Analiza i hibridizacija jednostavnog modela transformera za modularno zbrajanje koristeći mehanističku interpretabilnost

Nekić, Filip

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 576

ANALIZA I HIBRIDIZACIJA JEDNOSTAVNOG MODELA TRANSFORMERA ZA MODULARNO ZBRAJANJE KORISTEĆI MEHANISTIČKU INTERPRETABILNOST

Filip Nekić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 576

ANALIZA I HIBRIDIZACIJA JEDNOSTAVNOG MODELA TRANSFORMERA ZA MODULARNO ZBRAJANJE KORISTEĆI MEHANISTIČKU INTERPRETABILNOST

Filip Nekić

Zagreb, lipanj 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

DIPLOMSKI ZADATAK br. 576

Pristupnik:	**Filip Nekić (0036514650)**
Studij:	Računarstvo
Profil:	Računarska znanost
Mentor:	doc. dr. sc. Mario Brčić

Zadatak: **Analiza i hibridizacija jednostavnog modela transformera za modularno zbrajanje koristeći mehanističku interpretabilnost**

Opis zadatka:

Iznenadna ponašanja (engl. emergent capabilities) u neuronskim mrežama karakteriziraju se kao neočekivane ili kvalitativno nove sposobnosti. Neočekivana ponašanja naglašavaju kompleksnu i dinamičnu prirodu takvih sustava te je njihovo razumijevanje i korištenje od ključne važnosti za napredak sposobnosti neuronskih mreža te otključavanje njihovog punog potencijala u različitim područjima i aplikacijama. U ovom diplomskom radu treba provesti analizu jednostavnog transformera dizajniranog i utreniranog za modularno zbrajanje koristeći mehanističku interpretabilnost. Kroz proces analize koji uključuje stjecanje dubokog, algoritamskog razumijevanja unutarnjeg djelovanja modela, treba razotkriti algoritamske krugove (engl. algorithmic circuits) i mehanizme navedene neuronske mreže kako bi se razjasnilo kako uči izvršavati modularno zbrajanje. Također, treba testirati hipoteze zamjenom neuralnih dijelova sa ručno kodiranim komponentama koje implementiraju otkrivene neuralne algoritamske krugove. Takve tehnike hibridizacije u budućnosti mogu dovesti do veće efikasnosti, pouzdanosti i sigurnosti modernih sustava umjetne inteligencije.

Rok za predaju rada: 28. lipnja 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 576

ANALIZA I HIBRIDIZACIJA JEDNOSTAVNOG MODELA TRANSFORMERA ZA MODULARNO ZBRAJANJE KORISTEĆI MEHANISTIČKU INTERPRETABILNOST

Filip Nekić

Zagreb, kolovoz, 2024.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 4. ožujka 2024.

DIPLOMSKI ZADATAK br. 576

Pristupnik:	**Filip Nekić (0036514650)**
Studij:	Računarstvo
Profil:	Računarska znanost
Mentor:	doc. dr. sc. Mario Brčić

Zadatak: **Analiza i hibridizacija jednostavnog modela transformera za modularno zbrajanje koristeći mehanističku interpretabilnost**

Opis zadatka:

Iznenadna ponašanja (engl. emergent capabilities) u neuronskim mrežama karakteriziraju se kao neočekivane ili kvalitativno nove sposobnosti. Neočekivana ponašanja naglašavaju kompleksnu i dinamičnu prirodu takvih sustava te je njihovo razumijevanje i korištenje od ključne važnosti za napredak sposobnosti neuronskih mreža te otključavanje njihovog punog potencijala u različitim područjima i aplikacijama. U ovom diplomskom radu treba provesti analizu jednostavnog transformera dizajniranog i utreniranog za modularno zbrajanje koristeći mehanističku interpretabilnost. Kroz proces analize koji uključuje stjecanje dubokog, algoritamskog razumijevanja unutarnjeg djelovanja modela, treba razotkriti algoritamske krugove (engl. algorithmic circuits) i mehanizme navedene neuronske mreže kako bi se razjasnilo kako uči izvršavati modularno zbrajanje. Također, treba testirati hipoteze zamjenom neuralnih dijelova sa ručno kodiranim komponentama koje implementiraju otkrivene neuralne algoritamske krugove. Takve tehnike hibridizacije u budućnosti mogu dovesti do veće efikasnosti, pouzdanosti i sigurnosti modernih sustava umjetne inteligencije.

Rok za predaju rada: 28. lipnja 2024.

Sadržaj

1. Uvod

Pojava dubokog učenja je revolucionizirala mnoštvo domena umjetne inteligencije. Transformeri su se pokazali kao svestrani modeli te primjenu pronalaze u velikom broju zadataka vezanih za strojno učenje i umjetnu inteligenciju [1]. Transformeri su postali jedni od najutjecajnijih modela u rješavanju neurolingvističkih problema (engl. Natural language processing (NLP)) [2]. Danas modeli kao što su GPT i BERT pokazuju izvanredne rezultate na zadatcima generiranja teksta ili analize sentimenta [3]. Iako postižu izvrsne rezultate, unutrašnji procesi, kako transformeri funkcioniraju i donose odluke nam ostaje nepoznato. Kao posljedica sve je veći fokus na mehanističkoj interpretabilnosti.

Današnji glavni modeli su obično jako kompleksni i sadrže jako veliki broj parametara (ponekad preko 100 bilijuna) te se sastoje od velikog broja slojeva. Mehanistička interpretabilnost je relativno novo područje koje pokušava razotkriti unutrašnje mehanizme i načine na koji modeli obrađuju informacije. Takav pristup je ključan za poboljšanje pouzdanosti i sigurnosti modela pri donošenju bitnih odluka koje utječu na ljudske živote.

Korištenjem jednostavnog modela transformera za modularno zbrajanje možemo lakše razumjeti kako transformeri funkcioniraju i razotkriti unutrašnje algoritamske krugove koje koristi. Hibridizacijom modela možemo osigurati bolje performanse, sigurnost te pouzdanost model. Pod hibridizacijom u ovom radu se misli na zamjenu neuralnih dijelova ručno kodiranim komponentama.

Cilj ovog rada je analiza jednostavnog modela transformera za modularno zbrajanje koristeći mehanističku interpretabilnost. U 2. poglavlju objašnjena je arhitektura transformera, način rada, glavne komponente, njihova primjena te prednosti i nedostatci. U 3. poglavlju opisana je mehanistička interpretabilnost, obrasci mehanističke interpreta-

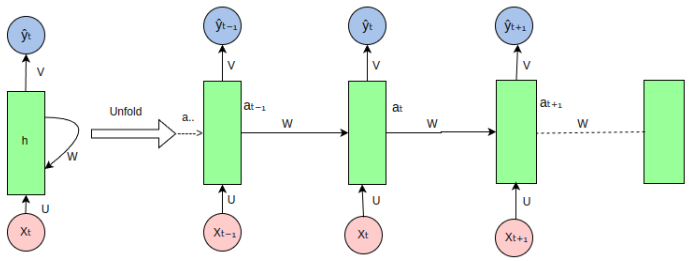
bilnosti, glavni koncepti te temeljene metode koje se koriste. U 4. poglavlju opisana je hibridizacija modela transformera. U 5. poglavlju opisan je model za modularno zbrajanje i algoritam koji koristi za računanje, prikazana je analiza modela i njegova hibridizacija.

2. Model transformera

2.1. Povijest i evolucija

Transformer je model dubokog učenja koji je po prvi put opisan 2017. godine u akademskom članku pod nazivom "Attention is All You Need" od strane znanstvenika iz Googlea i grupe sa sveučilišta u Torontu [2]. Objava članka smatra se povijesnim trenutkom s obzirom na njihovu raširenost danas [2]. Transformeri imaju primjenu u raznim područjima kao što su neurolingvističko programiranje, prevođenje jezika, analiza sekvenci DNK-a te analiza strukture proteina [4]. Prije pojave transformera, u području neurolingvističkog programiranja, modeliranje sekvenci se odvijalo uz pomoć RNN-a (engl. recurrent neural network) i LSTM-a (engl. Long short-term memory) [5].

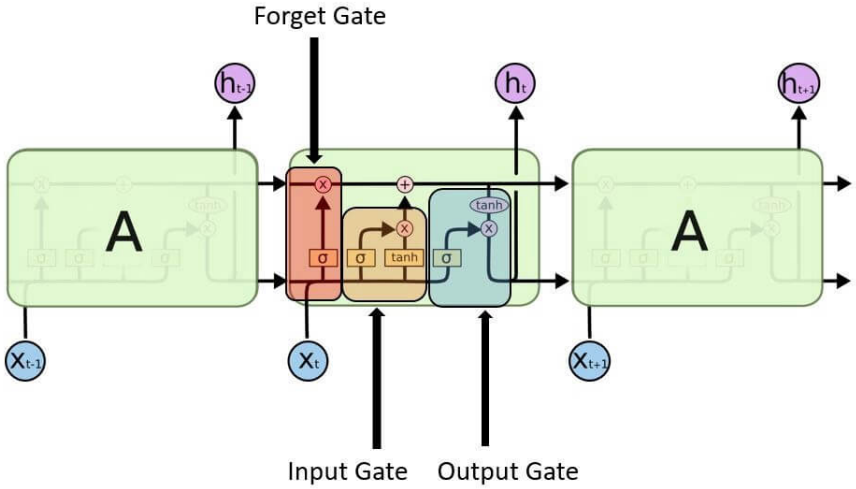
Povratna neuronska mreža (engl. recurrent neural network) je klasa neuronske mreže koja ima mogućnost rukovanja sekvencijalnim podatcima koristeći informacije iz prethodnih ulaza [5]. Uvedene su kako bi riješile ograničenja koje imaju tradicionalne neuronske mreže sa sekvencijalnim podatcima. Unaprijedne neuronske mreže procesiraju svaki ulaz zasebno, nezavisno o ostalim, kroz n skrivenih slojeva zbog čega nisu u stanju usvojiti veze među ulazima [6]. Umjesto procesiranja podataka u jednom prolazu, povratne neuronske mreže procesiraju podatke u više koraka [7].



Slika 2.1. Arhitektura povratne neuronske mreže [6]

Povratne neuronske mreže sadrže internu memoriju jer se izlaz svakog koraka koristi kao ulaz za sljedeći korak te se na taj način korisne informacije iz prethodnih koraka mogu iskoristiti za sljedeće korake [6]. Takva svojstva ih čine korisnim u modeliranju i obrađivanju teksta, govora i vremenskih serija [7]. Na slici 2.1 prikazana je arhitektura povratne neuronske mreže gdje X_t predstavlja ulazni niz, h skriveno stanje te $\hat{Y}_t$ izlazni niz, a U, V, W predstavljaju težine. S desne strane slike 2.1 prikazana je "odmotana" povratna mreža. Povratne mreže su efektivne u hvatanju kratkoročnih ovisnosti, ali imaju problem nestajućeg gradijenta gdje se utjecaj ranijih ulaza smanjuje eksponencijalno kako se ulazni niz obrađuje te je zbog toga teško uhvatiti dugoročne ovisnosti [5].

Dugotrajna kratkoročna memorija (engl. Long short-term memory, LSTM) je tip povratne neuronske mreže koja je nastala kako bi razriješila problem nestajućeg gradijenta, koji je veliki problem kod tradicionalnih RNN-a [8]. Koriste se za probleme vezane uz sekvencijalne podatke kao što su neurolingvističko programiranje, prepoznavanje govora te predviđanje vremenskih serija. LSTM mreže se sastoje od memorijskih ćelija, osnovnih gradivnih jedinica, koje sadrže ulazna vrata, zaboravna vrata te izlazna vrata [8]. Sva vrata imaju ulogu reguliranja toka informacija pri ulazu i izlazu u ćeliju [9].



Slika 2.2. Arhitektura dugotrajne kratkoročne memorije [10]

Ulazna vrata određuju količinu novog ulaza koja će biti spremljena u ćeliji. Zabo-

ravna vrata određuju koje informacije izbaciti iz ćelije dok izlazna vrata određuju koliki sadržaj ćelije iskoristiti za računanje skrivenog stanja [8]. Na slici 2.2. prikazana je arhitektura jedne ćelije. Iako rješavaju probleme koje imaju tradicionalne RNN mreže, LSTM mreže su kompleksnije za trening, zahtijevaju veću količinu računalne snage [11] te i dalje obrađuju podatke slijedno, a ne u paraleli [12].

Kod primjene RNN-a i LSTM-a na neurolingvističkim problemima, kao što su strojno prevođenje i sažimanje teksta, koristi se arhitektura model koder-dekoder [12]koja je detaljnije opisana u poglavlju 2.2.1. Problem kod ovakvog pristupa je taj što ne postoji nikakav mehanizam pažnje i vektor stanja je dostupan tek nakon zadnje riječi izvornog teksta [12]. Iako vektor teoretski zadržava informacije za cijelu izvornu rečenicu, u praksi se pokazalo da su informacije slabo sačuvane. Razlog za slabo očuvanje informacija je obrađivanje ulaza sekvencijalno u vektor fiksne veličine, koji je onda obrađen još jednom u izlaz [12]. U prije spomenutom akademskom članku "Attention is All You Need" predložen je model koji nema povratnu vezu te ima mogućnost obrađivanja svih tokena u paraleli i temeljen je na mehanizmima pažnje [12]. Taj model je transformer.

