Univerza v Ljubljani
Fakulteta za računalništvo in informatiko

Miha Bombek

Prilagajanje vnaprej naučenega modela BERT slovenskim klasifikacijskim nalogam

diplomsko delo

univerzitetni študijski program
prve stopnje
računalništvo in informatika

Mentor: prof. dr. Marko Robnik Šikonja

Ljubljana, 2021

Besedilo je oblikovano z urejevalnikom besedil LaTeX.

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Tematika naloge:

Veliki, vnaprej naučeni modeli BERT in njihove izvedenke so trenutno najuspešnejši način za reševanje številnih nalog za obdelavo naravnega jezika. Tipično jih za konkretne klasifikacijske naloge prilagodimo tako, da njihov zadnji sloj nadomestimo z novim, ki napoveduje oznake dane naloge, nato pa celotno mrežo doučimo na podatkih konkretne naloge. Mogoči so tudi drugi načini, npr. metoda združevanja adapterjev v nevronsko mrežo dodaja dodatne plasti in kombinira prilagoditve za različne sorodne naloge. Preučite različne načine prilagajanja modelov BERT klasifikacijskim nalogam v slovenščini. Osredotočite se predvsem na metode združevanja adapterjev. Razvite metode empirično ovrednotite.

Zahvaljujem se mentorju prof. dr. Marku Robniku Šikonji za pomoč in usmerjanje pri izdelavi diplomskega dela.

Kazalo

Seznam uporabljenih kratic

kratica	angleško	slovensko
NLP	natural language processing	obdelava naravnega jezika
GPU	graphics processing unit	grafična procesna enota
RNN	recurrent neural network	rekurenčna nevronska mreža
LSTM	long short-term memory neural network	nevronska mreža z dolgim kratkoročnim spominom
UPOS	universal part-of-speech	univerzalna besedna vrsta
NER	named entity recognition	prepoznavanje imenskih entitet
BERT	bidirectional encoder representations from transformers	predstavitev z dvosmernimi enkoderji arhitekture transformer

Povzetek

Naslov: Prilagajanje vnaprej naučenega modela BERT slovenskim klasifikacijskim nalogam

Avtor: Miha Bombek

Za reševanje nalog na področju obdelave besedil so trenutno najbolj uspešni modeli arhitekture transformer, kot je vnaprej naučen model BERT. Pri prilagajanju predhodno naučenega modela za specifično nalogo ponavadi prilagodimo vse parametre modela. V delu preučujemo metode prilagajanja modela BERT, ki prilagodijo le manjši del parametrov. Analiziramo rezultate pri reševanju klasifikacijskih nalog v slovenščini. Prilagajamo večjezikovna modela CroSloEngual BERT in mBERT na nalogah prepoznavanja imenskih entitet in označevanja univerzalnih besednih vrst. Uporabimo štiri različne metode prilagajanja: prilagajanje celotnega modela, prilagajanje le zadnje plasti, prilagajanje z adapterjem in prilagajanje z metodo združevanja adapterjev. Pokažemo, da prilagajanje z adapterjem, kljub majhnemu številu prilagojenih parametrov, dosega dobre rezultate in da lahko z združevanjem adapterjev dosežemo tudi boljše rezultate kot pri prilagajanju celotnega modela. Ugotovimo, da je metoda združevanja adapterjev koristnejša pri klasifikacijskih nalogah višjega nivoja. Slabost te metode je čas učenja, saj je celoten postopek združevanja adapterjev lahko dolgotrajen.

Ključne besede: strojno učenje, obdelava naravnega jezika, model BERT, klasifikacijska naloga, prilagajanje z združevanjem adapterjev.

Abstract

Title: Fine-tuning pretrained BERT model for Slovene classification tasks

Author: Miha Bombek

Transformer based models, such as pretrained BERT model, are currently the most successful approach to text processing tasks. When tuning BERT for a specific task, we usually fine-tune all the model's parameters. We investigate methods for fine-tuning BERT models, which fine-tune only a fraction of parameters for a specific task. We analyze results on Slovene classification tasks. We fine-tune multilingual models CroSloEngual BERT and mBERT on named entity recognition and UPOS tagging. We compare four fine-tuning methods: full model fine-tuning, tuning only the classification head, adapter tuning, and AdapterFusion fine-tuning. We show that adapter tuning achieves good results, despite the small number of tuned parameters, and that AdapterFusion tuning can achieve better results than full model fine-tuning. We discover that AdapterFusion tuning is more beneficial when solving higher level classification tasks. The downside of this method is that it is time consuming.

Keywords: machine learning, natural language processing, BERT model, classification task, AdapterFusion fine-tuning.

Poglavje 1

Uvod

Obdelava naravnega jezika (angleško *natural language processing*, NLP) je področje umetne inteligence, ki se ukvarja s strojnim razumevanjem in procesiranjem naravnega jezika. To področje je v zadnjih letih močno napredovalo zaradi razvoja številnih novih tehnologij. Pri obdelavi besedil je bila nedavno razvita tehnologija predstavitve z dvosmernimi enkoderji arhitekture transformer (angleško *bidirectional encoder representations from transformers*, BERT) [3, 4], ki se je izkazala za zelo učinkovito in je spremenila način predstavitve jezika. Tehnologija se hitro razvija.

Veliko dela je namenjenega razvoju prilagajanja modela BERT specifični NLP-nalogi. Poznamo več metod prilagajanja. Dolgo je za najboljšo metodo veljala metoda prilagajanja celotnega modela. Novejši pristop k prilagajanju modela so adapterji [10]. Ti bolje izkoristijo prilagojene parametre, ki jih je manj kot v celotnem modelu BERT. Modeli z adapterji dosegajo primerljive rezultate tistim, ki jih dosega prilagojen celotni model. Ta metoda poudarja modularnost prilagajanja in prenosljivost znanja.

Nedavno je bila razvita tehnologija združevanja adapterjev [14], ki uporablja znanje adapterjev, prilagojenih za različne NLP naloge. Modeli, prilagojeni s to metodo, dostikrat dosegajo boljše rezultate kot modeli, prilagojeni v celoti.

V nalogi nas zanima, kako učinkovite so različne metode pri prilagajanju

modela BERT slovenskim klasifikacijskim nalogam. Da bi izboljšali obstoječe modele za obdelavo slovenskih besedil, model prilagajamo z različnimi metodami in primerjamo rezultate. Z metodo združevanja adapterjev poskusimo izboljšati rezultat, dobljen s celotno prilagojenim modelom. Adapterje prilagodimo več različnim slovenskim klasifikacijskim nalogam in jih združimo z metodo združevanja adapterjev (AdapterFusion). Modele prilagajamo za klasifikacijski nalogi prepoznavanje imenskih entitet (angleško *named entity recognition*, NER) in označevanje univerzalnih besednih vrst (angleško *universal part-of-speech*, UPOS). Rezultate primerjamo z drugače prilagojenimi modeli.

Delo obsega šest poglavij. V 2. poglavju predstavimo področje obdelave besedil in podrobneje predstavimo obdelavo besedil z nevronskimi mrežami. V 3. poglavju predstavimo model BERT in podrobnosti o njegovem delovanju in opišemo različne metode prilagajanja modela za specifično nalogo. V 4. poglavju razložimo načrt evalvacije različnih metod prilagajanja modela BERT slovenskim klasifikacijskim nalogam. V 5. poglavju analiziramo dobljene rezultate. V 6. poglavju povzamemo narejeno in predstavimo ideje za izboljšave.

Poglavje 2

Metode obdelave besedil

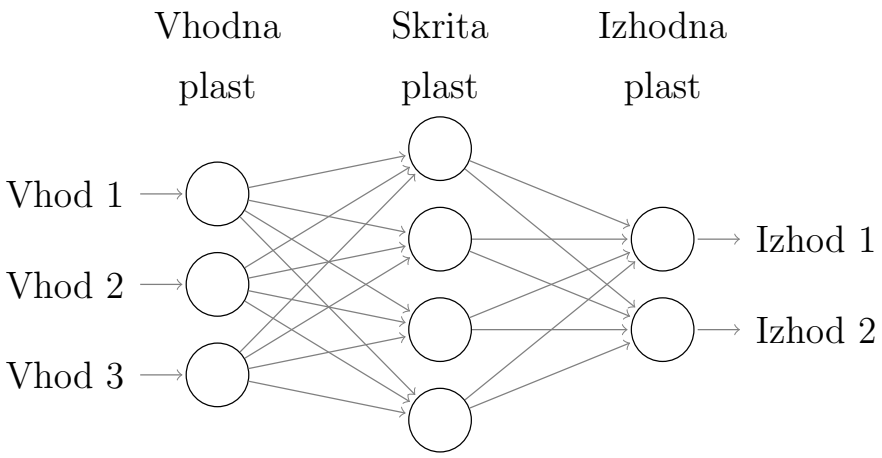
Področje NLP se ukvarja s strojnim razumevanjem in procesiranjem naravnega jezika v več možnih oblikah, kot so govor in pisana besedila. Podpodročje NLP, imenovano obdelava besedil, se ukvarja z razumevanjem in procesiranjem besedil v pisni obliki.

