Vidljivo je da Donut postiže najveću točnost uz najmanju brzinu obrade, što ga uz strukturiran izlaz čini prikladnim izborom za korištenje u aplikaciji u kojoj se ipak traži nešto brže vrijeme odziva zbog pružanja boljeg korisničkog iskustva.

			CORD [45]			Ticket [12]			Business Card			Receipt		
	OCR	#Params	Time (s)	F1	Acc.	Time (s)	F1	Acc.	Time (s)	F1	Acc.	Time (s)	F1	Acc.
BERT* [22]	✓	$86^{\dagger}_{M} + \alpha^{\ddagger}$	1.6	73.0	65.5	1.7	74.3	82.4	1.5	40.8	72.1	2.5	70.3	54.1
BROS [18]	✓	$86^{\dagger}_{M} + \alpha^{\ddagger}$	1.7	74.7	70.0									
LayoutLM [65]	✓	$89^{\dagger}_{M} + \alpha^{\ddagger}$	1.7	78.4	81.3									
LayoutLMv2* [64,66]	✓	$179^{\dagger}_{M} + \alpha^{\ddagger}$	1.7	78.9	82.4	1.8	87.2	90.1	1.6	52.2	83.0	2.6	72.9	78.0
Donut		$143^{\dagger}_{M}$	**1.2**	**84.1**	**90.9**	**0.6**	**94.1**	**98.7**	**1.4**	**57.8**	**84.4**	**1.9**	**78.6**	**88.6**
SPADE* [25]	✓	$93^{\dagger}_{M} + \alpha^{\ddagger}$	4.0	74.0	75.8	4.5	14.9	29.4	4.3	32.3	51.3	7.3	64.1	53.2
WYVERN* [21]	✓	$106^{\dagger}_{M} + \alpha^{\ddagger}$	1.2	43.3	46.9	1.5	41.8	54.8	1.7	29.9	51.5	3.4	71.5	82.9

Slika 4.3. Metrike za izdvajanje informacija s dokumenata [7]

Na slici 4.3. prikazani su rezultati metrika (vrijeme, F1, točnost) na četiri različita zadatka izdvajanja informacija iz dokumenata. Prva grupa koristi konvencionalni pristup ekstrakciji, slijedeći ustaljene konvencije. OCR izdvaja tekstove i pripadajuće okvire (engl. *bounding boxes*) iz slika, koji se zatim sortiraju u modulu za serijalizaciju. Ispitana su i četiri opća modela VDU-a (engl. *Visual Document Understanding*): BERT, BROS, LayoutLM i LayoutLMv2. Ispitani su i novi modeli za ekstrakciju informacija: SPADE i WYVERN. Model Donut pokazuje najbolje rezultate na svim domenama, uključujući javne i privatne skupove podataka, ističući se stabilnim performansama bez obzira na veličinu skupa podataka i složenost zadataka.