2.2. Arhitektura

2.2.1. Struktura koder-dekoder

Većina kompetitivnih modela za obrađivanje sekvenca u sekvencu zadataka imaju arhitekturu koder-dekoder [13]. Koder preslikava ulaznu sekvencu simbola $(x_1, ..., x_n)$ u sekvencu kontinuiranih reprezentacija $\boldsymbol{z} = (z_1, ..., z_n)$ te dekoder s dobivenim $\boldsymbol{z}$ generira izlaznu sekvencu $(y_1, ..., y_m)$ simbola, element po element [13].

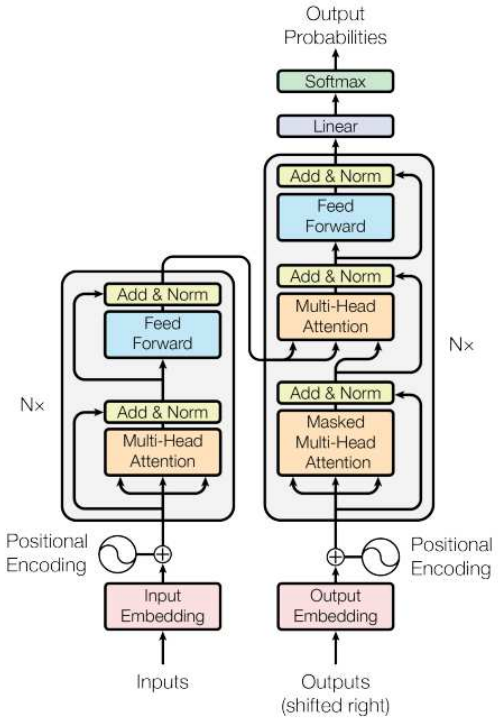
Koder se sastoji od ugradbenog sloja (engl. embedding layer) te n slojeva kodera. Svaki sloj kodera sadrži dvije glavne komponente: mehanizam samopažnje te unaprijednu neuronsku mrežu [12]. Rad kodera se može shematski prikazati na sljedeći način:

za zadane vektore $h_0, h_1, ...$

$$\text{stavi ih matricu } H = \begin{bmatrix} h_0 \\ h_1 \\ \vdots \end{bmatrix}$$

$$\text{EncoderLayer}(H) = \begin{bmatrix} \text{FFN}(\text{MultiheadedAttention}(H, H, H)_0) \\ \text{FFN}(\text{MultiheadedAttention}(H, H, H)_1) \\ \vdots \end{bmatrix}$$

Dekoder se sastoji od ugradbenog sloja (engl. embedding layer) , n slojeva dekodera te sloja povratnog preslikavanja (engl. un-embedding layer) . Svaki sloj dekodera se sastoji od uzročno maskiranog mehanizma samopažnje (engl. causally masked self-attention mechanism) , mehanizma međusobne pažnje (engl. cross-attention mechanism) te unaprijedne neuronske mreže [12].



Slika 2.3. Struktura koder-dekoder [13]

Shematski se može prikazati na sljedeći način:

$$H' = \text{MaskedMultiheadedAttention}(H, H, H)$$

$$\text{DecoderLayer}(H) = \text{FFN}(\text{MultiheadedAttention}(H', H^E, H^E))$$

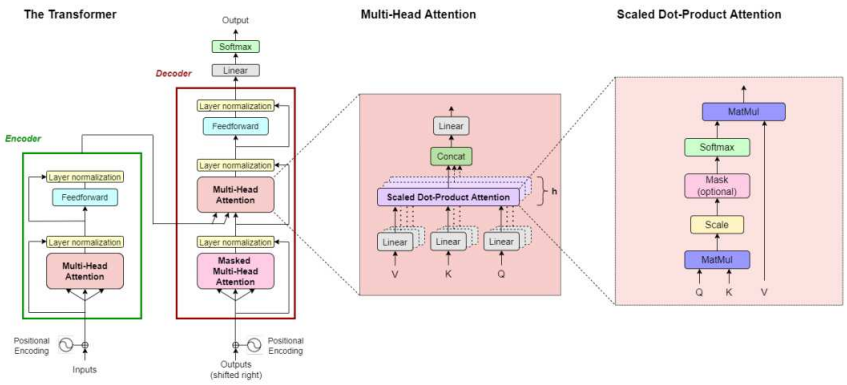
gdje je H^E matrica s redovima izlaza iz kodera.

2.2.2. Glavne komponente

Mehanizam pažnje (engl. attention mechanism)

Mehanizam pažnje je tehnika korištena u strojnom učenju te umjetnoj inteligenciji [14] koja omogućava modelima da se fokusiraju na relevantne dijelove ulaznih podataka. Ovakav pristup je ključan u problemima gdje je bitan kontekst, kao npr. u razumijevanju jezika ili prepoznavanju slika [15]. Kod neurolingvističkih problema to znači da se riječima u rečenici daju različite razine relevantnosti [16].

Transformeri koriste mehanizam samopažnje (engl. self attention mechanism) koji funkcionira na drukčiji način od mehanizama koji su korišteni prije 2017. godine, tj. prije izlaska akademskog članka "Attention is all you need" [17]. Takav mehanizam omogućuje modelu da težinski odredi relevantnost dijelova ulaza bez obzira na njegovu poziciju u sekvenci [15]. Mehanizam koristi tri vektora: upita, ključa i vrijednosti, za svaku riječ. Vektori se dobiju množenjem ulaza s matricama težina W_q, W_k i W_v koje se uče tijekom treninga modela. Svaka vrijednost (engl. value) je težinski određena funkcijom upita (engl. query) i ključa (engl. key), a izlaz se računa kao težinski zbroj vrijednosti [17]. U akademskom članku iz 2017. predložena su dva mehanizma za pažnju, pažnja temeljena na skalarnom produktu (engl. scaled dot-product attention) te višeglava pažnja (engl. multihead attention).



Slika 2.4. Arhitektura transformera s prikazanim mehanizmima pažnje [17]

Ulaz za pažnju temeljenu na skalarnom produktu su vektori upita i ključa dimenzije d_k te vektor vrijednosti dimenzije d_v. Pažnja se računa tako da se izračuna skalarni pro-

dukt upita i svih ključeva te se skalira tako što se podijeli s $\sqrt{d_k}$ i konačno se primjeni funkcija softmax kako bi se dobile težine za vrijednosti. U praksi se pažnja računa u isto vrijeme na setu upita tako što se svi upiti ubace u matricu Q. Također, kao i upiti, ključevi i vrijednosti su također ubačeni u matrice, respektivno K, V [17]. Na taj način izračun se može prikazati na kompaktniji način:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}})V \tag{2.1}$$

Na slici 2.4, skroz desno može se vidjeti slikoviti prikaz računanja, u jednoj jedinici.

Mehanizam višeglave pažnje linearno preslikava upite, ključeve i vrijednosti h puta s različitim, naučenim linearnim preslikavanjima u dimenzije d_q, d_k, d_v. Funkcija pažnje se odvija paralelno na svim preslikanim verzijama upita, ključeva i vrijednosti te se zato mehanizam zove višeglavi. Za konačni rezultat se izlazne vrijednosti dimenzije d_v konkateniraju te se još jednom preslikavaju [17]. Višeglava pažnja je vidljiva na slici 2.4 u sredini slike. Prikazuje se shematski na sljedeći način:

$$\begin{gathered} MultiHead(Q, K, V) = concat(head_1, ..., head_h)W^O \\ \text{gdje je} \quad head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \end{gathered} \tag{2.2}$$

Maskirana višeglava pažnja je mehanizam dekoderskog dijela transformera, koji se primjenjuje prvi kako bi samo riječi koje dolaze prije određene riječi imale utjecaja na izlaz. Vektorske reprezentacije koje dekoder ne bih trebao vidjeti su maskirane tako da ih se prije funkcije softmax postavi na $-\infty$ [17].

Pozicijsko kodiranje (engl. positional encoding)

Transformeri u svojoj arhitekturi nemaju povratnu vezu ni konvoluciju te se iz tog razloga na neki način trebaju ubaciti informacije o relativnoj ili apsolutnoj poziciji tokena u sekvenci [13]. Pozicijska kodiranja imaju istu dimenziju d_{model} kao i vektorske reprezentacije. Ako t predstavlja poziciju u sekvenci, $\vec{p}_t \in \mathbb{R}^d$ predstavlja vektorsku reprezentaciju, gdje je d dimenzija vektorske reprezentacije [18] pozicijska kodiranja se računaju

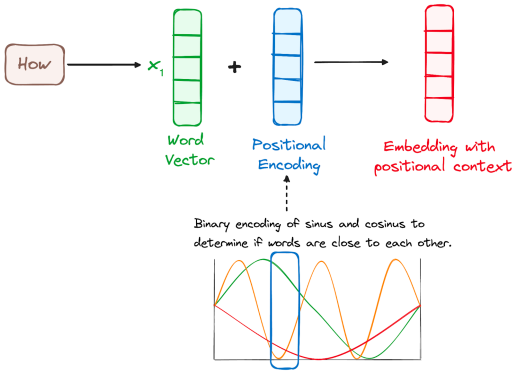
na sljedeći način:

$$\vec{p_t^{(i)}} = f(t)^{(i)} := \begin{cases} sin(\omega \cdot t) & \text{ako } i = 2k \\ cos(\omega \cdot t) & \text{ako } i = 2k+1 \end{cases} \tag{2.3}$$

gdje je:

$$\omega_k = \frac{1}{N^{2k/d}} \tag{2.4}$$

N je parametar koji bi trebao biti veći od najvećeg ulaza k u funkciju kodiranja. U izvornom članku korišten je N=10000 [12].



Slika 2.5. Vektorska reprezentacija s pozicijskim enkodiranjem [19]

Ako imamo rečenicu "Danas je lijep dan", pozicijska kodiranja se računaju na sljedeći način:

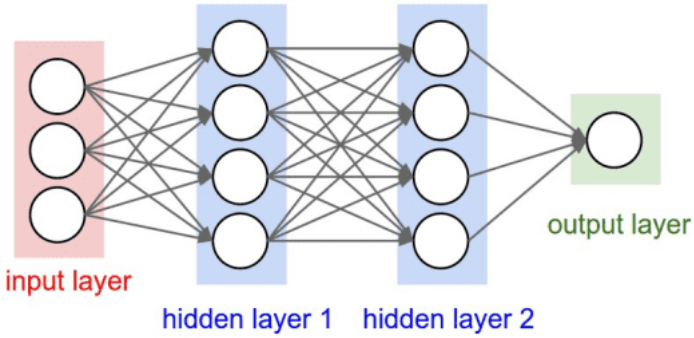
$$\underset{\text{Sekvenca}}{\begin{bmatrix} Danas \\ je \\ lijep \\ dan \end{bmatrix}} \rightarrow \underset{\text{Indeks u sekvenci}}{\begin{bmatrix} 0 \\ 1 \\ 2 \\ 3 \end{bmatrix}} \rightarrow \underset{\text{Pozicijska kodiranja za } d_{model}=4, n=10000}{\begin{bmatrix} sin(\frac{0}{1000^{2x0/4}}) & cos(\frac{0}{1000^{2x0/4}}) & sin(\frac{0}{1000^{2x1/4}}) & cos(\frac{0}{1000^{2x1/4}}) \\ sin(\frac{1}{1000^{2x0/4}}) & cos(\frac{1}{1000^{2x0/4}}) & sin(\frac{1}{1000^{2x1/4}}) & cos(\frac{1}{1000^{2x1/4}}) \\ sin(\frac{2}{1000^{2x0/4}}) & cos(\frac{2}{1000^{2x0/4}}) & sin(\frac{2}{1000^{2x1/4}}) & cos(\frac{2}{1000^{2x1/4}}) \\ sin(\frac{3}{1000^{2x0/4}}) & cos(\frac{3}{1000^{2x0/4}}) & sin(\frac{3}{1000^{2x1/4}}) & cos(\frac{3}{1000^{2x1/4}}) \end{bmatrix}} \tag{2.5}$$

Izračunata pozicijska kodiranja se nadodaju na vektorske reprezentacije na početku kao što je vidljivo na slici 2.5. Glavni razlog za računanje pozicijskog kodiranja na ovaj način je da su pomaci linearne transformacije. To omogućuje transformeru da uzme bilo koju kodiranu poziciju te pronađe vektorsku reprezentaciju koja je n-koraka ispred ili iza množenjem matrica [12]. Indeks riječi u sekvenci se ne koristi kao reprezentacija

pozicije jer kod duljih sekvenci brojevi mogu biti jako veliki, a normalizacijom indeksa između 0 i 1 problemi nastaju kada su sekvence varijabilne duljine [20].