Področje obdelave besedil že od devetdesetih let prejšnjega stoletja temelji na strojnem učenju. Težava te metode je sprva bilo pomanjkanje ustreznih učnih podatkov, zato je bilo veliko razvoja namenjenega izboljševanju delovanja modelov z omejeno količino podatkov. Zaradi interneta se je konec devetdesetih hitro povečevala količina dostopnega besedila, zato se je začel razvoj modelov, ki se učijo nenadzorovano na neoznačenem besedilu. Ti modeli nimajo ročno zgrajenih slovarjev, ampak si slovar ustvarijo sami. Besede predstavimo kot vektorje realnih števil, ki so učinkoviti pri obdelavi. Zaradi ogromne količine učnih podatkov lahko ti modeli prinesejo dobre rezultate. V zadnjem desetletju je najbolj priljubljena in učinkovita uporaba globokih nevronskih mrež.

V tem poglavju opišemo uporabo nevronskih mrež za obdelavo besedil. Predstavimo delovanje rekurenčnih nevronskih mrež (angleško *reccurent neural nework*, RNN), nevronskih mrež z dolgim kratkoročnim spominom (angleško *long short-term memory*, LSTM) in arhitekturo transformerjev. Govorimo tudi o pomembnosti smeri obdelave besedil.

2.1 Nevronske mreže

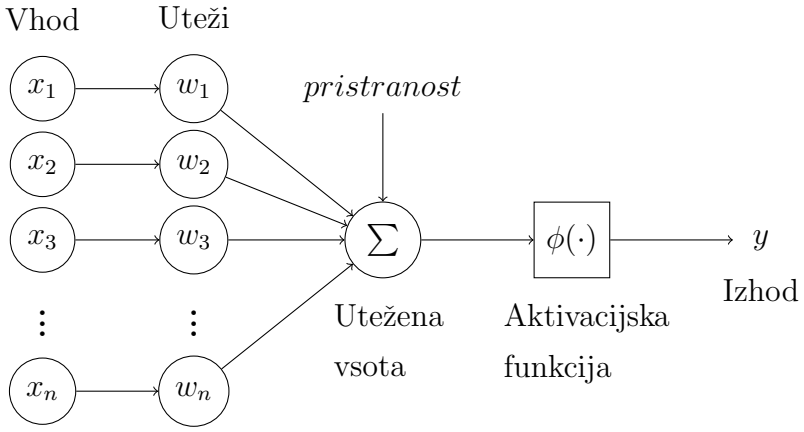
Nevronska mreža je računski model, katerega delovanje je podobno delovanju človeških možganov. Sestavljena je iz procesnih enot, imenovanih nevroni. V mreži so nevroni razporejeni v plasti in med seboj povezani. Vsak nevron je povezan le z nevroni prejšnje plasti, od koder dobiva vhodne podatke, in nevroni naslednje plasti, kamor pošilja svoje izhodne podatke. Na sliki 2.1 je prikazana shema primera nevronske mreže s tremi plastmi. Podatki v nevronski mreži potujejo v smeri povezav naprej.



Slika 2.1: Shema nevronske mreže.

Vsaka vhodna povezava v nevronu ima svojo utež. Ko med delovanjem mreže posamezen nevron dobi vhodne podatke, najprej izračuna njihovo uteženo vsoto. To vsoto pošlje skozi aktivacijsko funkcijo, ta pa jo nelinearno preslika. Ta preslikava je izhod nevrona. Shemo posameznega nevrona vidimo na sliki 2.2.

Učenje nevronske mreže je nadzorovano in poteka nad označenimi učnimi podatki. Podatki potujejo čez mrežo, ta pa vrne nek izhod. Če izhod ni pravilen, se izračuna napaka. Na podlagi te napake se oceni gradient funkcije napake, s katerim se prilagodi uteži v mreži, da bo napaka manjša. Informacija o gradientu se „širi nazaj“ (angleško *backpropagation*) po mreži proti vhodnim



Slika 2.2: Shema nevrona.

plastem. Za predhodne nevrone se gradient izračuna po verižnem pravilu odvajanja, ki se za funkciji f in g zapiše v naslednji obliki: $(f \circ g)' = (f' \circ g) \cdot g'$. Med „širjenjem nazaj“ se v nevronih uteži prilagodijo tako, da zmanjšajo napako. To se ponavlja, dokler niso uteži v nevronih nastavljene tako, da mreža vrača majhno napako.

Da lahko besedilo obdelujemo z nevronskimi mrežami, moramo besede predstaviti kot vektorje realnih števil. To dosežemo z uporabo metode vektorskih vložitev (angleško *word embeddings*). Ti vektorji pogosto nosijo informacije o besedi, kot so kontekst besede, položaj besede v stavku, odvisnost besede od drugih besed v stavku itd. Pri obdelavi besedil imamo opravka s procesiranjem zaporedja podatkov, zato se pogosto uporabljajo rekurenčne nevronske mreže (RNN).

RNN je nevronska mreža, katere nevroni imajo poleg povezav naprej še povratne povezave. Vsak nevron poleg vhodnih podatkov prejme še izhode, dobljene v prejšnjem časovnem koraku. Pri obdelavi besedila je vhod v obliki zaporedja besed. Nevroni imajo tako pri obdelavi besede tudi informacije o obdelavi prejšnjih besed. Tako ima RNN ob vsakem vhodu informacije o prejšnjih vhodih istega zaporedja in zato rečemo, da ima spomin, kar omogoča obdelavo zaporedij podatkov.

RNN imajo kratkoročni spomin. Ker delajo z zaporedji podatkov, morajo med obdelovanjem zaporedja hraniti informacije o že obdelanem delu zaporedja. Če je to zaporedje dolgo, se čez čas izgubijo informacije o bolj zgodnjih delih zaporedja, saj jih povozijo informacije o bližnjih delih. To se zgodi zaradi zgradbe celice, saj ta ne nadzoruje, koliko informacij se obdrži iz prejšnjih časovnih korakov.

Pogosta težava, ki se pojavi ob učenju velikih RNN, je problem izginjajočega in naraščujočega gradienta [8]. Ker je gradient pri širjenju nazaj ob vsakem prehodu v naslednjo plast pomnožen z odvodom aktivacijske funkcije in ker imajo te funkcije pri velikih ali malih vhodih odvod zelo velik ali blizu 0, se gradient ob potovanju nazaj skozi plasti zelo poveča ali bliža 0. Če je teh plasti veliko, je lahko gradient na začetnih plasteh tako velik, da zaduši vse ostale vhode, ali tako majhen, da sploh ne vpliva na uteži. Ker so začetne plasti dostikrat ključne za prepoznavanje bistvenih podatkov iz vhoda, lahko to povzroča nenatančnost modela.

Do nedavnega je v obdelovanju besedil prevladovala uporaba nevronskih mrež LSTM. Leta 2017 so bile predstavljene nevronske mreže arhitekture transformer [16]. Zaradi številnih prednosti se ta tehnologija hitro razvija.

2.1.1 Mreže LSTM

Nevronske mreže LSTM [9] so bile predstavljene leta 1997. Razvite so bile z namenom, da odpravijo kratkoročni spomin RNN in zmanjšajo problem izginjajočega in naraščujočega gradienta.

Mreže LSTM so vrsta RNN. Sestavljene so iz bolj kompleksnih nevronov, imenovanih celice LSTM. Te celice poleg skritega stanja na izhod pošiljajo še informacije o „stanju celice“ (angleško *cell state*). Vsaka celica ob prejemu teh podatkov oceni njihovo pomembnost. Če so podatki stanja celice pomembni, jih bo trenutna celica le malo prilagodila, preden jih pošlje naprej. Če oceni, da podatki niso pomembni, bo nekaj teh podatkov pozabila in naprej poslala prilagojeno stanje celice. Tako se lahko pomembne informacije v mreži obdržijo dalj časa in zmanjšamo problem kratkoročnega spomina

RNN. Tudi gradient se nazaj proti vhodnim plastem širi bolj nemoteno in tako zmanjšamo problem izginjajočega in naraščujočega gradienta.

Ker so mreže LSTM kompleksnejše, so počasne pri učenju. Še eno slabost predstavlja nezmožnost paralelizacije operacij. Mreža na vhod sprejema besede v zaporedju in ne more obravnavati vseh istočasno.