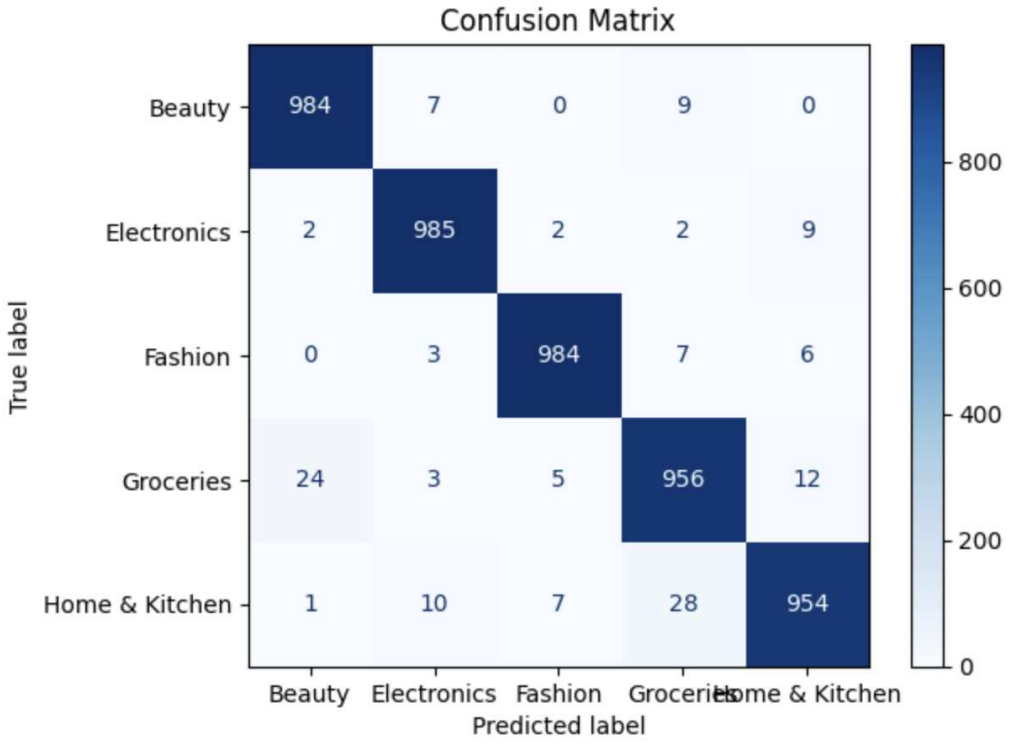
Klasifikacija stavki računa

Za klasifikaciju stavki računa korišten je model BERTić, koji je dodatno fino podešen skupom podataka *Jio Mart Product Items*. Prije početka procesa finog podešavanja, skup podataka preveden je na hrvatski korištenjem modela Seamless. U radu *BERTić - The Transformer Language Model for Bosnian, Croatian, Montenegrin and Serbian* [9] prikazana je usporedba BERTić-a i nekih sličnih BERT modela na različitim jezicima i skupovima podataka. Na kraju procesa finog podešavanja model je evaluiran na testnom skupu podataka (10% podataka) i za zadatak klasifikacije artikala u pet kategorija dobiveni su rezultati prikazani na slici 4.4. Model postiže preciznost između 0.95 i 0.99, odziv između 0.95 i 0.98, i F1-score između 0.96 i 0.98 za sve kategorije. Ukupna točnost modela je 97%, dok su i macro i weighted prosjeci za preciznost, odziv i F1-score također 0.97. Ovi rezultati ukazuju na vrlo dobru sposobnost modela da ispravno klasificira artikle u odgovarajuće kategorije.

	precision	recall	f1-score	support
Beauty	0.97	0.98	0.98	1000
Electronics	0.98	0.98	0.98	1000
Fashion	0.99	0.98	0.98	1000
Groceries	0.95	0.96	0.96	1000
Home & Kitchen	0.97	0.95	0.96	1000
accuracy			0.97	5000
macro avg	0.97	0.97	0.97	5000
weighted avg	0.97	0.97	0.97	5000

Slika 4.4. Metrike modela BERTić na testnom skupu podataka

Pripadajuća matrica konfuzije prikazana je na slici 4.5., gdje je vidljivo da su najčešće zamjenjivane kategorije proizvoda *Beauty* i *Groceries*, te *Groceries* i *Home & Kitchen*.



Slika 4.5. Matrica konfuzije na testnom skupu podataka

5. Zaključak

Korišteni fino podešeni model BERTić postiže točnost od 97% na skupu podataka za testiranje u zadatku klasifikacije naziva artikala u pet kategorija: *Beauty, Electronics, Fashion, Groceries, Home & Kitchen*. Mobilna aplikacija koja ga koristi ispunjava postavljene funkcionalne zahtjeve. Aplikacija je osmišljena kao demonstracija praktične primjene modela transformera. Unaprijeđena verzija ove aplikacije koja bi mogla s većom točnošću klasificirati više od 5 kategorija proizvoda imala bi potencijal za praktičnu uporabu. Radom na izradi ovakvog sustava stečena su iskustva u izgradnji aplikacija za Android, kao i praktična primjena transformera za rješavanje zadataka obrade prirodnog jezika.

Literatura

[1] H. Lane, C. Howard, i H. M. Hapke, *Natural Language Processing in Action*. Manning Publications, 2019., all rights reserved.