Aciklička neuronska mreža (engl. feedforward neural network)

Aciklička neuronska mreža jedna je od najosnovnijih neuronskih mreža. Informacije se kreću unaprijed, u jednom smjeru, od ulaza kroz skrivene slojeve (ako postoje) do izlaza. [21]



Slika 2.6. Unaprijedna neuronska mreža s dva skrivena sloja [22]

Na slici 2.5 može se vidjeti primjer acikličke potpuno povezane neuronske mreže s dva skrivena sloja, gdje se svaki skriveni sloj sastoji od 4 neurona. Aciklička neuronska mreža se sastoji od tri tipa sloja: ulazni sloj, skriveni slojevi te izlazni sloj. Svaki sloj se sastoji od neurona te su slojevi međusobno povezani težinama [21]. Ulazni sloj sastoji se od neurona koji prosljeđuju ulaze do sljedećeg sloja. Broj neurona u ulaznom sloju ovisi o dimenziji podataka. Skriveni slojevi uzimaju težinski zbroj izlaza iz prethodnog sloja, primjenjuju aktivacijsku funkciju te prosljeđuju rezultat u sljedeći sloj [21]. Broj skrivenih slojeva i broj neurona u svakom sloju su hiperparametri modela te određuju kompleksnost i kapacitet [23]. Neuronska mreža može i ne mora imati skrivene slojeve. Izlazni sloj daje rezultat za dani ulaz te njegova dimenzija ovisi o broju izlaza koji mreža treba izdati [21].

Aktivacijske funkcije u skrivenim slojevima unose nelinearnost što omogućava modelu da nauči kompleksne uzorke [21]. Najčešće aktivacijske funkcije su tangens hiperbolni, logistička funkcija te ReLU (engl. rectified linear unit) funkcija [24].

Rad acikličkih neuronskih mreža sastoji se od dvije faze: propagacije unaprijed te propagacije unatrag. Kod propagacije unaprijed, podatci se propagiraju kroz mrežu, gdje

se kod svakog skrivenog sloja uzima težinski zbroj izlaza iz prethodnog sloja te se primjenjuje aktivacijska funkcija, sve dok se ne dođe do konačnog sloja, tj. izlaznog sloja. Na izlazu iz konačnog sloja mreža daje predikciju. U fazi propagacije unatrag računa se pogreška koju je model napravio usporedbom dobivenog i željenog izlaza. Greška se šalje unatrag kroz mrežu te se na taj način težine podešavaju kako bi se smanjila pogreška. Proces podešavanja težina se obično radi algoritmom gradijentog spusta [21].

2.3. Učenje i optimizacija

Transformeri se obično pretreniraju na zadatcima samonadziranog učenja te se nakon toga specijaliziraju nadziranim učenjem na manjem, specifičnom skupu podataka [12]. Zadatci za pretreniranje su obično modeliranje jezika, predviđanje sljedeće riječi u rečenici, odgovaranje na pitanja i slično.

2.3.1. Pretprocesiranje podataka

Tokenizacija i vektorska reprezentacija (engl. tokenization and embedding)

Tokenizacija je proces pretvaranja teksta u tokene. Token je prirodni broj koji predstavlja slovo ili segment slova. Modul zadužen za pretvaranje teksta u tokene naziva se tokenizator (engl. tokenizer) te se skup svih tokena u tokenizatoru naziva vokabular i označava s $n_{vocabulary}$. Za tokene koji se ne nalaze unutar vokabulara koristi se poseban token "[UNK]" (engl. unknown). [12].

Pretvaranje tokena u vektorske reprezentacije odvija se uz pomoć tablice pretraživanja. Ako postoji matrica M koja sadrži vektorske reprezentacije za sve tokene i svaki token ima one-hot reprezentaciju, vektorska reprezentacija se dobije množenjem vektora one-hot reprezentacije i matrice M. Na vektorsku reprezentaciju se nadodaje pozicijsko kodiranje [12].

2.3.2. Funkcija gubitka i optimizatori

Funkcija gubitka u modelima dubokog učenja mjeri koliko su točne predikcije uspoređujući ih sa željenim izlazom. Gubitak je numerička vrijednost koja predstavlja grešku

koja je nastala tijekom procesa učenja [25]. Funkcija gubitka ovisi o zadatku na kojem transformer uči. Najčešće funkcije gubitka su srednja kvadratna pogreška (engl. mean squared error), logaritamski gubitak (engl. cross entropy loss) te Huber gubitak (engl. Huber loss). [26].

Srednja kvadratna pogreška se obično koristi na regresijskim zadatcima te se računa na sljedeći način:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{2.6}$$

gdje Y_i predstavlja ciljanu vrijednost, a $\hat{Y}_i$ predviđenu vrijednost.

Logaritamski gubitak se koristi za klasifikacijske zadatke te mjeri razliku između stvarne distribucije i predviđene distribucije vjerojatnosti koju daje model. Računa se na sljedeći način:

$$L = -\sum_{i=1}^{N} y_i \; log(\hat{y}_i) \tag{2.7}$$

gdje je y_i stvarna oznaka (1 za ispravnu klasu, 0 za ostale), a $\hat{y}_i$ predviđena vjerojatnost za tu klasu.

Huber gubitak je funkcija gubitaka koja se koristi u regresijskim zadatcima. Kombinira najbolja svojstva funkcija srednje kvadratne pogreške (MSE) i srednje apsolutne pogreške (MAE) te se računa na sljedeći način [27]:

$$L_\delta(e) = \begin{cases} \frac{1}{2}e^2 & \text{for } |e| \le \delta \\ \delta \cdot (|e| - \frac{1}{2}\delta) & \text{for } |e| > \delta \end{cases} \tag{2.8}$$

gdje je $e = y - \hat{y}$, a δ parametar praga koji određuje točku na kojoj funkcija gubitka prelazi iz kvadratnog u linearni oblik.

Optimizatori koji se najčešće koriste u transformerima su optimizator Adam te stohastički gradijentni spust [2]. Stohastički gradijentni spust je optimizacijski algoritam koji procjenjuje grešku trenutnog stanja modela te ažurira težine modela koristeći propagaciju unatrag [28]. Optimizator Adam je ekstenzija algoritma stohastičkog gradijentnog spusta [29]. Stohastički gradijentni spust ima fiksnu stopu učenja (engl. learning rate) dok Adam ima adaptivnu stopu učenja temeljenu na momentima gradijenata (srednja

vrijednost i varijanca). Stopa učenja određuje koliko će se težine ažurirati [28]. U akademskom članku "Attention is all you need" autori su predložili korištenje zagrijavanja za stopu učenja. Stopa učenja treba linearno rasti od 0 do maksimalne vrijednosti prije nego što se počne ponovno smanjivati. Obično se preporučuje da se zagrijavanje stope učenja radi na 2% ukupnog broja koraka učenja [12].

2.4. Tipovi i primjene transformer modela

Transformeri imaju mnoge primjene u neurolingvističkom programiranju, no i u mnogim ostalim područjima kao što su analiza bioloških sekvenci, analiza strukture proteina, evaluacija pozicija u šahu i slično. Transformeri osiguravaju efikasnost u procesiranju informacija kod velikih jezičnih modela kao što su GPT-2, GPT-3, GPT-4, CLAUDE, BERT te ostali [12, 30]. Imaju ključnu ulogu jer osiguravaju bolju preciznost, brži proces učenja te imaju široku primjenu. Tipovi transformera se mogu podijeliti po arhitekturi i po načinu pretreniranja.

2.4.1. Podjela po arhitekturi

Transformer sa samo koderom (engl. encoder only transformer) koristi samo kodersku komponentu osnovne arhitekture koder-dekoder. Fokusira se na razumijevanje i kodiranje ulaznih sekvenci bez generiranja izlazne sekvence. Obično se koristi za klasifikaciju teksta, analizu sentimenta i detekciju anomalija [30]. Popularni primjer transformera sa samo koderom je BERT.

Transformer sa samo dekoderom (engl. decoder-only transformer) koristi samo dekoderski dio za generiranje teksta za ulaznu sekvencu. Obično se koristi za sumiranje teksta, generiranje teksta te za chatbotove [30]. Popularan primjer su GPT modeli.

Transformeri s koder-dekoder arhitekturom su se koriste za transdukcijske probleme gdje se ulazna sekvenca transformira u različitu sekvencu, npr. kod prijevoda u drugi jezik. Obično se koriste za sumiranje teksta, odgovaranje na pitanja te strojno prevođenje [30].

2.4.2. Podjela po zadatcima za pretreniranje

Maskirani jezični modeli (engl. masked language models, MLMs) su po arhitekturi obično transformeri sa samo koderom. Naučeni su za predviđanje maskirane riječi s obzirom na kontekst u kojem se riječ nalazi. Kao takvi, efektivni su u razumijevanju veza u jeziku te činjenica da su naučeni na velikim skupovima podataka ih čini versatilnim i izvrsnim alatom u raznim neurolingvističkim primjenama.

Autoregresivni modeli (engl. autoregressive models) obično imaju arhitekturu sa samo dekoderom. Korišteni za iterativno predviđanje sekvenci, predviđaju sljedeću riječ na temelju prethodne. Koriste se za generiranje teksta, chatbotove te strojno prevođenje.

Uvjetni transformer (engl. conditional transformer) osim ulazne sekvence uzima dodatne uvjete što ga čini specifičnijim. Koristi se kod strojnog prevođenja, sumiranja s uvjetima te prepoznavanju govora s uvjetima [30].

2.5. Prednosti i nedostatci

Transformeri su revolucionizirali područje neurolingvističkih programiranja svojom mogućnosti razumijevanja i generiranje teksta te ostalim zadatcima neurolingvističkih problema, no to ne znači da nemaju svoje nedostatke. Transformeri se ističu svojom mogućnosti paralelizacije, koja između ostalog ubrzava proces učenja. Nadalje, imaju mogućnost hvatanja dugoročnih zavisnosti u tekstu što im omogućava bolje razumijevanje konteksta te kao rezultat generiraju smisleniji tekst. Također su fleksibilni i skalabilni što ih čini versatilnim i korištenim u mnogo različitih domena. S obzirom na njihovu veličinu i kompleksnost, učenje zahtijeva veliku količinu računalnih resursa i vremena [31]. Nadalje zahtijevaju veliku količinu podataka za učenje te za njihovu primjenu. Također, razumijevanje unutrašnjih procesa, tj, zašto i kako model predviđa je poznati problem u strojnom učenju [32]. Područje koje pokušava shvatiti unutrašnji način rada modela strojnog učenja naziva se mehanistička interpretabilnost i objašnjena je detaljnije u poglavlju 3.

3. Mehanistička interpretabilnost

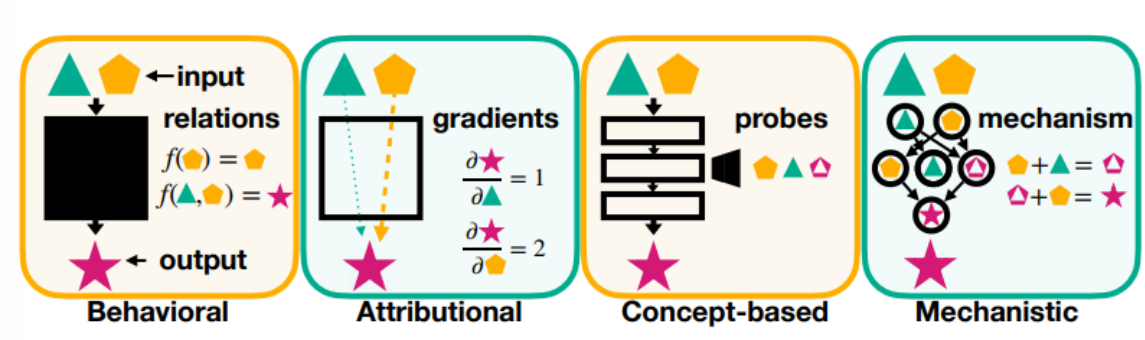
Mehanistička interpretabilnost je relativno novo područje znanosti koje proučava unutrašnje procese neuronskih mreža, tj. generalno modela strojnog učenja. [33]. Opisuje se kao obrnuto inženjerstvo računalnih mehanizama i reprezentacija naučenih u neuronskoj mreži u algoritme i koncepte koji su poznati čovjeku [34]. Iako današnji modeli daju izvrsne rezultate, kako oni funkcioniraju ostaje nepoznato te ih se često opisuje kao crne kutije. Osim što imamo mogućnost vidjeti ulaz i izlaz u model te imamo mogućnost analizirati izlaz, kako je model došao do tog izlaza nam ostaje nepoznanica .Razlog za kompleksnost današnjih modela kao što su npr. GPT-4 ili BERT su mnoštvo slojeva i mnoštvo parametara, preko milijardu te ponekad i preko trilijun. Danas, sa sve bržim tehnološkim razvojem, umjetna inteligencija je integrirana u mnoštvo procesa. Sve više se koriste za donošenje odluka te je uskoro moguća njihova integracija u zdravstvene, pravosudne, financijske i mnoge druge sustave [33]. Poznavanje kako model donosi odluke je bitno u slučaju kada model utječe na živote velikog broja ljudi. Osim toga, lakše bi ispravili greške koje model radi.

3.1. Obrasci interpretabilnosti

Kada ispitujemo ponašanje modela strojnog učenja, postoje razni obrasci interpretabilnosti kao što su ponašajni, atribucijski, mehanistički te obrazac temeljen na konceptu.