2.1.2 Transformerji

Leta 2014 je bil za namene strojnega prevajanja naravnih jezikov z RNN razvit koncept „pozornosti“ (angleško *Attention*) [1, 12]. Deluje tako, da model vsaki vhodni besedi priredi še dodaten vektor dolžine celotnega stavka. V tem vektorju oceni, kako povezana je obravnavana beseda z drugimi besedami v stavku. To modelu omogoča, da se ob obdelavi besede bolj osredotoči na besede, ki so z njo bolj povezane. Tako model za posamezno besedo dobi več informacij o njenem kontekstu. To je močno izboljšalo učinkovitost modelov za strojno prevajanje naravnega jezika.

Transformer [16] je model, originalno namenjen strojnemu prevajanju, ki za učenje in obdelavo besedil uporablja pozornost. Zgrajen je iz bloka kodirnikov, bloka dekodirnikov in povezav med njima. Med prevajanjem naravnega jezika blok kodirnikov na vhodu dobi zaporedje besed izvornega jezika. Zgradba kodirnikov transformerju omogoči paralelizacijo obdelovanja. Tako lahko več besed obdeluje istočasno. Za vse besede naenkrat generira njihove vektorske predstavitve, ki opišejo njihov kontekst v stavku. Te predstavitve pošlje v blok dekodirnikov. Ta generira besede prevoda eno za drugo. Za generiranje posamezne besede, poleg dobljenih vektorskih predstavitev, upošteva tudi predhodno generirane besede prevoda.

Ta arhitektura nalogo strojnega prevajanja razdeli na dve nalogi. Blok kodirnikov se za ustvarjanje dobrih vektorskih predstavitev nauči različnih kontekstov in povezav med besedami. Tako dobi dobro predstavitev izvornega jezika in njegovih pravil. Blok dekodirnikov pa se za generiranje dobrih prevodov nauči preslikav besed iz izvornega v ciljni jezik. Ti dve komponenti lahko ločimo in ju uporabimo posebej. Na osnovi bloka kodirnikov

arhitekrure transformer je zgrajen model BERT [3].

2.2 Smer obdelave besedil

Pri učenju konteksta besed v stavku ima pomembno vlogo smer obdelave besedila. Beseda je v stavku povezana s svojimi predhodniki in nasledniki. Za določanje konteksta besede potrebujemo informacije o vseh besedah v stavku, vendar jih nimamo, ko besedilo obdelujemo enosmerno.

Nevronske mreže LSTM besedilo obdelujejo enosmerno, z leve proti desni ali obratno. To pomeni, da ima model ob obdelovanju neke besede le informacije o njenih predhodnikih ali naslednikih, odvisno od smeri obdelave. Zato ti modeli ne morejo biti tako natančni pri določanju konteksta besede v stavku. BiLSTM je različica mreže LSTM, ki besedilo obdeluje iz obeh smeri. Pri tem ne gre za pravo, istočasno dvosmernost, saj model najprej besedilo obdela iz ene smeri, nato pa še iz druge, na koncu pa dobljene vektorje stakne.

Pristop dvosmerne obdelave besedil se je okrepil leta 2018 z razvojem modela BERT [3]. Ker ta model uporablja transformerje, je zmožen zaradi paralelizacije več besed obdelovati istočasno. To mu omogoča natančnejše določanje konteksta besede, saj vidi tako njene predhodnike kot naslednike. To vodi do boljše predstavitve besed. Ta dvosmernost je ena večjih prednosti modela BERT, ki ga podrobneje opišemo v tretjem poglavju.

Poglavje 3

Model BERT

Leta 2018 je ekipa Google AI razvila nov pristop za predstavitev besedil, imenovan BERT [3]. Ta za obdelovanje besedil uporablja arhitekturo transformerjev. Opišemo ga lahko kot blok več kodirnikov.

Učenje modela BERT je sestavljeno iz dveh stopenj: predhodno učenje in fino prilagajanje. V prvi stopnji se model nauči predstavitve jezika. V drugi stopnji model prilagodimo, da se nauči opravljati specifično nalogo. Zaradi paralelizacije, ki jo omogoča arhitektura transformer, je BERT prvi resnično dvosmerni model, saj lahko obdeluje vse besede hkrati ter se bolje nauči njihovih odvisnosti.

V nadaljevanju predstavimo postopek predhodnega učenja in prilagajanja modela BERT. Podrobneje opišemo več metod prilagajanja modela BERT specifičnim NLP-nalogam.

3.1 Predhodno učenje

Za predhodno učenje model BERT uporablja velike in neoznačene korpuse besedil, kar je koristno zaradi pomanjkanja označenih učnih podatkov. Model se uči nenadzorovano na dveh različnih nalogah hkrati.

Prva naloga je učenje maskirnega jezikovnega modela, kjer so iz stavka vzete oz. „zamaskirane“ nekatere besede. Primer takega vhoda je prikazan

na sliki 3.1. Na podlagi stavka, poskuša BERT predvideti katere so zamaskirane besede. S tem se ugotovi kontekst zamaskiranih besed v različnih stavkih in te besede model po učenju bolje predstavi. Tu se pokaže premoč dvosmerne obdelave nad enosmerno, saj dvosmerna besedo vidi v celotnem kontekstu stavka. Model BERT stavke „zamaskira“ enkrat med predobdelavo podatkov. Med vsako epoho učenja ima model na vhodnih podatkih „zamaskirane“ iste besede.

Vhod: `Šel sem v [MASK]`$_1$`. Kupil sem štruco [MASK]`$_2$`.`
Oznake: `[MASK]`$_1$ `= trgovino; [MASK]`$_2$ `= kruha`

Slika 3.1: Možen vhod za učenje maskirnega jezikovnega modela.

Druga naloga pri učenju modela BERT je predvidevanje naslednjega stavka. Ker se model uči na neoznačenih besedilih, opazuje zaporedja stavkov in se uči smiselnih povezav med njimi. V nalogi sta dana dva stavka, model pa mora določiti, ali drugi stavek sledi prvemu ali ne. Na sliki 3.2 sta prikazana primera vhodov za nalogo predvidevanja naslednjega stavka.

Stavek A: `Šel sem v trgovino.`
Stavek B: `Kupil sem štruco kruha.`
Oznaka: `IsNextSentence`

Stavek A: `Šel sem v trgovino.`
Stavek B: `Pingvini ne letijo.`
Oznaka: `NotNextSentence`

Slika 3.2: Različna vhoda in izhoda pri predvidevanju naslednjega stavka.

Zaradi učenja na teh dveh nalogah se model BERT nauči konteksta v več stavkih in po učenju dobro „razume“ jezik. Razvite so bile tudi različice modela BERT, ki pri predhodnem učenju izpustijo nalogo predvidevanja naslednjega stavka in dobijo enako dobre predstavitve jezika. Primer takega modela je RoBERTa [11]. Ta za predhodno učenje uporablja veliko večjo

količino podatkov in se uči dlje časa kot model BERT. Naloga učenja maskirnega jezikovnega modela se pri tem modelu razlikuje od tiste pri modelu BERT. Model RoBERTa „zamaskira“ vhodne podatke vsakič, preden jih model dobi na vhod, medtem ko BERT to opravi le enkrat pri predobdelavi podatkov. Tako model skozi epohe učenja vidi različne verzije istega stavka z maskami na drugih besedah.

3.2 Prilagajanje modela BERT

Po učenju model dokaj dobro predstavlja splošno znanje o jeziku. Če pa hočemo to znanje uporabiti pri neki specifični nalogi, moramo model prilagoditi tej nalogi. V tej fazi se model uči nadzorovano na označenem besedilu, primernem za ciljno nalogo. Prilagajanje modela BERT je zaradi znanja jezika, pridobljenega v fazi predhodnega učenja, zelo hitro v primerjavi s prilagajanjem drugih večjih modelov za neko NLP-nalogo. To je tudi ena izmed večjih prednosti modela BERT, saj je predhodno učenje treba pognati le enkrat, dobljeni model pa lahko nato uporabimo za osnovo pri prilagajanju za vrsto specifičnih NLP-nalog, kot npr. klasifikacija morfoloških značilnosti besed, odgovarjanje na vprašanja in analiza čustev.

Veliko dela je že bilo vloženega v izboljševanje in nadgrajevanje prilagajanja modela, zato poznamo več različnih metod, ki se med seboj razlikujejo v času učenja, deležu prilagojenih parametrov, dobljenih rezultatih in tudi v velikosti končnega modela. Vsem je skupno to, da se modelu pred prilagajanjem doda sveža izhodna plast (angleško *output layer*) brez kakršnegakoli predhodnega znanja. Ta plast služi generiranju izhoda pri reševanju neke specifične NLP-naloge. Ker nima predhodnega znanja, jo dostikrat inicializiramo kar z naključno generiranimi utežmi.