[2] K. W. Church i L. F. Rau, "Commercial applications of natural language processing", *Commun. ACM*, sv. 38, br. 11, str. 71–79, nov 1995. https://doi.org/10.1145/219717.219778

[3] P. M. Nadkarni, L. Ohno-Machado, i W. W. Chapman, "Natural language processing: an introduction", *Journal of the American Medical Informatics Association*, sv. 18, br. 5, str. 544–551, 2011.

[4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, i I. Polosukhin, "Attention is all you need", 2023. [Mrežno]. Adresa: https://arxiv.org/abs/1706.03762

[5] L. Tunstall, L. von Werra, i T. Wolf, *Natural Language Processing with Transformers, Revised Edition*, revised izd. O'Reilly Media, 2022., all rights reserved.

[6] J. Devlin, M.-W. Chang, K. Lee, i K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding", 2019. [Mrežno]. Adresa: https://arxiv.org/abs/1810.04805

[7] G. Kim, T. Hong, M. Yim, J. Nam, J. Park, J. Yim, W. Hwang, S. Yun, D. Han, i S. Park, "Ocr-free document understanding transformer", 2022. [Mrežno]. Adresa: https://arxiv.org/abs/2111.15664

[8] S. Communication, L. Barrault, Y.-A. Chung, M. C. Meglioli, D. Dale, N. Dong, M. Duppenthaler, P.-A. Duquenne, B. Ellis, H. Elsahar, J. Haaheim, J. Hoffman,

M.-J. Hwang, H. Inaguma, C. Klaiber, I. Kulikov, P. Li, D. Licht, J. Maillard, R. Mavlyutov, A. Rakotoarison, K. R. Sadagopan, A. Ramakrishnan, T. Tran, G. Wenzek, Y. Yang, E. Ye, I. Evtimov, P. Fernandez, C. Gao, P. Hansanti, E. Kalbassi, A. Kallet, A. Kozhevnikov, G. M. Gonzalez, R. S. Roman, C. Touret, C. Wong, C. Wood, B. Yu, P. Andrews, C. Balioglu, P.-J. Chen, M. R. Costa-jussà, M. Elbayad, H. Gong, F. Guzmán, K. Heffernan, S. Jain, J. Kao, A. Lee, X. Ma, A. Mourachko, B. Peloquin, J. Pino, S. Popuri, C. Ropers, S. Saleem, H. Schwenk, A. Sun, P. Tomasello, C. Wang, J. Wang, S. Wang, i M. Williamson, "Seamless: Multilingual expressive and streaming speech translation", 2023. [Mrežno]. Adresa: https://arxiv.org/abs/2312.05187

[9] N. Ljubešić i D. Lauc, "BERTić - the transformer language model for Bosnian, Croatian, Montenegrin and Serbian", u *Proceedings of the 8th Workshop on Balto-Slavic Natural Language Processing*, B. Babych, O. Kanishcheva, P. Nakov, J. Piskorski, L. Pivovarova, V. Starko, J. Steinberger, R. Yangarber, M. Marcińczuk, S. Pollak, P. Přibáň, i M. Robnik-Šikonja, Ur. Kiyv, Ukraine: Association for Computational Linguistics, travanj 2021., str. 37–42. [Mrežno]. Adresa: https://aclanthology.org/2021.bsnlp-1.5

Sažetak

Mobilna aplikacija za analizu osobne potrošnje na temelju slika maloprodajnih računa

Petar Miličević

Ovim radom demonstrirana je praktična primjena tri modela transformera (Seamless, Donut i BERTić) u zadacima obrade prirodnog jezika. Prikazana je upotreba modela za obradu slika računa i klasifikaciju pojedinih stavki u jednu od pet kategorija, s klasifikacijskom točnošću od oko 97% koristeći model BERTić. Izrađena je mobilna aplikacija za operacijski sustav Android koja koristi izgrađene modele.

Ključne riječi: Android; NLP; Seamless; Donut; BERT

Abstract

Mobile Application for Analyzing Personal Spending Based on Images of Retail Receipts

Petar Miličević

This paper demonstrates the practical application of three transformer models (Seamless, Donut, and BERTić) in natural language processing tasks. It shows the use of these models for processing receipt images and classifying individual items into one of five categories, achieving a classification accuracy of approximately 97% using the BERTić model. A mobile application for the Android operating system, which utilizes the developed models, has been created.

Keywords: Android; NLP; Seamless; Donut; BERT