Ponašajna interpretabilnost (engl. behavioral interpretability) na modele gleda kao crnu kutiju te analizira vezu između izlaza i ulaza. Neke od tehnika korištene su analiza minimalnog para (engl. minimal pair analysis), osjetljivost i analiza uznemiravanja (engl. sensitivity and perturbation analysis). Ovakav pristup je praktičan za kompleksne ili posjedničke modele.

Atribucijska interpretabilnost pokušava objasniti izlaze modela praćenjem predikcija do pojedinačnih doprinosa ulaza korištenjem gradijenta. Izvorni gradijenti mogu biti diskontinuirani ili osjetljivi na promjene pa su neke od tehnika koje se koriste integrirani gradijenti, srednja vrijednost gradijenta propagacija po slojevima na osnovu relevantnosti. Omogućava transparentnost jer pokazuje utjecaj ulazne značajke bez poznavanja unutrašnjeg načina rada modela [34].



Slika 3.1. Obrasci interpretabilnosti [34]

Obrazac temeljen na konceptu nastoji shvatiti način rada ispitivanjem naučenih reprezentacija radi pronalaženja visokih koncepata i obrazaca koji upravljaju ponašanjem. Neke od tehnika koje se koriste su učenje nadziranih umjetnih klasifikatora, korištenje nenadziranih kontrastnih i strukturnih ispitivanja.

Mehanistička interpretabilnost je pristup koji proučava temeljne komponente modela kroz detaljnu analizu značajki, neurona, slojeva i veza. Mehanistička interpretabilnost daje detaljan pogleda u mehanizme koje model koristi za računanje. Za razliku od obrasca temeljenog na konceptu, pokušava otkriti uzročne veze i točne izračune koji pretvaraju ulaz u izlaz, obično kroz otkrivanje neuronskih krugova. U ovome radu detaljnije je opisana samo mehanistička interpretabilnost. [34]

Na slici 3.1 prikazani su svi tipovi interpretabilnosti i osnovni koncept na kojem se temelje.

3.2. Glavni koncepti

3.2.1. Značajke kao osnovna jedinica

Značajke su osnovne jedinice reprezentacija neuronskih mreža. Označavamo ih kao osnovne jedinice jer ih ne možemo daljnje rastaviti na jednostavnije, različite faktore. U tradicionalnom pristupu strojnom učenju, značajke su karakteristike ili atributi koji su izvedeni iz ulaznih podataka. U kompleksnijim sustavima značajke mogu nastati kao reprezentacijski uzorci unutar modela, čak i kod obrađivanja podataka nevezanih za ulaz. Zato se značajke predstavljaju kao bilo koje mjerljivo svojstvo ili karakteristika fenomena. Ključno svojstvo značajki je da se ne mogu pojednostavniti, tj. ne mogu se prikazati kao kombinacija jednostavnijih nezavisnih faktora. Značajke ne moraju predstavljati koncepte koji su razumljivi čovjeku. Kako današnji model nadmašuju ljudske mogućnosti, naučene značajke mogu postati apstraktne, kodirajući informacije koje se protive ljudskoj intuiciji. Ovakav pristup je sveobuhvatan i otporniji na buduće pristupe [34].

3.2.2. Značajke neuronskih mreža

U neuronskim mrežama, neuroni se smatraju računskim jedinicama, potencijalno značajkama.

Neuron koji odgovara jednom semantičkom konceptu naziva se monosemantički neuron (engl. monosemantic neuron). Analizom koji ulaz aktivira određeni neuron otkrivamo njegovo semantičko značenje. Da su neuroni reprezentacijski primitivi neuronskih mreža svi neuroni bi bili monosemantički i potpuna interpretabilnost bila bi jednako izvediva kao i karakterizacija svih neurona i njihovih veza. Obično to nije slučaj te su neuroni polisemantički (engl. polysemantic), tj. vezani su za više nepovezanih koncepata. Polisemantičnost je razlog zašto se neuroni ne smatraju osnovnim jedinicama neuronskih mreža i zbog toga je izazovno razumjeti kako neuronske mreže obrađuju informacije.

Hipoteza superpozicije predlaže da neuronske mreže imaju sposobnost upravljanja visoko dimenzionalnim prostorima kako bi prikazali više značajka nego što je neurona tako da kodiraju značajke u skoro-ortogonalnim smjerovima. Neortogonalnost znači

da se značajke međusobno ometaju, no prikazivanje više značajki nego što je neurona obično nadilazi trošak ometanja. Neuronske mreže superpozicijom mogu simulirati izračune s više neurona nego što postoji u neuronskoj mreži tako da značajke prikazuju kao linearnu kombinaciju više neurona. Ovakav pristup predlaže da se polisemantički modeli mogu gledati kao kompresirane verzije većih neuronskih mreža gdje svaki neuron predstavlja jedan koncept. Učenje modela bez superpozicije koristeći softmax funkciju kao aktivacijsku funkciju da bi povećali broj monosemantičkih neurona te učenje mreže na standardan način i korištenje analize nakon učenja neki su od načina razumijevanja neuronskih mreža bez obzira na polisemantičnost [34].

Značajke u neuronskim mrežama su raspetljani koncepti koje bi veća, rjeđa mreža s dovoljnim kapacitetom mogla predstaviti pojedinačnim neuronima.

Druga hipoteza predlaže da su značajke smjerovi u aktivacijskom prostoru, tj. linearne kombinacije neurona. Akademski članak autora Engels i drugi [35] pružaju dokaze koji se protive toj hipotezi pronalaženjem kružnih značajki koje predstavljaju dane u tjednu i mjesece u godini. Takve nelinearne reprezentacije su korištene za rješavanje problema modularne aritmetike za mjesece i dane.

U akademskom članku Li i drugi [36] su zaključili da u GPT modelu naučenom na Othello igri, stanje ploče je moguće dekodirati samo sa nelinearnim ispitivanjem kada je predstavljeno crnim i bijelim figurama. Ta pretpostavka je opovrgnuta, Nanda [37] je dokazao da je linearno ispitivanje dovoljno kada su figure predstavljene kao protivničke i svoje. To je potvrdilo hipotezu linearne reprezentacije, no u akademskom članku Engels i drugi [35] pružaju dokaze o postojanju nelinearnih, višedimenzionalnih reprezentacija u jezičnim modelima.

Dok je hipoteza linearnih reprezentacija korisno pojednostavljenje, ono ima svoja ograničenja. Bitno je uočiti mogućnosti koje nelinearne reprezentacije mogu donijeti.

3.2.3. Algoritamski krugovi (engl. circuits) kao računski primitivi

Neuronske mreže se mogu opisati kao računski grafovi, unutar kojih su algoritamski krugovi podgrafovi koji se sastoje od vezanih značajki i težina koje ih vežu. Kao što su

značajke reprezentacijski primitivi, algoritamski krugovi se smatraju računskim primitivima i smatraju se osnovnim gradivnim jedinicama mreža. Dekompozicija neuronskih mreža u algoritamske krugove se pokazala kao obećavajući pristup, pogotovo u manjim modelima koji su naučeni za specifične zadatke.

Motivi su uzorci koji se ponavljaju u mrežama te mogu predstavljati značajke ili algoritamske krugove koji se pronalaze u različitim modelima i zadatcima. Neki od motiva su detektori krivulja u modelima za obradu vizualnih podataka, indukcijski krugovi koji omogućavaju učenje s kontekstom te fenomen specijalizacija grana u neuronskim mrežama. Motivi su značajni u otkrivanju zajedničkih struktura, mehanizama i strategija koje se pojavljuju u modelima.

Slaba univerzalnost je hipoteza koja predlaže konvergenciju modela na sličnim zadatcima, no značajke i algoritamski krugovi se razlikuju od modela do modela zbog raznih hiperparametara, slučajnih brojeva i arhitekturalnih različitosti.

Jaka univerzalnost predlaže da će u svim mrežama naučenim na sličnim zadatcima biti otkriveni isti algoritamski krugovi i značajke.

Iako mnogo radova podržava hipotezu univerzalnosti, dokazi za jaku univerzalnost nisu pronađeni dok za slabu univerzalnost jesu.

3.3. Temeljne metode

Mehanistička interpretabilnost se fokusira na uzročne metode. Metode se mogu podijeliti u dvije skupine: opažačke i intervencijske.

3.3.1. Opažačke metode

Opažačke metode se dalje mogu podijeliti u dvije kategorije: metode temeljene na primjerima i metode temeljene na značajkama.

Metode temeljene na primjerima prepoznavaju ulaze koji najviše aktiviraju određene neurone ili slojeve.

Metode temeljene na značajkama koriste tehnike sintetičkih ulaza kako bi optimizi-

rali aktivaciju neurona.

Ispitivanje značajki uključuje treniranje klasifikatora koristeći aktivacije modela, pri čemu se naknadno promatra učinkovitost klasifikatora kako bi naučili o ponašanju modela i njegovim unutarnjim reprezentacijama. Problem kod ovakvog pristupa je da ponekad učinkovitost ispitivanja ovisi o mogućnostima ispitivača. Hipoteza linearne reprezentacije nam govori da ako jednostavno linearno ispitivanje ne otkriva nikakve značajke, model ne sadrži linearne značajke. Veliki nedostatak ispitivanja je nedostatak mogućnosti donošenja ponašajnih ili uzročnih zaključaka. Ispitivanje se koristi u analizi razumijevanja znanja šaha u AlphaZero te reprezentacije lingvističkih informacija u BERT modelu.

Strukturirana ispitivanja pridonose razumijevanju koncepta tako što ispitivaju jezične modele za kompleksne značajke.

Logitna perspektiva (engl. logit lens) daje uvid u način predikcije modela. Omogućuje uvid u prediktivni proces modela primjenom završnog sloja za klasifikaciju (koji preslikava aktivaciju rezidualnog toka u prostor logita/rječnika) na međufazne aktivacije rezidualnog toka, otkrivajući kako se povjerenje u predikciju razvija kroz različite faze obrade informacija.

Razdvajanje značajki kroz učenje s rijetkim rječnikom (engl. Feature Disentanglement via Sparse Dictionary Learning) temelji se na činjenici da su ključni elementi u neuronskim mrežama linearne kombinacije neurona koje predstavljaju značajke u superpoziciji. Rijetki autokoderi (engl. sparse autoencoders) pružaju način dekompozicije aktivacija neuronskih mreža u individualne komponente značajki. Ovakav proces uključuje rekonstrukciju aktivacijskih vektora kao rijetke linearne kombinacije vektora smjera u aktivacijskom procesu. Učenje s rijetkim rječnikom je doprinijelo razvoju raznih algoritama za rijetko programiranje. Među njima se ističe rijetki autokoder (engl. sparse autoencoder) svojom jednostavnosti i skalabilnosti. Rijetki autokoder je varijanta standardnog autokodera koji koristi regularizaciju rijetkosti (engl. sparsity regularization) kako bi naučio rijetke, no značajne reprezentacije u podatcima. Pokazalo se da rijetki autokoderi mogu povećati interpretabilnost neuronskih mreža. Koriste se za mjerenje rijetkosti značajki te za razumijevanje modela nagrade u podržanom učenju.

3.3.2. Intervencijske metode

Tretiranjem neuronskih mreža kao uzročnih modela, dobivamo mogućnost intervencije i ispitivanja uloga pojedinih parametara. Takav pristup je pridonio razvoju intervencijskih metoda kao što su zakrpavanje aktivacija (engl. activation patching), uzročna apstrakcija (engl. causal abstraction) te metode testiranja hipoteze.

Zakrpavanje aktivacija se odnosi na skup uzročnih intervencija koje manipuliraju aktivacije mreže kako bi objasnile način na koji model donosi odluke. Neke od tehnika su uzročno praćenje (engl. causal tracing), intervencija izmjene (engl. interchange intervention), analiza uzročnog posredovanja (engl. causal mediation analysis) te uzročno izostavljanje (engl. causal ablation). Zajedničko svojstvo svih tehnika je izmjena unutrašnjeg stanja modela zamjenom aktivacija alternativnim vrijednostima, nulom, srednjom vrijednosti aktivacije, šumom ili aktivacijom iz drugog unaprijednog prolaza. Glavni cilj zakrpavanja aktivacija je izolirati i razumjeti ulogu koju komponenta ili algoritamski krug ima u mijenjanju aktivacija te kako to utječe na izlaz modela. Tipično se proces zakrpavanja aktivacija provodi na sljedeći način:

- pokrenuti model s čistim ulazom i spremiti aktivacije
- pokrenuti model s "pokvarenim" ulazom
- pokrenuti model s "pokvarenim" ulazom uz zamjenu pojedinih aktivacija sa spremljenim "čistim" aktivacijama
- promatrati izlaze u trećem koraku koji ukazuju na značajnost zamijenjene komponente

Zakrpavanje aktivacija ima dosta nedostataka, kao što su ručno modeliranje skupova podataka, potreba za ljudskom intervencijom za izolaciju bitnih podgrafova te komplikacije pri interpretaciji rezultata.