Govorimo tudi o prilagajanju modela več različnim nalogam in izkoriščanju znanja, dobljenega od drugih nalog pri reševanju ciljne naloge. Če isti model prilagajamo za več različnih nalog zapored, pride do problema katastrofalnega pozabljanja [6, 13]. To pomeni, da model, ko se nauči novih informacij,

zelo hitro pozabi starejše, že naučene informacije. Tako bi se ob prilagajanju za vsako novo nalogo izgubilo večino znanja, pridobljenega pri prilagajanju za prejšnje naloge. Uporabimo metodo združevanja adapterjev, ki prilagajanje deli na dve fazi in tako zmanjša verjetnost katastrofalnega pozabljanja.

3.2.1 Prilagajanje celotnega modela

Pri tej metodi se ob učenju prilagajajo vse uteži v modelu BERT. Uteži se prilagodijo tako, da bo po učenju model čim bolje reševal neko določeno NLP-nalogo. Ker izhodna plast nima predhodnega znanja, se njene uteži močno spremenijo. Ker pa ima preostali predhodno naučeni model že dobro razumevanje jezika, se njegove uteži le malo prilagodijo. Od tod tudi izhaja izraz „prilagajanje“. Ker je prilagojenih parametrov veliko, učenje s to metodo konvergira relativno hitro. To pomeni, da je za pretirano prilagajanje učnim podatkom (angleško *overfitting*) potrebnih manj epoh učenja.

3.2.2 Prilagajanje le zadnje plasti

Zaradi velikega števila parametrov je pri prilagajanju celotnega modela učenje precej kompleksno in dolgotrajno. Prilagajanje le zadnje plasti je hitrejša alternativa z veliko manj prilagojenimi parametri, vendar pa pri NLP-nalogah dosega vidno slabše rezultate kot prilagajanje celotnega modela.

Pri tej metodi gre za prilagajanje izključno uteži v izhodni plasti, medtem ko so uteži v predhodno naučenem modelu zamrznjene. Izhodna plast v modelu BERT predstavlja približno 0.001 % vseh parametrov modela. Zato je učenje veliko bolj enostavno in hitro. Sicer konvergira nekoliko počasneje, zato moramo model učiti z več epohami, vendar pa se te zaradi enostavnosti učenja hitro prilagodijo. Ker se med učenjem prilagajajo le uteži dodane izhodne plasti, ostaja predhodno naučeni model nespremenjen. To pomeni, da lahko kasneje isti model uporabimo za prilagajanje za neko drugo NLP-nalogo. To storimo tako, da modelu odstranimo prilagojeno izhodno plast in mu enostavno dodamo svežo izhodno plast, ki jo prilagodimo za drugo

NLP-nalogo.

3.2.3 Prilagajanje z adapterji

Prilagajanje celotnega modela je lahko nepraktično, če je potrebno reševanje več različnih NLP-nalog, saj je za vsako nalogo posebej potreben nov, celoten model BERT. Ker so ti modeli veliki, lahko to povzroči težave. Zato je bila leta 2019 razvita metoda učenja z adapterji [10]. Gre za modularen pristop k prilagajanju modela BERT.

Ko model prilagajamo za neko specifično nalogo, mu pred učenjem dodamo manjše število novih parametrov. To pomeni vstavljanje novih plasti, katerih uteži so inicializirane naključno, v predhodno naučeno nevronsko mrežo. Tem plastem pravimo „adapter plasti“, skupaj z izhodno plastjo pa tvorijo „adapter“. Med učenjem se uteži osnovne mreže zamrznejo, kar pomeni, da med učenjem ostanejo nespremenjene. Prilagajajo se torej le adapter plasti in izhodna plast. Zaradi manjšega števila prilagojenih parametrov učenje s to metodo konvergira nekoliko počasneje kot pri prilagajanju celotnega modela, kar pomeni, da je za dober model potrebnih več epoh učenja, vendar se te izvedejo hitreje.

Kljub velikosti in veliko manjšemu številu parametrov pa je zmogljivost modela učenega z adapterji primerljiva s tisto, dobljeno s finim prilagajanjem celotnega modela. Izkaže se, da so rezultati učenja z adapterji, ki zavzamejo približno 0,5 – 5 % velikosti celotnega modela, približno 1 % slabši od tistih, dobljenih s prilagajanjem celotnega modela [10].

Ena od prednosti metode prilagajanja z adapterji je prenosljivost znanja. Po končanem učenju lahko adapter ločimo od predhodno učenega modela, ki ostane nespremenjen, obdržimo le adapter plasti in izhodno plast. Ker je parametrov v adapterju malo v primerjavi s predhodno učenim modelom, je shranjevanje le adapterja za neko nalogo veliko bolj praktično. Za reševanje naloge predhodno učenemu modelu dodamo le adapter za to nalogo.

3.2.4 Prilagajanje z metodo združevanja adapterjev

Če predhodno naučeni model prilagajamo z adapterji za več različnih NLP-nalog, lahko pri reševanju neke naloge upoštevamo znanje, pridobljeno z modeli, naučenimi na drugih nalogah. To dosežemo, če model prilagajamo z metodo združevanja adapterjev (AdapterFusion [14]).

Združevanje adapterjev je sestavljeno iz dveh faz: pridobivanje znanja in kombinacija znanja. V fazi pridobivanja znanja adapterje prilagodimo različnim NLP-nalogam. V fazi združevanja znanja te adapterje združimo z metodo AdapterFusion, da izkoristimo njihovo znanje. Pred prilagajanjem za specifično NLP-nalogo predhodno naučenemu modelu dodamo manjše število parametrov, ki jih imenujemo AdapterFusion parametri. Uteži osnovne mreže zamrznemo, prilagajajo se le dodani AdapterFusion parametri in izhodna plast. Med učenjem se ob prilagajanju teh parametrov, poleg vhodnih podatkov, upoštevajo informacije vseh adapterjev, učenih za različne NLP-naloge, vključno z adapterjem, učenim na dani nalogi. AdapterFusion parametre vstavimo med vsako plast transformerja. Ob vsakem časovnem koraku ti parametri prejmejo informacije predhodne plasti in izhode adapterjev. Naučijo se najboljše linearne kombinacije adapterjev za dane podatke in adapterjem določijo ustrezne uteži. Adapter, katerega znanje je bolj koristno za reševanje ciljne naloge, bo imel večjo utež in zato večji vpliv na izhod kot adapter, katerega znanje je manj koristno. Z deljenjem prilagajanja na dve fazi zmanjšamo verjetnost katastrofalnega pozabljanja in učnih nestabilnosti.

AdapterFusion parametre, prilagojene za reševanje neke NLP-naloge, lahko tako kot adapterje shranimo ločeno od predhodno učenega modela in jih uporabimo le ob reševanju te NLP-naloge. Ta metoda prilagajanja konvergira zelo hitro, zato je za dober model potrebnih malo epoh učenja.

Poglavje 4

Prilagajanje modela za obdelovanje slovenskih besedil

Cilj naloge je raziskati delovanje različnih metod prilagajanja predhodno naučenega modela BERT ter ugotoviti, kako učinkovite so pri obdelavi slovenskih besedil. V ta namen potrebujemo model BERT, predhodno naučen na slovenskih besedilih. Ker je predhodno učenje takega modela računsko in časovno zahtevno, uporabimo že naučena modela CroSloEngual BERT [15] in mBERT. Prvi je večjezikovni model, učen na slovenskih, hrvaških in angleških besedilih z boljšo predstavitvijo teh treh jezikov kot večjezikovni modeli, učeni na več jezikih. Model mBERT je večjezikovni model, učen na 104 jezikih.

V tem poglavju razložimo postopek evalvacije različnih metod prilagajanja. Predstavimo večjezikovna modela CroSloEngual BERT in mBERT ter učni korpus ssj500k. Opišemo ciljni nalogi NER in označevanje UPOS ter pojasnimo izbiro adapterjev za metodo združevanja.

4.1 Evalvacija metod prilagajanja

Prilagajanje modelov analiziramo na dveh različnih NLP-nalogah, označevanje univerzalnih besednih vrst (angleško *universal part-of-speech*, UPOS) in pre-

poznavanju imenskih entitet (angleško *named entity recognition*, NER). Pri obeh nalogah gre za klasifikacijo besed. Pri označevanju UPOS model vsaki vhodni besedi določi njeno univerzalno besedno vrsto. Pri NER model v vhodnem besedilu prepozna imenske entitete in jih ustrezno označi. Predhodno naučena modela CroSloEngual BERT in mBERT prilagajamo s štirimi različnimi metodami: prilagajanje celotnega modela, prilagajanje le zadnje plasti, prilagajanje z adapterji in prilagajanje z metodo združevanja adapterjev. Za slednjo uporabimo adapterje, učene na več različnih NLP-nalogah za klasifikacijo besed. Za učenje in evalvacijo vseh nalog uporabimo učni korpus ssj500k [5].

Učinkovitost modela pri reševanju neke naloge merimo z oceno F_1. Ta ocena prestavlja harmonično sredino točnosti (angleško *precision*) in priklica (angleško *recall*) modela. Točnost je delež pozitivnih klasifikacij, ki so pravilne (množica pozitivnih klasifikacij vsebuje tudi lažno pozitivne klasifikacije). Priklic modela je delež vseh pozitivnih testnih podatkov, ki so bili pozitivno klasificirani. Tej meri pravimo tudi občutljivost (angleško *sensitivity*). Nizka točnost modela kaže na veliko število lažno pozitivnih klasifikacij. Nizka vrednost priklica kaže na veliko število lažno negativnih klasifikacij.

4.2 CroSloEngual BERT in mBERT

Večina modelov za obdelovanje besedil s transformerji podpira angleščino in nekaj drugih dobro podprtih jezikov, kot so kitajščina, nemščina in francoščina. Javno dostopna sta tudi dva velika večjezikovna modela, mBERT in XLM-R.

Model mBERT je večjezikovni model, učen na 104 jezikih. Ti jeziki so izbrani, ker imajo Wikipedie v teh jezikih največjo količino besedil. Ta besedila so uporabljena pri predhodnem učenju modela. mBERT uporablja osnovno BERT arhitekturo (*bert-base*), kar pomeni, da ima 12 plasti, velikost skritih plasti pa je 768. Model ima skupaj 110 milijonov parametrov. Na voljo sta dve različici. Ena pri predstavitvi jezika upošteva velike in male začetnice (*mbert-cased*), druga pa ne (*mbert-uncased*).

Izkaže se, da večjezikovni modeli, učeni na manjšem številu različnih jezikov, pri obdelavi teh jezikov dajejo boljše rezultate kot modeli, učeni na večjem številu [17]. Zato je bil razvit model CroSloEngual BERT [15], ki je naučen na približno 6 milijardah besed iz angleščine, slovenščine in hrvaščine. Zaradi tega je ta model primernejši za obdelavo slovenskih besedil kot drugi večjezikovni modeli. CroSloEngual BERT uporablja isto arhitekturo kot mBERT (*bert-base*) in je iste velikosti. Pri predstavitvi jezika upošteva velike in male začetnice. Predhodno učenje modela je potekalo na Google Cloud TPU v2 in je trajalo približno tri tedne.

4.3 Korpus ssj500k

Ssj500k je učni korpus za obdelavo slovenskih besedil. Vsebuje okoli 500.000 besed z ročno preverjenimi oznakami. Korpus je na voljo v treh formatih: CoNLL-U, vert in TEI (*Text Encoding Initiative*).

4.3.1 Format CoNLL-U

Format CoNLL-U je nekoliko razširjena verzija formata CoNLL-X [2]. Vsaka beseda ali znak je napisana v svoji vrstici, skupaj tvorijo stavke. Stavki so med seboj ločeni s praznimi vrsticami. Zapis besede v vrstici je sestavljen iz desetih polj v obliki:

```
ID FORM LEMMA UPOS XPOS FEATS HEAD DEPREL DEPS MISC
```

- `ID`: Indeks besede ali znaka v stavku. Indeksiranje se začne z 1.
- `FORM`: Oblika besede ali znaka, uporabljenega v stavku.
- `LEMMA`: Osnovna oblika besede ali znaka.
- `UPOS`: Univerzalna besedna vrsta.
- `XPOS`: Besedne vrste posameznega jezika.

- `FEATS`: Seznam univerzalnih morfoloških značilnosti besede ali znaka. Številnost, spol in sklon so primeri univerzalnih morfoloških značilnosti.
- `HEAD`: Indeks sintaktičnega starša besede ali znaka. Vrednost je 0, če beseda nima sintaktičnega starša.
- `DEPREL`: Razmerje univerzalnih odvisnosti med besedo in njenim sintaktičnim staršem. Vrednost je *root*, če beseda nima sintaktičnega starša.
- `DEPS`: Graf odvisnosti med besedami. Ima obliko seznama parov (`HEAD`, `DEPREL`).
- `MISC`: Kakršnekoli dodatne označbe besed ali znakov.

Zapis posamezne besede ali znaka mora vsebovati vsa našteta polja in nobeno polje ne sme biti prazno. Če katera beseda za določeno polje nima določene vrednosti, to označimo z znakom „_“.

4.3.2 Format vert

V formatu vert, ki ga uporablja predvsem programsko orodje *Sketch Engine*, je vsaka beseda, znak ali številka napisana v svoji vrstici, podobno kot pri CoNLL-U. Vsaka vrstica lahko vsebuje več polj za dodatne označbe, ki so med seboj ločene s tabulatorjem. Za razliko od formata CoNLL-U format vert vsebuje oznake tipa XML, ki označujejo razne strukture v besedilu. Z njimi so lahko označene kakršnekoli strukture, kot npr. stavki, odstavki, dokumenti in tudi poimenovane entitete (angleško *named entity*).

4.4 Ciljni nalogi

Da lahko ocenimo učinkovitost različnih metod prilagajanja, vnaprej naučeni model prilagajamo za dve različni nalogi: NER in označevanje UPOS . Pri obeh nalogah gre za klasifikacijo besed.

4.4.1 Naloga NER

Imenske entitete so objekti iz sveta, ki imajo imena. V korpusu ssj500k so označene tri različne vrste imenskih entitet: oseba, lokacija in organizacija. Oznake za njih so „PER“, „LOC“ in „ORG“. Če je neka beseda prva v imenski entiteti, ima pred oznako še predpono „B-“. Če je neka beseda del imenske entitete, ni pa prva, ima pred oznako predpono „I-“. Vse besede, ki niso del imenskih entitet, imajo oznako „O“. Pri nalogi NER vsaki besedi določimo ustrezno oznako iz tega nabora. Na sliki 4.1 vidimo primer stavka, označenega z oznakami UPOS, predstavljenega enako kot v učnih podatkih. Za učenje in evalvacijo te naloge uporabimo korpus ssj500k v formatu vert.

```
Janez       B-PER
Novak       I-PER
živi        O
v           O
Ljubljani   B-LOC
.           O
```

Slika 4.1: Primer stavka, označenega z imenskimi entitetami.

4.4.2 Označevanje UPOS

Označevanje UPOS je klasifikacijska naloga, pri kateri vsaki besedi določimo njeno besedno vrsto iz nabora univerzalnih besednih vrst. V nabor spada 16 različnih besednih vrst, primeri so samostalnik, glagol in ločilo. Na sliki 4.2 vidimo primer stavka, označenega z oznakami UPOS, predstavljenega enako kot v učnih podatkih. Za učenje in evalvacijo te naloge uporabimo korpus ssj500k v CoNLL-U formatu, ki vsebuje tudi oznake UPOS.

```
Janez      PROPN
Novak      PROPN
živi       VERB
v          ADP
Ljubljani  PROPN
.          PUNCT
```

Slika 4.2: Primer stavka, označenega z oznakami UPOS.

4.5 Uporabljeni adapterji

Pri metodi združevanja adapterjev več adapterjev, učenih na različnih nalogah, prispeva znanje za reševanje želene naloge. Bolj kot je znanje adapterja koristno pri reševanju ciljne naloge, bolj ga bo model uporabljal. Malo verjetno je, da bi vključitev nekega adapterja v model negativno vplivala na rezultat. Če bi bilo znanje adapterja škodljivo za reševanje ciljne naloge, bi model to zaznal in adapterja ne bi uporabljal. Sklepamo, da učinkovitost modela pri reševanju neke naloge raste z dodajanjem novih adapterjev, saj več adapterjev pomeni večjo verjetnost, da imajo nekateri znanje, koristno za reševanje ciljne naloge. To drži le z manjšim številom dodanih adapterjev. Pri združevanju velikega števila adapterjev pride do pretiranega prilagajanja učni množici.

Za prilagajanje z metodo združevanja adapterjev pripravimo osem adapterjev, prilagojenih za osem različnih klasifikacijskih nalog. Pri vseh gre za klasifikacijo besed, enako kot pri obeh ciljnih nalogah. Te naloge izberemo, ker se osredotočimo na vpliv, ki ga ima znanje o ostalih osnovnih lastnostih besede na njeno klasifikacijo. Učeni so na naslednjih nalogah:

- NER (Model v besedilu prepozna in ustrezno označi imenske entitete.)
- Določanje `UPOS` oznake besede. (Model vsem besedam določi njihovo besedno vrsto iz nabora univerzalnih besednih vrst.)