Uzročna apstrakcija pruža matematičku bazu tretiranjem neuronskih mreža i njihovih objašnjena kao uzročnih modela. Primjenu pronalaze u analizi lingvističkih fenomena, vrednovanja metoda interpretabilnosti, poboljšanja performansi podešavanjem reprezentacija te poboljšanje efikasnosti destilacijom modela.

Kako bi formalizirali i omogućilo empiričko vrednovanje ponašanja neuronskih mreža razvijene su metode uzročnog struganja (engl. causal scrubbing) i lokalno konzistentnih apstrakcija. Uzročno struganje formalizira hipotezu kao trojku (G, I, c) gdje G predstavlja računski graf, I predstavlja interpretabilni računski graf koji pod pretpostavkom objašnjava ponašanje te c koji preslikava čvorove grafa I u graf G. Metoda zamjenjuje aktivacije u G s pretpostavljenim ekvivalentima te mjeri performanse. Metoda lokalno konzistentnih apstrakcija provjerava konzistentnost između neuronske mreže i objašnjenja korak do intervencijskog čvora.

Često se kombiniraju promatračke i intervencijske metode[34].

4. Hibridizacija

Hibridizacija u strojnom učenju predstavlja kombiniranje dvaju modela, algoritma ili tehnika kako bi poboljšali performanse, povećali efikasnost, pouzdanost i sigurnost [38]. Ideja hibridizacije je kombiniranje algoritama kako bi se međusobno nadopunjavali te imali mogućnost rješavanja problema kompleksnih problema koje samostalno ne mogu riješiti [39].

Neke metode mogu rukovati s podatcima gdje postoji puno šuma dok mogu biti jako loše pri obrađivanju podataka u visokodimenzionalnim prostorima. Neke metode mogu izvrsno obrađivati podatke u visokodimenzionalnim prostorima no imati problema kada su podatci rijetki. Kombiniranjem metoda, slabosti jednog modela su nadjačane drugim modelom. Postoji ogroman broj načina kombiniranja modela.

Hibridizacija temeljena na arhitekturi je kombinacija dva ili više poznata algoritma, dijelom ili u cijelosti kako bi stvorili otporniji model. Popularan primjer je sustav ANFIS, koji kombinira principe neizrazite logike i neuronske mreže.

Hibridizacija temeljena na manipulaciji podataka kombinira procese manipulacije podataka s metodama strojnog učenja. Primjer hibridizacije na ovaj način je korištenje PCA modula za izvlačenje podmatrice podataka koja je dovoljna za objašnjavanje skupa podataka te primjena neuronske mreže nad tim podatcima.

Hibridizacija temeljena na optimizaciji parametara modela nastoji zamijeniti metode optimizacije korištenjem naprednih metoda temeljenih na evolucijskim algoritmima. Primjer je kombiniranje genetičkog algoritma za optimiziranje parametara ANFIS metode, koja na taj način postaje GANFIS model [40].

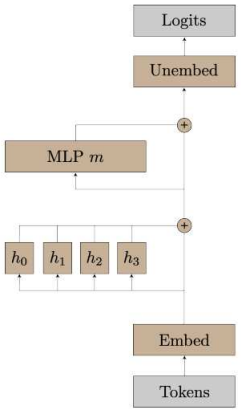
U ovom radu, pod hibridizacijom smatramo zamjenu neuralnog dijela transformera

ručno kodiranim komponentama i taj dio je prikazan u poglavlju 5.5.

5. Analiza i hibridizacija modela transformera

5.1. Opis modela

U ovom radu korišten je model transformera za modularno zbrajanje, $x+y \bmod P$, $x, y \in \mathbb{N}, x, y < P, P = 113$. Transformer se sastoji od jednog sloja u kojem se nalazi 4 glave pažnje (engl. attention heads) te acikličke potpuno povezane neuronske mreže s jednim skrivenim slojem. Vektorske reprezentacije tokena su dimenzije $d = 128$, svaka glava pažnje dimenzije $d = 32$ te skriveni sloj dimenzije $n = 512$. Na slici 5.1 prikazana je arhitektura modela.



Slika 5.1. Arhitektura transformera za modularno zbrajanje [41]

Ulaz u model je sekvenca oblika "a b =" gdje su a i b kodirani kao one-hot vektori dimenzije P. Token "=" je poseban token iznad kojega čitamo izlaz c. U modelu nije korištena normalizacija slojeva (engl. layer normalisation).

5.2. Učenje i optimizator

Skup podataka je skup svih uređenih parova brojeva modulo P, tj. skup čine 113x113 mogućih parova. Skup podataka podijeljen na način da je 70% skupa korišteno za učenje, a ostalih 30% korišteno za testiranje. Kao optimizator korišten je algoritam AdamW sa stopom učenja $\gamma = 0.001$ i regularizator težina $\lambda = 1$. Transformeri su učeni u 50000 epoha.

5.3. Fourierov algoritam množenja

U izvornom akademskom članku "Progress measures for grokking via mechanistic interpretability" autori tvrde da naučene mreže koriste sljedeći algoritam za računanje:

- Dva zadana, one-hot kodirana tokena a i b mapiraj u: $sin(w_k a),\ cos(w_k a),\ sin(w_k b)$ i $sin(w_k b)$ koristeći matricu kodiranja, gdje je $w_k = \frac{2k\pi}{P}, k \in \mathbb{N}$

- Izračunaj $cos\ (w_k\ (a+b))$ i $sin\ (w_k\ (a+b))$ koristeći trigonometrijske identitete:

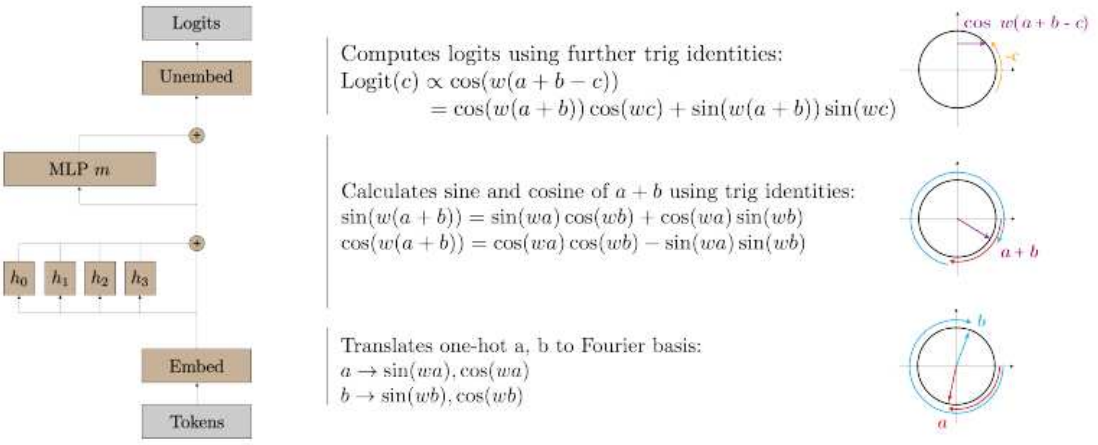
$$\begin{aligned} cos(w_k(a+b)) &= cos(w_k a)\ cos(w_k a) - sin(w_k a)\ sin(w_k b) \\ sin(w_k(a+b)) &= sin(w_k a)\ cos(w_k a) + cos(w_k a)\ sin(w_k b) \end{aligned} \tag{5.1}$$

- Za svaki izlazni logit c, izračunaj $cos(w_k(a+b-c))$ koristeći trigonometrijski identitet:

$$cos(w_k(a+b-c)) = cos(w_k(a+b))\ cos(w_k c) + sin(w_k(a+b))\ sin(w_k c) \tag{5.2}$$

  Ovo je linearna funkcija već izračunatih vrijednosti $cos(w_k(a+b))$ i $sin(w_k(a+b))$ te je implementirana u produktu izlazne matrice i matrice dekodiranja.

- Matrica dekodiranja također zbraja $cos(w_k(a+b-c))$ za različite k-ove. Na taj način kosinusoide konstruktivno interferiraju u $c^* \equiv a+b \mod p$ što rezultira velikim logitom c^*, dok kosinusoide destruktivno interferiraju u ostalim slučajevima(što rezultira malim logitom c)



Slika 5.2. Fourierov algoritam množenja [41]

Na slici 5.2 prikazan je rad modela. U sredini prikazan je Fourierov algoritam množenja te je usporedno prikazano i koji dio modela izvodi koji dio Fourierovog algoritma množenja. S desne strane slike je također prikazan algoritam u drukčijem obliku. Kako bi opravdali svoju hipotezu autori su predstavili 4 dokaza:

- Prvi dokaz je da su analizom težina mreže i aktivacija uočili periodičke strukture koje su rijetko slučajna pojava. Također, Fourierovim transformacijama, uočili su da su mnoge komponente rijetke ili gotovo rijetke u Fourierovoj domeni te su podržane od strane par ključnih frekvencija.

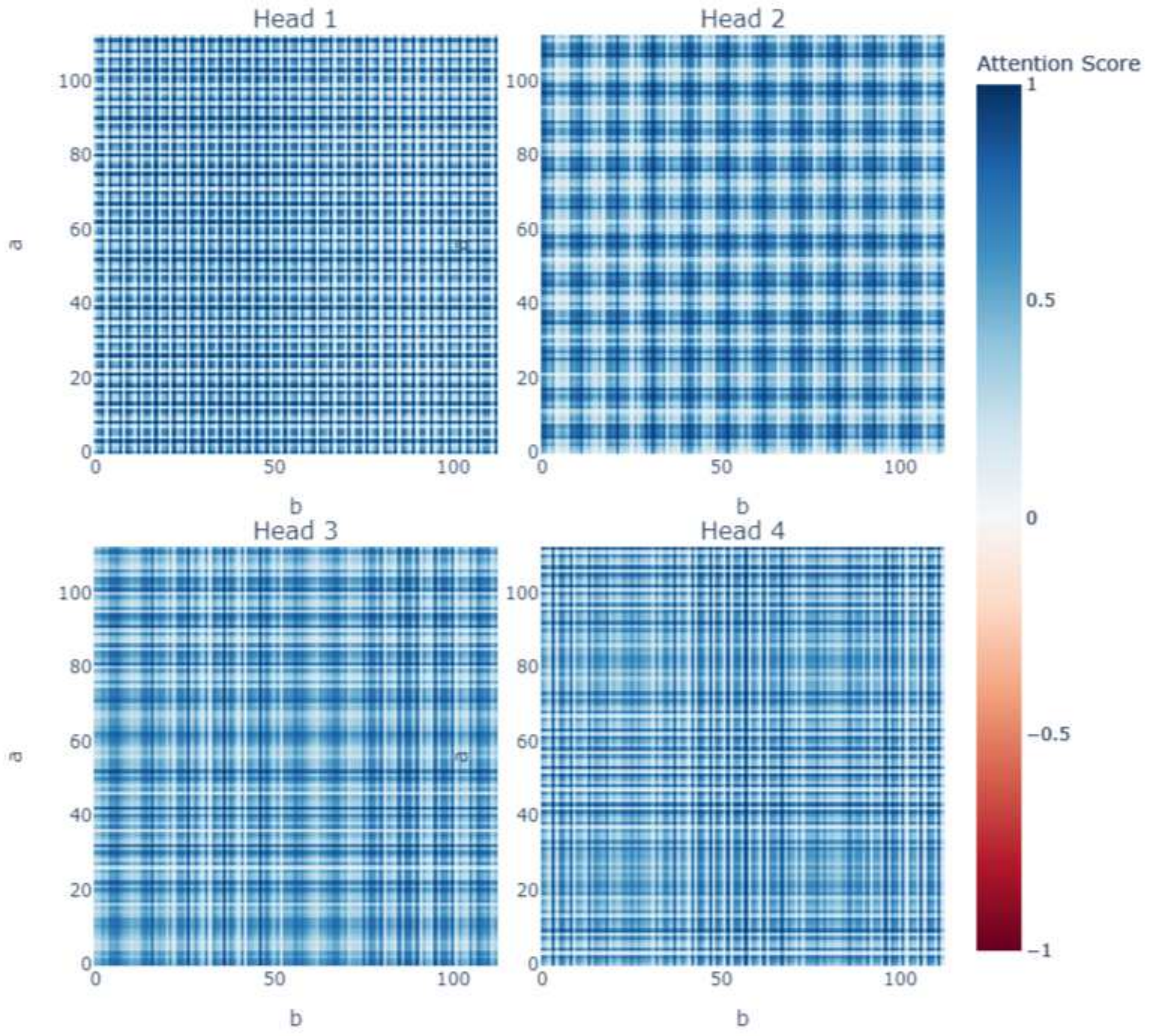
- Drugi dokaz je da je matrica dekodiranja otprilike ranga 10, gdje svaki smjer određuje sinus ili kosinus jedne od 5 ključnih frekvencija. Preslikavanjem aktivacija iz potpuno povezane neuronske mreže na komponente W_L približno stvara višekratnike funkcija $cos(w_k(a+b))$ i $sin(w_k(a+b))$ što pokazuje da MLP sloj izračunava te zbrojeve.