- Določanje XPOS oznake besede. (Model vsem besedam določi njihovo besedno vrsto iz nabora besednih vrst, značilnih za slovenščino.)
- Določanje DEPREL oznake besede. (Določanje vrste sintaktičnega razmerja med besedo in njenim sintaktičnim staršem.)
- Določanje slovničnega spola besede. (Model vsem besedam določi slovnični spol. Na voljo ima štiri različne oznake: moški, ženski, srednji spol in oznako "_", če beseda nima spola.)
- Določanje sklona besede. (Model vsem besedam določi sklon. Na voljo ima sedem različnih oznak, šest za vsakega izmed različnih sklonov in oznako "_", če beseda nima sklona.)
- Določanje slovničnega števila besede. (Model vsem besedam določi slovnično število. Na voljo ima štiri različne oznake: ednina, dvojina, množina in oznako "_", če beseda nima slovničnega števila.)
- Določanje slovnične osebe besede. (Model vsem besedam določi slovnično osebo. Na voljo ima štiri različne oznake: prva, druga, tretja oseba in oznako "_", če beseda nima slovnične osebe.)

Učni podatki, razen za nalogo NER, so dostopni v CoNLL-U formatu korpusa ssj500k. Adapterje smo za ciljno nalogo prilagodili na predhodno učenih modelih CroSloEngual BERT in mBERT.

Poglavje 5

Rezultati

Primerjamo prilagajanje modelov z različnimi metodami pri nalogi NER in označevanju UPOS. Vsa prilagajanja modelov smo izvajali na Geforce RTX 2070 GPU z 8 GB spomina. Izvajali smo učenje s polovično preciznostjo (angleško *half-precision learning*) [7], kjer model decimalna števila hrani s 16 biti namesto s privzetimi 32. To zmanjša porabljeno količino spomina in omogoča učenje večjih modelov. Prilagajanja z metodo združevanja adapterjev pri obeh modelih in prilagajanja celotnega modela mBERT smo izvajali z velikostjo serije (angleško *batch size*), nastavljeno na 8. Pri vseh ostalih prilagajanjih je velikost serije nastavljena na 16.

5.1 Adapterji

Za različne klasifikacijske naloge smo prilagodili adapterje, ki jih uporabimo tudi pri metodi združevanja adapterjev. Za vsako nalogo smo prilagodili več adapterjev z različnim številom epoh učenja. Ker imata CroSloEngual BERT in mBERT različne slovarje, adapterjev prilagajanih na enem od teh modelov, ne moremo uporabiti na drugem. Zato smo adapterje klasifikacijskim nalogam prilagajali na obeh modelih. Za uporabo v metodi združevanja adapterjev smo izbrali tiste, ki so dosegli najboljšo oceno F_1 na validacijski množici pri svoji nalogi.

V tabeli 5.1 vidimo podrobnosti izbranih adapterjev, prilagojenih na modelu CroSloEngual BERT. Podrobnosti izbranih adapterjev, prilagojenih na modelu mBERT, vidimo v tabeli 5.2. Posamezen adapter je velik 9.6 MB, medtem ko sta CroSloEngual BERT in mBERT oba velika 499.5 MB. Vsi adapterji so iste velikosti in predstavljajo 1.89 % vseh parametrov modela.

NLP naloga	Št. epoh	Čas učenja	F_1
NER	100	47 min	81.68 %
UPOS	80	1 h 39 min	98.68 %
XPOS	50	1 h 6 min	96.87 %
DEPREL	50	1 h 3 min	96.50 %
Sklon	50	1 h 2 min	97.57 %
Slovnični spol	50	1 h 1 min	96.63 %
Slovnično število	50	1 h 1 min	99.27 %
Slovnična oseba	10	12 min	99.65 %

Tabela 5.1: Podrobnosti adapterjev, prilagojenih klasifikacijskim nalogam na modelu CroSloEngual BERT.

NLP naloga	Št. epoh	Čas učenja	F_1
NER	100	49 min	85.10 %
UPOS	80	1 h 42 min	98.62 %
XPOS	50	1 h 5 min	94.57 %
DEPREL	50	1 h 4 min	95.07 %
Sklon	50	1 h 3 min	94.46 %
Slovnični spol	50	1 h 4 min	94.57 %
Slovnično število	50	1 h 3 min	98.37 %
Slovnična oseba	10	13 min	99.06 %

Tabela 5.2: Podrobnosti adapterjev, prilagojenih klasifikacijskim nalogam na modelu mBERT.

Opazimo, da se epohe pri NER izvajajo hitreje kot pri ostalih klasifikacijskih nalogah. To se zgodi zaradi neenakomerne porazdelitve oznak. Delež imenskih entitet v besedilu je majhen, zato ima velika večina besed oznako „O“. Pri ostalih klasifikacijskih nalogah so oznake bolj enakomerno porazdeljene.

Označevanje UPOS je klasifikacijska naloga nižjega nivoja, saj je vsaka oznaka UPOS določena eni besedi, medtem ko so oznake NER lahko določene zaporedju besed. Pri NER mora model poleg imenskih entitet prepoznati tudi njihov začetek in konec. Zato je NER klasifikacijska naloga višjega nivoja.

Vidimo, da so adapterji, prilagojeni na CroSloEngual BERT, boljši od adapterjev, prilagojenih na mBERT, pri vseh klasifikacijskih nalogah razen NER. To pomeni, da so za reševanje klasifikacijskih nalog nižjega nivoja bolj učinkoviti adapterji, prilagojeni na modelu CroSloEngual BERT.

Da smo ugotovili, ali je adapterjev za združevanje preveč in ali kateri od adapterjev škodi reševanju ciljne naloge, smo model prilagajali z združevanjem vseh različnih kombinacij teh adapterjev, kjer manjka eden izmed adapterjev. Izkaže se, da z izključitvijo kateregakoli adapterja iz združevanja prilagojeni model dosega slabše rezultate kot pri prilagajanju z združevanjem vseh adapterjev. To pomeni, da najboljša kombinacija združenih adapterjev za prilagajanje vsebuje vse adapterje, ki smo jih prilagajali drugim klasifikacijskim nalogam.

5.2 Naloga NER

Predstavimo rezultate različnih metod prilagajanj modela nalogi NER. Vsako metodo prilagajanja smo poskusili večkrat z različnim številom epoh. Za primerjavo smo izbrali modele, ki so pri NER dosegli najvišjo oceno F_1 na validacijski množici.

Ocena F_1 predstavlja harmonično sredino točnosti in priklica modela. Točnost modela je pri nalogi NER delež vseh prepoznanih imenskih entitet,

ki so bile pravilno prepoznane. Da je entiteta pravilno prepoznana, se mora popolnoma ujemati z ustrezno entiteto v testnih podatkih. Imeti mora pravilno določen začetek, konec in vrsto entitete. Priklic modela predstavlja delež imenskih entitet v testnih podatkih, ki jih model pravilno prepozna.

V tabeli 5.3 vidimo rezultate prilagajanj modela CroSloEngual BERT za nalogo NER. Rezultate prilagajanj modela mBERT predstavimo v tabeli 5.4. Za vsako metodo prilagajanja v tabeli vidimo število epoh učenja modela, čas učenja modela, delež prilagojenih parametrov v modelu in F_1 oceno modela.

Način prilagajanja	Št. epoh	Čas učenja	Parametri	F_1
Celotni model	5	3 m	100 %	83.42 %
Zadnja plast	80	20 m	0.01 %	58.42 %
Adapter	100	47 m	1.89 %	81.68 %
Združevanje adapterjev	6	12 m	25.64 %	82.75 %

Tabela 5.3: Rezultati prilagajanja modela CroSloEngual BERT za NER.

Način prilagajanja	Št. epoh	Čas učenja	Parametri	F_1
Celotni model	7	6 min	100 %	85.45 %
Zadnja plast	100	22 min	0.01 %	61.33 %
Adapter	100	49 min	1.89 %	85.10 %
Združevanje adapterjev	5	11 min	25.64 %	85.58 %

Tabela 5.4: Rezultati prilagajanja modela mBERT za NER.

Najboljšo oceno F_1 za nalogo NER pri modelu CroSloEngual BERT doseže celotno prilagojeni model. Vidimo, da upoštevanje drugih adapterjev pri metodi združevanja adapterjev izboljša rezultat modela, prilagojenega s posameznim adapterjem. To pomeni, da je znanje adapterjev, prilagajanih drugim klasifikacijskim nalogam, koristno pri reševanju NER. Vidimo, da model, prilagajan s posameznim adapterjem, kljub zelo malemu deležu prilagojenih parametrov dosega primerljive rezultate kot model, ki ima pri-

lagojene vse parametre, kar pomeni, da so adapterji veliko bolj učinkoviti pri izrabi parametrov.