- Kao treći dokaz su pokazali da se većina glava pažnje i neurona može aproksimirati polinomima drugog stupnja od sinusa i kosinusa jedne frekvencije. Nadalje, odgovarajući smjer u matrici W_L također sadrži samo tu frekvenciju.

- Konačno,uklanjanjem određenih komponenti te njihovom izmjenom s Fourierovim komponentama, pokazalo se da se performanse modela u većini slučajeva poboljšavaju [41].

5.4. Analiza modela

Pogledom u aktivacije glava pažnje možemo uočiti periodičke strukture u svakoj od 4 glava pažnje.

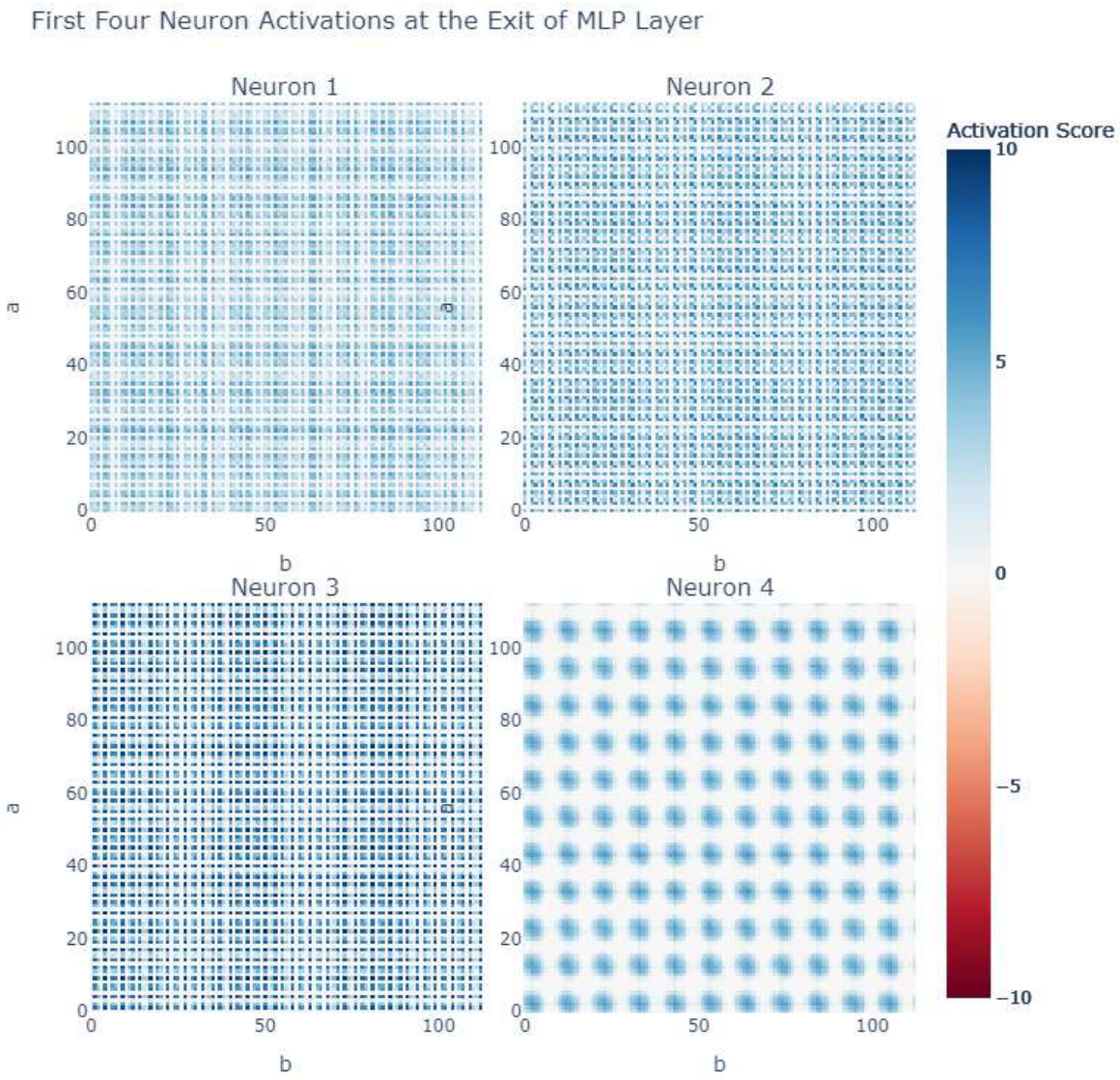


Slika 5.3. Aktivacije pažnje u svakoj od 4 glave [41]

Na slici 5.3 prikazani su periodički uzorci. Iako se razlikuju, svi predstavljaju periodičke strukture. Prikazani su rezultati pažnje od tokena "=" do "a" kao funkcija ulaza a i b. U izvornom radu autori su dokazali da se ovakav uzorak može aproksimirati sljedećim izrazom:

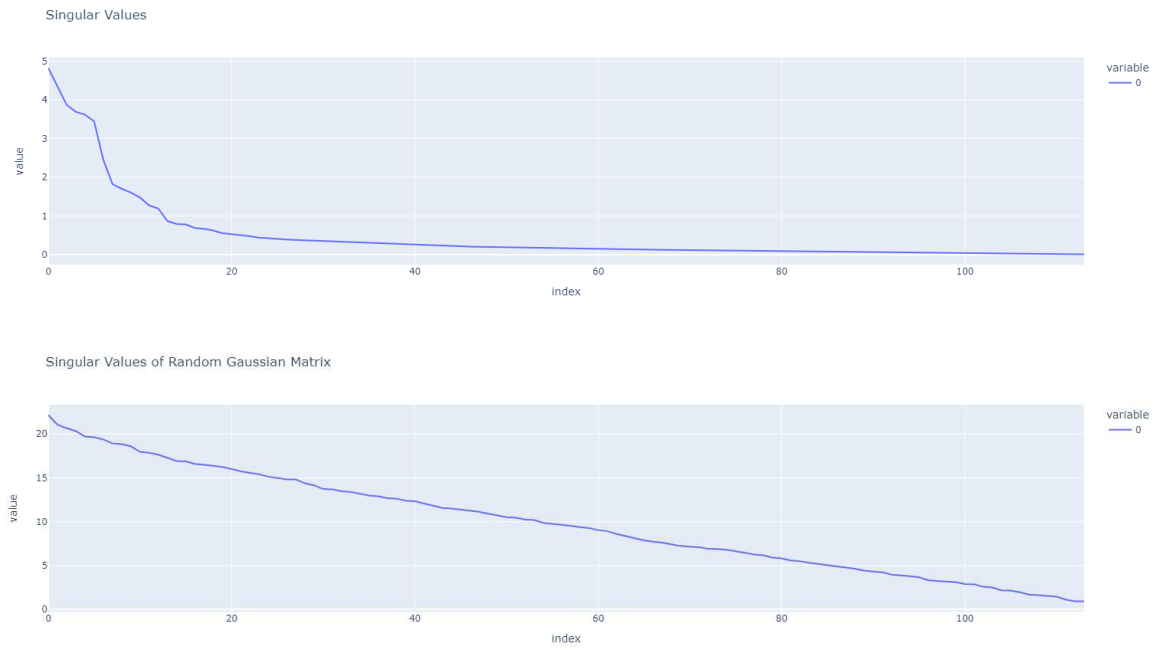
$$0.5 + \alpha(cos(w_k a) - cos(w_k b)) + \beta(sin(w_k a) - sin(w_k b)) \tag{5.3}$$

Također, kao i kod glava pažnje, ako pogledamo aktivacije prvih četiri neurona možemo uočiti periodičke uzorke.



Slika 5.4. Aktivacije prvih četiri neurona

Na slici 5.4 prikazane su periodičke strukture. Nadalje, ako uzmemo matricu vektorskih reprezentacija (engl. embedding matrix) i napravimo rastav na singularne vrijednosti možemo uočiti neke zanimljivosti. Uz matricu vektorskih reprezentacija usporedit ćemo i matricu istih dimenzija koja se sastoji od slučajno odabranih brojeva kako bi napravili razliku.

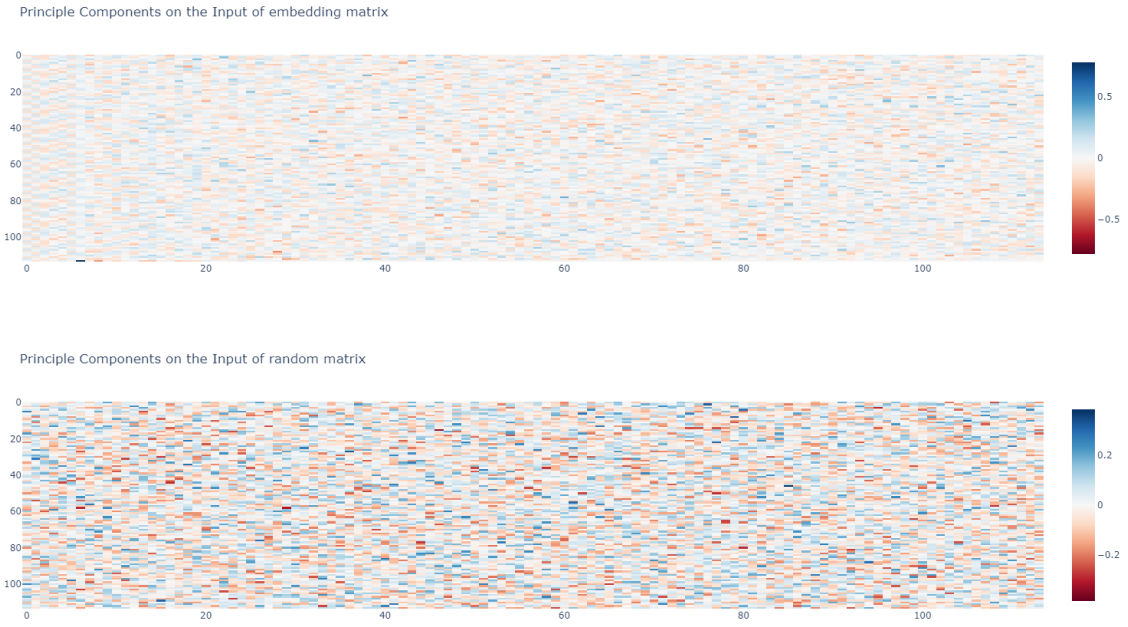


Slika 5.5. Usporedba rastava na singularne vrijednosti matrice vektorskih reprezentacija (gornja slika) i matrice slučajnih vrijednosti (donja slika)

Na slici 5.5 (gore) možemo uočiti da matrica vektorskih reprezentacija ima otprilike 10 smjerova koji su bitni od ukupno 128 smjerova. To možemo uočiti padajućom krivljom koja se brzo približava nuli. Kod matrice istih dimenzija sa slučajnim vrijednostima možemo uočiti do to nije slučaj. Gornja slika nam govori da kad ulaz pretvaramo u vektorske reprezentacije, izlaz je kombinacija 10 vektora koji odgovaraju vrijednostima s gornje slike dok ostatak nije bitan.

Pogledom na glavne komponente matrice vektorskih reprezentacija također možemo uočiti periodičke strukture. Za usporedbu su također korištene glavne komponente za matricu sa slučajnim vrijednostima.

Na slici 5.6 možemo vidjeti da slika gore ima periodičku strukturu u prvih 10 dimenzija, dok kod matrice sa slučajnim vrijednostima ne možemo lako uočiti nikakvu periodičku strukturu.

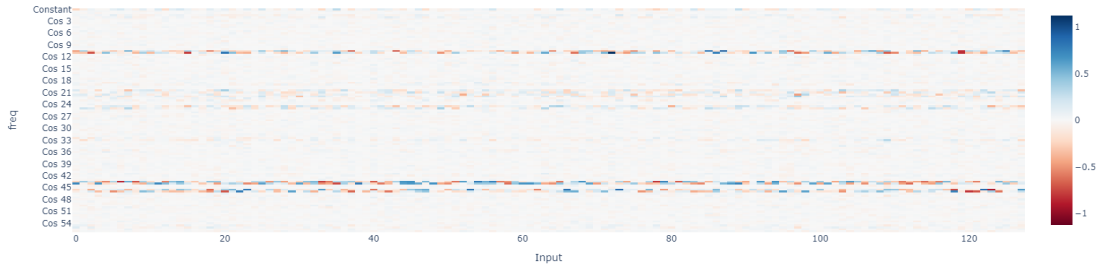


Slika 5.6. Usporedba glavnih komponenta matrice vektorskih reprezentacija(gore) i matrice sa slučajnim vrijednostima (dolje)

Kako bi dobili detaljniji prikaz periodičkih struktura, na slici 5.7 prikazan je približen pogled na prvih 10 dimenzija matrica.

Slika 5.7. Približen pogled na prvih 10 dimenzija glavnih komponenta matrice vektorskih reprezentacija (gore) i matrice sa slučajnim vrijednostima (dolje)

Množenjem vektorskih reprezentacija s Fourierovom bazom, dobivamo smjerove koji odgovaraju funkcijama sinusa i kosinusa u različitim frekvencijama.

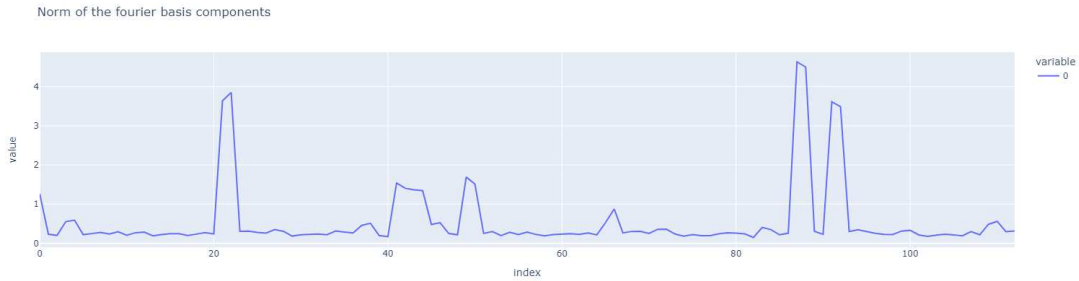


Slika 5.8. Ključne frekvencije mreže

"Linije" koje su vidljivije na slici 5.8 su bitnije modelu. Možemo reći da je model naučio tablicu preslikavanja te da mu je samo nekoliko frekvencija važno. Dio gdje model koristi tablicu preslikavanja pripisujemo početnom dijelu algoritma, gdje se one-hot vektori a i b preslikavaju u Fourierovu bazu:

$$\begin{aligned} a &\rightarrow sin(\omega a),\ cos(\omega a) \\ b &\rightarrow sin(\omega b),\ cos(\omega b) \end{aligned} \tag{5.4}$$

Kako bi dobili bolji prikaz koje su frekvencije bitne, uzimamo normu preko dimenzije rezidualnog toka i dobivamo sljedeću sliku.



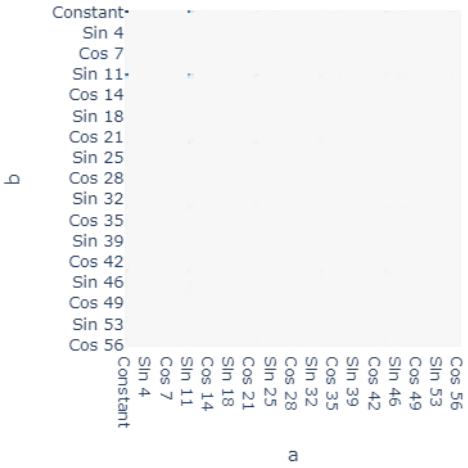
Slika 5.9. Ključne frekvencije mreže (norma)

Iz slike možemo iščitati koje su frekvencije bitne. Frekvencije s većom normom su bitnije za model. Te frekvencije nazivamo ključnim frekvencijama te su nam te frekven-

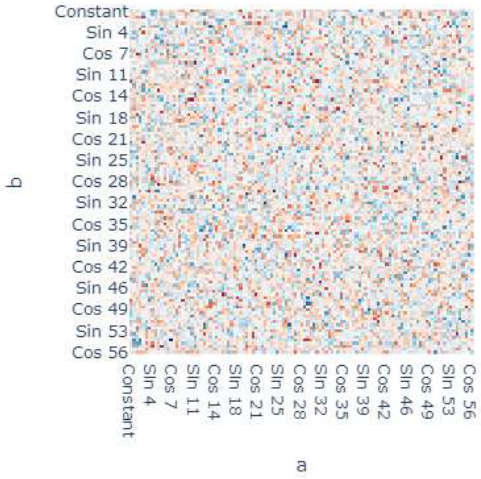
cije jako bitne za daljnju analizu modela. Ključne frekvencije ovog modela su:

$$11, 21, 25, 44, 46 \tag{5.5}$$

Kao što smo množenjem matrice vektorskih reprezentacija s Fourierovom bazom dobili smjerove koji su bitni, množenjem aktivacija neurona s Fourierovom bazom također dobivamo zanimljive rezultate. Kao usporedba korištena su slučajne aktivacije, tj. umjesto smislenih aktivacija korišteni su slučajni brojevi.

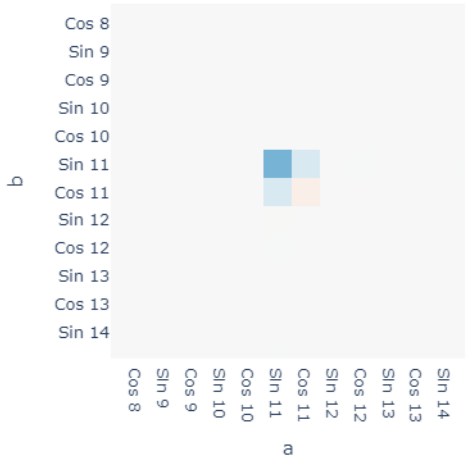


Slika 5.10. 2D Fourierova transformacija 4. neurona



Slika 5.11. 2D Fourierova transformacija slučajnih brojeva

Na slici 5.10 možemo vidjeti jako mali broj aktivacija koje su znatno veće od nule dok su ostale približno 0. Na slici 5.11, gdje imamo aktivacije koje su slučajni brojevi ne možemo izvući nikakav zaključak. Kada približimo sliku 5.10 u područje gdje aktivacije nisu zanemarive, možemo vidjeti da aktivacije koje su bitne odgovaraju u ovom slučaju frekvenciji 11, koja je jedna od frekvencija koje smo ranije označili kao ključnu frekvenciju.



Slika 5.12. Aktivacije prvih četiri neurona

Autori su u izvornom radu naveli da se sljedeći izrazi računaju u sloju glava pažnje te u MLP sloju:

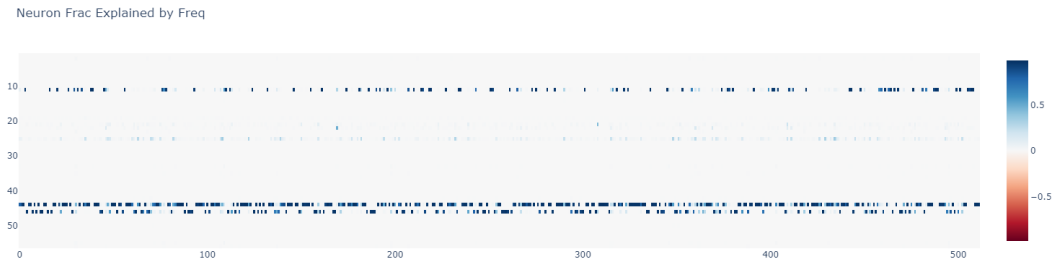
$$\begin{aligned} cos(w_k(a+b)) &= cos(w_k a)\, cos(w_k a) - sin(w_k a)\, sin(w_k b) \\ sin(w_k(a+b)) &= sin(w_k a)\, cos(w_k a) + cos(w_k a)\, sin(w_k b) \end{aligned} \tag{5.6}$$

Ako pogledamo aktivacije na slici 5.12, možemo uočiti da su izrazi:

$$sin(11a)\, sin(11b),\ sin(11a)\, cos(11b),\ cos(11a)\, sin(11b),\ cos(11a)\, cos(11b) \tag{5.7}$$

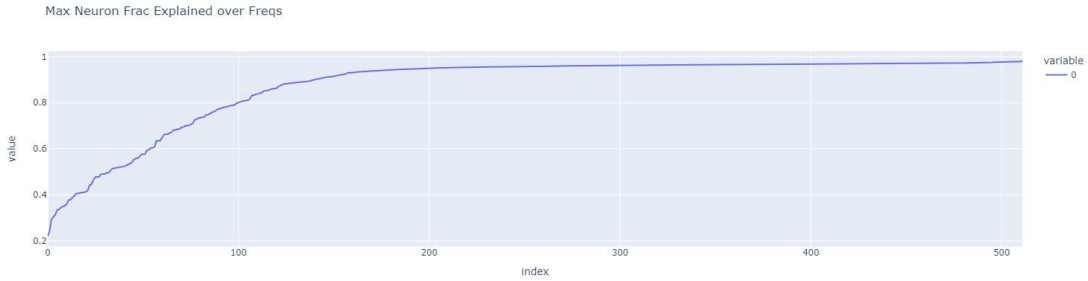
jedini izrazi koji nisu zanemarivi te analizom možemo primijetiti da odgovaraju izrazima koji se nalaze s desne strane jednadžbe 5.6. Ovo nam pokazuje da se ti izrazi računaju u slojevima višeglave pažnje te u MLP sloju.

Kako bi pokazali da to nije slučaj samo za jedan neuron, na slici 5.13 možemo vidjeti koliko je neuron objašnjen s određenim frekvencijama.



Slika 5.13. Objašnjenost neurona frekvencijama

Slika 5.13 prikazuje da je većina neurona velikim dijelom objašnjeno samo s jednom frekvencijom. Npr, neuron sa slike 5.10, je objašnjen 95% s frekvencijom 11.



Slika 5.14. Maksimalna objašnjenost neurona

Slika 5.14 pokazuje neurone i koliko su objašnjeni s frekvencijama. Na slici možemo vidjeti da je preko 400 neurona objašnjeno na ovaj način preko 80%.

Kako bi dokazali da model funkcionira na zadani način, tj. da računa gore spomenuti algoritam moramo isprobati performanse modela kada uklonimo sve frekvencije koje nisu ključne. To radimo 2D Fourierovim transformacijama tako da ih napišemo kao linearnu kombinaciju valova u a i b te postavimo sve ostale izraze osim konstanti te izraza koji odgovaraju izrazima $cos(\omega_k(a+b))$ i $sin(\omega_k(a+b))$ za ključne frekvencije na nula. Također, očekujemo da će se performanse modela znatno pogoršati ako umjesto ključnih frekvencija izaberemo neke slučajne frekvencije.

	Originalni gubitak	**Gubitak s ključnim frek.**	**Gubitak sa sluč. frek.**
Skup za učenje	3.9622e-07	5.2454e-08	11.8878
Skup za testiranje	3.5552e-07	5.0613e-08	11.8476

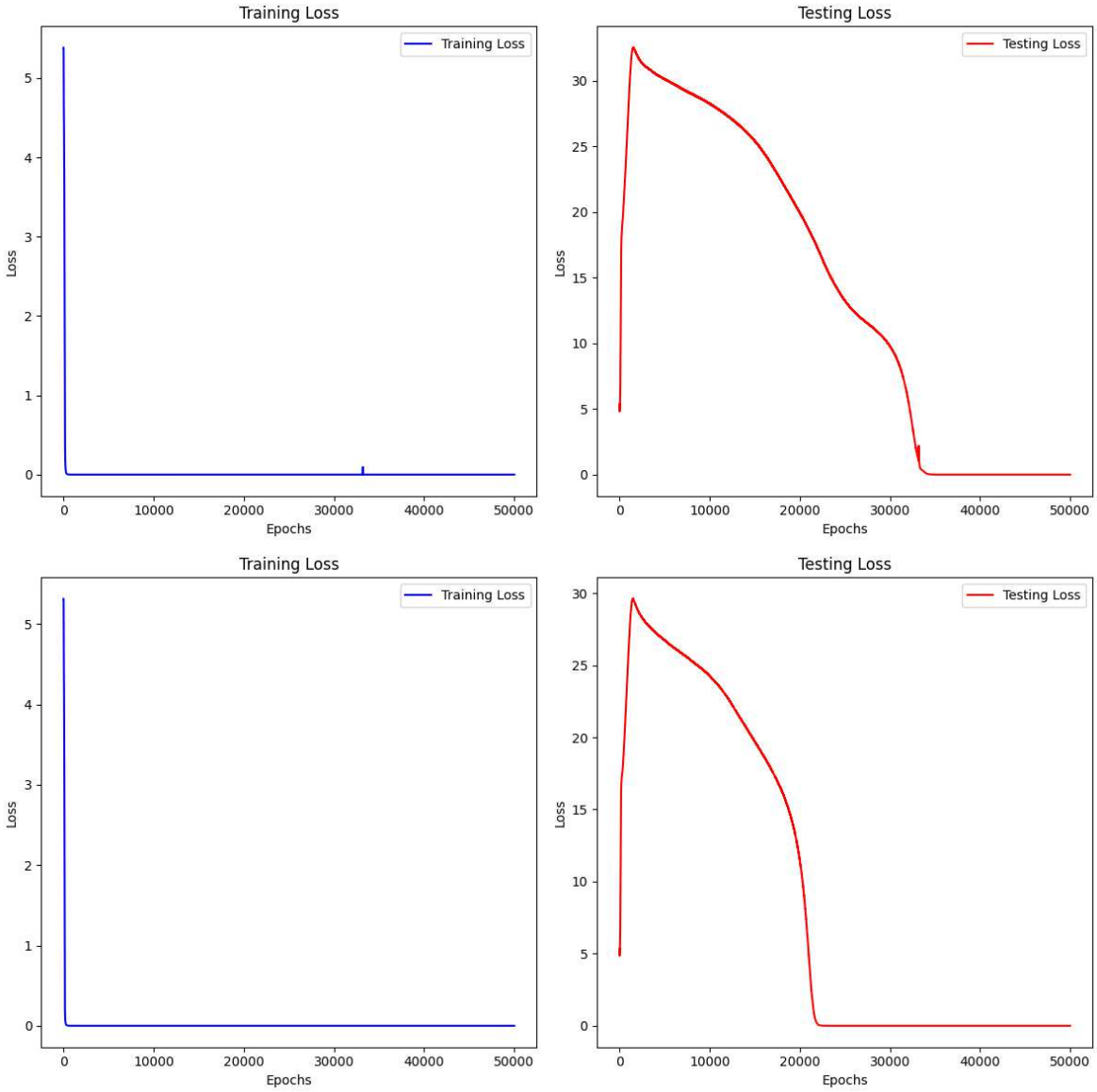
Tablica 5.1. Gubitak modela

Iz tablice 5.1 možemo iščitati da se u oba slučaja, i kod skupa za treniranje i skupa za testiranje izostavljanjem svih frekvencija osim ključnih performanse modela poboljšavaju, tj. gubitak je u tom slučaju manji. Kod odabira slučajnih frekvencija koje ne sadrže ključne frekvencije možemo iščitati da se gubitak znatno povećao, što nam daje dodatnu potvrdu da model funkcionira kako je opisano u odjeljku 5.3.