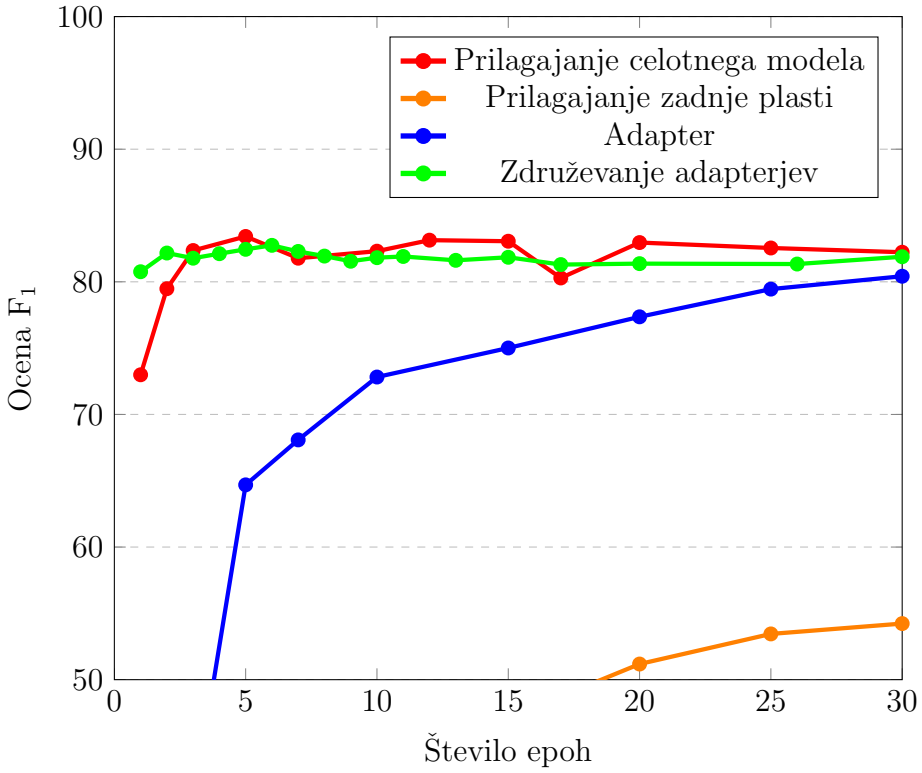
Pri prilagajanju modela mBERT za nalogo NER najboljšo oceno F_1 doseže model, prilagojen z metodo združevanja adapterjev. Epohe se pri prilagajanju celotnega modela mBERT izvajajo nekoliko počasneje kot pri prilagajanju celotnega modela CroSloEngual BERT. To je posledica različnih velikosti serije, saj je ta pri prilagajanju mBERT 8, pri prilagajanju CroSloEngual BERT pa 16.

Znanje adapterjev, prilagojenih za druge klasifikacijske naloge pri mBERT dovolj izboljša rezultat posameznega adapterja, da preseže rezultat prilagajanja celotnega modela. To nam pri prilagajanju modela CroSloEngual BERT ni uspelo. Rezultat posameznega adapterja je pri mBERT bližji rezultatu prilagajanja celotnega modela kot pri CroSloEngual BERT. Metoda združevanja adapterjev pri prilagajanju CroSloEngual BERT oceno F_1 posameznega adapterja izboljša za 1,07 %, pri prilagajanju mBERT pa za 0,48 %.

Pri nalogi NER dosega model mBERT boljše rezultate kot CroSloEngual BERT, kar je v nasprotju z rezultatom, da je CroSloEngual BERT pri opravljanju te naloge bolj učinkovit [15]. Razlike v rezultatih so lahko posledica učenja s polovično točnostjo ali razlike v velikosti serij.

Vidimo, da se pri nekaterih metodah epohe učenja izvedejo hitreje kot pri drugih. Najhitreje se izvedejo pri prilagajanju le zadnje plasti, vendar ta metoda dosega slabše rezultate. Zaradi manjšega števila parametrov se epohe učenja pri prilagajanju z adapterjem izvedejo hitreje kot pri prilagajanju celotnega modela. Ker je prilagajanje z metodo združevanja adapterjev bolj kompleksno, se epohe učenja pri tej metodi izvajajo najdlje.

Za najboljše rezultate je pri različnih metodah prilagajanja bilo potrebno različno število epoh učenja. To pomeni, da modeli, prilagajani z različnimi metodami, konvergirajo različno hitro. Model smo nehali prilagajati z več epohami, ko se njegova ocena F_1 že dolgo ni izboljšala ali se je začela slabšati. Hitrost konvergence modelov pri prilagajanju CroSloEngual BERT za nalogo NER si lahko ogledamo na sliki 5.1.



Slika 5.1: Hitrost konvergence modelov pri prilagajanju CroSloEngual BERT za NER.

Pri prilagajanju z metodo združevanja adapterjev ima model že na začetku učenja na voljo znanje posameznega adapterja, prilagojenega za NER, saj je ta eden izmed adapterjev, ki jih uporabimo pri združevanju. Zaradi tega model, prilagojen s to metodo, konvergira najhitreje. Pri metodi prilagajanja celotnega modela model konvergira hitreje kot tisti, prilagojen z adapterji. To je posledica večjega števila prilagojenih parametrov. Model, prilagojen z adapterji, tako potrebuje več epoh učenja, da doseže primerljive rezultate. Najpočasneje konvergira model s prilagojeno le zadnjo plastjo, ker ima ta najmanj prilagojenih parametrov.

5.3 Označevanje UPOS

Rezultate različno prilagojenih modelov CroSloEngual BERT za označevanje UPOS predstavimo v tabeli 5.5. V tabeli 5.6 vidimo rezultate prilagajanj modela mBERT za označevanje UPOS. Za primerjavo smo izbrali rezultate modelov, ki so z določeno metodo prilagajanja dosegli najboljšo oceno F_1 na validacijski množici.

Način prilagajanja	Št. epoh	Čas učenja	Parametri	F_1
Celotni model	10	17 min	100 %	98.82 %
Zadnja plast	80	42 min	0.01 %	95.05 %
Adapter	80	1 h 39 min	1.89 %	98.68 %
Združevanje adapterjev	7	40 min	25.64 %	98.73 %

Tabela 5.5: Rezultati prilagajanja modela CroSloEngual BERT za označevanje UPOS.

Način prilagajanja	Št. epoh	Čas učenja	Parametri	F_1
Celotni model	5	13 min	100 %	98.72 %
Zadnja plast	80	45 min	0.01 %	90.62 %
Adapter	80	1 h 42 min	1.89 %	98.62 %
Združevanje adapterjev	5	29 min	25.64 %	98.73 %

Tabela 5.6: Rezultati prilagajanja modela mBERT za označevanje UPOS.

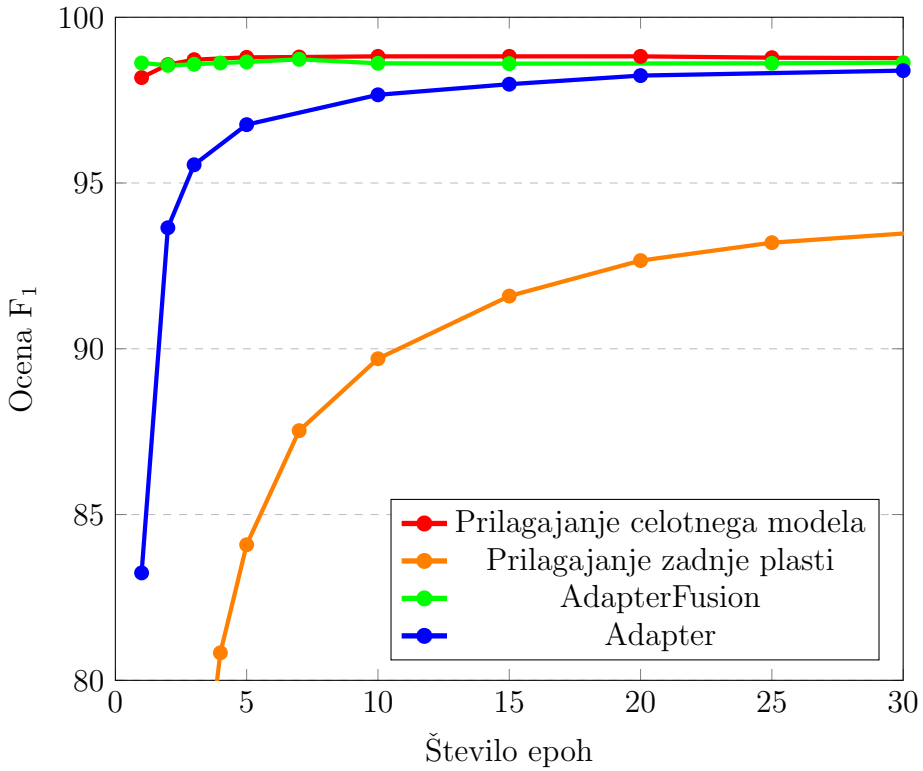
Pri nalogi označevanja UPOS z modelom CroSloEngual BERT je najvišjo oceno F_1 dosegel celotno prilagojeni model. Model, prilagojen z adapterjem, dosega primerljive rezultate, njegovo učenje pa je dolgotrajno. Model, prilagojen z metodo združevanja adapterjev, dosega boljše rezultate kot prilagojeni s posameznimi adapterji. Znanje adapterjev, prilagojenih za druge klasifikacijske naloge, je torej koristno pri označevanju UPOS. Ta izboljšava vseeno ni dovolj, da bi model dosegal boljše rezultate kot celotno prilagojeni

model. Vidimo, da so pri tej nalogi rezultati modela s prilagojeno le zadnjo plastjo bližje ostalim rezultatom kot pri NER.

Rezultati modela mBERT pri označevanju UPOS so podobni rezultatom CroSloEngual BERT. Najboljšo oceno F_1 doseže celotno prilagojeni model. Prilagajanje z združevanjem adapterjev dosega enako dobre rezultate kot pri CroSloengual BERT. Vsi ostali rezultati so pri mBERT nekoliko slabši. Največja razlika je pri rezultatih prilagajanja z zadnjo plastjo. CroSloEngual BERT s tem prilagajanjem pri označevanju UPOS doseže 4,43 % boljše rezultate kot mBERT.

Posamezne epohe se pri tej nalogi izvajajo dlje kot pri NER. Vidimo, da modeli, prilagajani z različnimi metodami, konvergirajo različno hitro. Hitrost konvergence pri nalogi označevanje UPOS je predstavljena na sliki 5.2.

Vidimo, da model prilagojen z metodo združevanja adapterjev, že po prvi epohi učenja vrača dobre rezultate. Ti rezultati se s povečevanjem epoh učenja le malo spreminjajo. Razlika med najboljšim in najslabšim rezultatom, doseženim s to metodo, je 0,1 %. Celotno prilagojeni model konvergira zelo hitro. Rezultati, dobljeni z dvema ali več epohami, se med seboj zelo malo razlikujejo. Model, prilagajan z adapterji, potrebuje manj epoh učenja kot pri NER, da začne dosegati rezultate, primerljive s prilagajanjem celotnega modela in metodo združevanja adapterjev. Zaradi najmanjšega števila prilagojenih parametrov model s prilagojeno le zadnjo plastjo konvergira najpočasneje.



Slika 5.2: Hitrost konvergence modelov pri prilagajanju CroSloEngual BERT za označevanje UPOS.

Poglavje 6

Zaključek

Cilj naloge je bil preučiti različne metode prilagajanja modelov BERT klasifikacijskim nalogam v slovenščini. Za evalvacijo teh metod smo večjezikovna modela CroSloEngual BERT in mBERT prilagajali na klasifikacijskih nalogah NER in označevanju UPOS. Uporabili smo štiri metode prilagajanja: prilagajanje celotnega modela, prilagajanje le zadnje plasti modela, prilagajanje z adapterjem in prilagajanje z metodo združevanja adapterjev. V slednji smo uporabili adapterje, ki smo jih prilagodili različnim nižjenivojskim klasifikacijskim nalogam v slovenščini.

Pri obeh ciljnih nalogah so rezultati modela, prilagojenega z združevanjem adapterjev, boljši od rezultatov modela, prilagojenega s posameznim adapterjem. To pomeni, da je znanje adapterjev, prilagajanih drugim klasifikacijskim nalogam, koristno tudi pri nalogah NER in označevanje UPOS.

Prilagajanje z metodo združevanja adapterjev je pri modelu mBERT doseglo najboljšo oceno F_1 pri nalogi NER in pri označevanju UPOS. Pri prilagajanju modela CroSloEngual BERT je pri obeh nalogah najboljši rezultat dosegla metoda prilagajanja celotnega modela.

Pri NER je razlika med rezultati prilagajanja s posameznim adapterjem in rezultati prilagajanja z metodo združevanja adapterjev večja kot pri označevanju UPOS. To pomeni, da združevanje adapterjev, prilagojenih drugim klasifikacijskim nalogam, bolj pozitivno vpliva na rezultat pri reševanju

višjenivojskih klasifikacijskih nalog. Pri nižjenivojskih klasifikacijskih nalogah je razlika v rezultatih manjša.

Ugotovili smo, da prilagajanje modela z metodo združevanja adapterjev pri nalogi NER in označevanju UPOS lahko doseže primerljive ali celo boljše rezultate kot prilagajanje celotnega modela. Slabost tega pristopa je čas učenja. Prilagajanje celotnega modela dosega dobre rezultate po le nekaj epohah učenja, ki trajajo skupaj nekaj minut. Za metodo združevanja adapterjev moramo v fazi pridobivanja znanja prilagoditi vsak adapter posebej. Adapterji konvergirajo počasneje, zato potrebujejo več epoh učenja. V fazi kombiniranja adapterjev model že po prvi epohi vrača dobre rezultate, ki se lahko z več epohami le malo izboljšajo. Pri tej metodi se epohe učenja izvajajo najdlje.

Za združevanje adapterjev smo uporabili le adapterje, prilagojene na klasifikacijskih nalogah. Združevanje bi lahko poskusili izboljšati z vključitvijo adapterjev, prilagojenih nalogam drugih vrst, recimo analize čustev in odgovarjanja na vprašanja.

Zaradi omejenega spomina smo vsa prilagajanja z združevanjem adapterjev in prilagajanje celotnega modela mBERT izvajali z velikostjo serije 8, vsa druga prilagajanja pa z velikostjo serije 16. Primerjava rezultatov bi bila boljša, če bi vsa prilagajanja izvajali z isto velikostjo serije.

V delu smo adapterje prilagajali le nižjenivojskim klasifikacijskim nalogam z izjemo NER. Vemo, da je znanje adapterjev, prilagojenih nižjenivojskim klasifikacijskim nalogam, koristno pri reševanju NER. Ne vemo pa, kako koristno bi bilo znanje adapterja, prilagojenega na neki drugi, višjenivojski klasifikacijski nalogi. V ta namen bi v združevanje adapterjev za nalogo NER dodali nov adapter, ki je bil prilagojen na neki drugi, višjenivojski klasifikacijski nalogi, npr. skladenjskem razčlenjevanju ali logičnem sklepanju.

Literatura

[1] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2015.

[2] Sabine Buchholz and Erwin Marsi. Conll-x shared task on multilingual dependency parsing. In *Proceedings of the Tenth Conference on Computational Natural Language Learning*, CoNLL-X '06, page 149–164, 2006.

[3] Jacob Devlin and Ming-Wei Chang. Open sourcing BERT: State-of-the-art pre-training for natural language processing. Dosegljivo: `https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html`, 2018. [Dostopano: 29. 6. 2020].

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[5] Kaja Dobrovoljc, Tomaž Erjavec, and Simon Krek. The Universal Dependencies treebank for Slovenian. In *Proceedings of the 6th Workshop on Balto-Slavic Natural Language Processing*, pages 33–38, 2017.

[6] Robert French. Catastrophic forgetting in connectionist networks. *Trends in cognitive sciences*, 3:128–135, 05 1999.

[7] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan. Deep learning with limited numerical precision. In *Proceedings of ICML*, 2015.

[8] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6:107–116, 1998.

[9] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

[10] N. Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin de Laroussilhe, Andrea Gesmundo, Mona Attariyan, and S. Gelly. Parameter-efficient transfer learning for NLP. In *Proceedings of ICML*, 2019.

[11] Y. Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, M. Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: A robustly optimized BERT pretraining approach. *ArXiv*, abs/1907.11692, 2019.

[12] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, 2015.

[13] M. McCloskey and N. J. Cohen. Catastrophic interference in connectionist networks: The sequential learning problem. *Psychology of Learning and Motivation*, 24:109–165, 1989.

[14] Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. *ArXiv*, abs/2005.00247, 2020.

[15] Matej Ulčar and Marko Robnik Šikonja. FinEst BERT and CroSloEngual BERT: less is more in multilingual models. In *Proceedings of Text,*

Speech, and Dialogue - 23rd International Conference, TSD 2020, pages 104–111. Springer, 2020.

[16] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010, 2017.

[17] Antti Virtanen, Jenna Kanerva, Rami Ilo, Jouni Luoma, Juhani Luotolahti, Tapio Salakoski, Filip Ginter, and Sampo Pyysalo. Multilingual is not enough: BERT for Finnish. *CoRR*, abs/1912.07076, 2019.