Konačno, uzimamo 2D diskretnu Fourierovu transformaciju na logit matrici dimenzija $113 \cdot 113 \cdot 113$ za sve parove ulaza od $113 \cdot 113$ kako bismo dobili logite u Fourierovoj bazi. Izostavljanjem svih komponenti osim ključnih, gubitak se smanjuje s $3.6773e-07$ na $3.0580e-08$.

5.4.1. Grokking

Grokking opisujemo kao iznenadan prijelaz na rješenje koje ima dobru generalizaciju nakon velikog broja koraka učenja iako je na početku bio prenaučen.



Slika 5.15. Gubitak na skupu za učenje i na skupu za treniranje za dva modela

Na slici 5.15 prikazani su gubitci na skupu za učenje te gubitci na skupu za testiranje za dva modela. U oba slučaja možemo primijetiti da gubitak na skupu za učenje jako brze padne približno nuli dok gubitak na testnom skupu raste i dostiže svoj maksimum. To nam govori da je model na početku prenaučen. Nakon što gubitak na testnom skupu dosegne svoj maksimum, on počinje padati i možemo vidjeti da u donjem slučaju nakon 20000. epohe gubitak počne jako brzo padati i postane približno 0. Tada znamo da je mreža naučila generalizirati. U izvornom radu [41] autori su zaključili da se treniranje može podijeliti u tri faze: memorizacija podataka za treniranje, formiranje sklopova, gdje mreža uči mehanizam koji generalizira te čišćenje, gdje raspad težina uklanja kom-

ponente memorizacije. Grokking se događa nakon faze formiranja sklopova.

5.5. Hibridizacija modela

Pod hibridizacijom modela misli se na zamjenu neuralnih dijelova s ručno kodiranim komponentama koje implementiraju otkrivene neuralne algoritamske krugove. Hibridizacija je obavljena nad 3 različita modela te su rezultati prikazani u tablici. Hibridi-

	Originalni model		**Hibridni model**	
	Trening	**Test**	**Trening**	**Test**
Model #1	1.1885e-07	1.2222e-07	1.0169e-07	1.0263e-07
Model #2	1.7458e-07	1.6773e-07	1.3421e-07	1.2737e-07
Model #3	2.9343e-07	2.4663e-07	1.1437e-07	1.1659e-07

Tablica 5.2. Rezultati hibridizacije modela

zacijom smo postigli bolje rezultate modela. Iz tablice 5.2 možemo iščitati da za sva tri modela, zamjenom neuralnih dijelova ručno kodiranim komponentama postižemo bolje rezultate. U sva tri slučaja pogreška na skupu za učenje je veća u slučaju kada ne koristimo ručno kodirane komponente. Također, na skupu za testiranje možemo uočiti da je pogreška manja kada koristimo ručno kodirane komponente. To nam govori da zamjenom neuralnih dijelova ručno kodiranim komponentama, poboljšavamo performanse modela.

6. Zaključak

Ovaj diplomski rad bavio se analizom i hibridizacijom jednostavnog transformacijskog modela za modularno zbrajanje, koristeći pristup mehanističke interpretabilnosti. U početku smo detaljno razmotrili arhitekturu, način rada i povijest razvoja transformera, naglašavajući njihovu ključnu ulogu u suvremenim aplikacijama strojnog učenja. Zatim smo istražili mehanističku interpretabilnost, fokusirajući se na glavne obrasce i metode koje omogućuju dublje razumijevanje unutarnjeg djelovanja ovih složenih modela.

Kroz analizu modela transformera otkrili smo da model koristi Fourierov algoritam za množenje, što nam daje značajan uvid u način na koji model obrađuje i manipulira informacijama.

Osim toga, zamjena određenih dijelova neuralne mreže s hibridnim komponentama pokazala se uspješnom, jer je dovela do poboljšanja performansi modela. Ovaj pristup hibridizaciji pokazuje potencijal za optimizaciju modela u specifičnim zadatcima, istovremeno održavajući ili čak povećavajući njihovu interpretabilnost.

Ovim radom doprinosimo boljem razumijevanju kako transformeri rade na osnovnoj razini i kako se mogu poboljšati kroz mehanističku analizu i hibridizaciju. Zaključujemo da je mehanistička interpretabilnost ključna za razvoj transparentnih i učinkovitih modela u strojnom učenju, te da kombiniranje različitih arhitektonskih elemenata može voditi do značajnih unapređenja u njihovoj izvedbi. Ovi nalazi pružaju čvrstu osnovu za buduća istraživanja i razvoj naprednijih i kompleksnijih modela s fokusom na razumljivost i optimizaciju.

Literatura

[1] S. Corzo, "The Versatility of Transformers in Machine Learning: Beyond Just NLP", prosinac 2023. [Mrežno]. Adresa: https://gcptips.medium.com/the-versatility-of-transformers-in-machine-learning-beyond-just-nlp-aad6f913296a

[2] "What is a Transformer Model? | IBM", listopad 2023. [Mrežno]. Adresa: https://www.ibm.com/topics/transformer-model

[3] "Transformer Models: Versatile Uses in Machine Learning - Deepgram Blog ". [Mrežno]. Adresa: https://deepgram.com/learn/applications-of-transformer-models-beyond-gpt-and-chat

[4] "What are Transformers? - Transformers in Artificial Intelligence Explained - AWS". [Mrežno]. Adresa: https://aws.amazon.com/what-is/transformers-in-artificial-intelligence/

[5] "From RNNs to Transformers | Baeldung on Computer Science", srpanj 2023. [Mrežno]. Adresa: https://www.baeldung.com/cs/rnns-transformers-nlp

[6] S. Poudel, "Recurrent Neural Network (RNN) Architecture Explained", kolovoz 2023. [Mrežno]. Adresa: https://medium.com/@poudelsushmita878/recurrent-neural-network-rnn-architecture-explained-1d69560541ef

[7] "Recurrent neural network", kolovoz 2024., page Version ID: 1239877654. [Mrežno]. Adresa: https://en.wikipedia.org/w/index.php?title=Recurrent_neural_network&oldid=1239877654

[8] "Long short-term memory", kolovoz 2024., page Version ID: 1239238315. [Mrežno].

Adresa: https://en.wikipedia.org/w/index.php?title=Long_short-term_memory&oldid=1239238315

[9] R. Hamad, "What is LSTM? Introduction to Long Short-Term Memory", prosinac 2023. [Mrežno]. Adresa: https://medium.com/@rebeen.jaff/what-is-lstm-introduction-to-long-short-term-memory-66bd3855b9ce

[10] "An Overview on Long Short Term Memory (LSTM) - Analytics Vidhya". [Mrežno]. Adresa: https://www.analyticsvidhya.com/blog/2022/03/an-overview-on-long-short-term-memory-lstm/

[11] N. Yehia, "Understanding Long Short-Term Memory (LSTM) Networks", travanj 2024. [Mrežno]. Adresa: https://mlarchive.com/deep-learning/understanding-long-short-term-memory-networks/

[12] "Transformer (deep learning architecture)", kolovoz 2024., page Version ID: 1239899494. [Mrežno]. Adresa: https://en.wikipedia.org/w/index.php?title=Transformer_(deep_learning_architecture)&oldid=1239899494

[13] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, i I. Polosukhin, "Attention Is All You Need", kolovoz 2023., arXiv:1706.03762 [cs]. [Mrežno]. Adresa: http://arxiv.org/abs/1706.03762

[14] "What is Attention Mechanism?". [Mrežno]. Adresa: https://h2o.ai/wiki/attention-mechanism/

[15] R. Kalra, "Introduction to Transformers and Attention Mechanisms", veljača 2024. [Mrežno]. Adresa: https://medium.com/@kalra.rakshit/introduction-to-transformers-and-attention-mechanisms-c29d252ea2c5

[16] "Attention (machine learning)", kolovoz 2024., page Version ID: 1240246796. [Mrežno]. Adresa: https://en.wikipedia.org/w/index.php?title=Attention_(machine_learning)&oldid=1240246796

[17] D. Soydaner, "Attention Mechanism in Neural Networks: Where it Comes and Where it Goes", *Neural Computing and Applications*, sv. 34, br. 16, str. 13 371–13 385, kolovoz 2022., arXiv:2204.13154 [cs]. https://doi.org/10.1007/s00521-022-07366-3

[18] “Transformer Architecture: The Positional Encoding - Amirhossein Kazemnejad’s Blog”. [Mrežno]. Adresa: https://kazemnejad.com/blog/transformer_architecture_positional_encoding/

[19] “How Transformers Work: A Detailed Exploration of Transformer Architecture”. [Mrežno]. Adresa: https://www.datacamp.com/tutorial/how-transformers-work

[20] M. Saeed, “A Gentle Introduction to Positional Encoding in Transformer Models, Part 1”, rujan 2022. [Mrežno]. Adresa: https://machinelearningmastery.com/a-gentle-introduction-to-positional-encoding-in-transformer-models-part-1/

[21] “Feed Forward Neural Network”, svibanj 2019. [Mrežno]. Adresa: https://deepai.org/machine-learning-glossary-and-terms/feed-forward-neural-network

[22] M. Rathi, “FeedForward Neural Networks”. [Mrežno]. Adresa: https://mukulrathi.com/demystifying-deep-learning/feed-forward-neural-network/

[23] S. Cota, “Deep Learning Basics — Part 7 — Feed Forward Neural Networks (FFNN)”, prosinac 2023. [Mrežno]. Adresa: https://medium.com/@sasirekharameshkumar/deep-learning-basics-part-10-feed-forward-neural-networks-ffnn-93a708f84a31

[24] “Feedforward neural network”, kolovoz 2024., page Version ID: 1239074812. [Mrežno]. Adresa: https://en.wikipedia.org/w/index.php?title=Feedforward_neural_network&oldid=1239074812

[25] “Loss Functions: Everything You Need to Know When Assessing Loss Functions Skills”. [Mrežno]. Adresa: https://www.alooba.com/skills/concepts/deep-learning/loss-functions/

[26] “Understanding the Loss Function in Hugging Face’s Transformers Trainer - Salow Studios”. [Mrežno]. Adresa: https://www.salowstudios.com/understanding-the-loss-function-in-hugging-faces-transformers-trainer/#what-is-loss-function

[27] “Huber Loss – Loss function to use in Regression when dealing with Outliers”, srpanj 2023. [Mrežno]. Adresa: https://mlexplained.blog/2023/07/31/huber-loss-loss-function-to-use-in-regression-when-dealing-with-outliers/

[28] J. Brownlee, “How to Configure the Learning Rate When Training Deep Learning Neural Networks”, siječanj 2019. [Mrežno]. Adresa: https://machinelearningmastery.com/learning-rate-for-deep-learning-neural-networks/

[29] “Gentle Introduction to the Adam Optimization Algorithm for Deep Learning - MachineLearningMastery.com”. [Mrežno]. Adresa: https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

[30] “6 Types of Transformer Models and their Use Cases”. [Mrežno]. Adresa: https://datasciencedojo.com/blog/transformer-models-types-their-uses/

[31] “What Are Transformers in NLP: Benefits and Drawbacks”. [Mrežno]. Adresa: https://blog.pangeanic.com/what-are-transformers-in-nlp

[32] “Transformer”. [Mrežno]. Adresa: https://www.clickworker.com/ai-glossary/transformer/

[33] A. Jones, “Introduction to Mechanistic Interpretability”, kolovoz 2024. [Mrežno]. Adresa: https://aisafetyfundamentals.com/blog/introduction-to-mechanistic-interpretability/

[34] L. Bereska i E. Gavves, “Mechanistic Interpretability for AI Safety – A Review”, srpanj 2024., arXiv:2404.14082 [cs]. [Mrežno]. Adresa: http://arxiv.org/abs/2404.14082

[35] J. Engels, I. Liao, E. J. Michaud, W. Gurnee, i M. Tegmark, “Not All Language Model Features Are Linear”, svibanj 2024., arXiv:2405.14860 [cs]. [Mrežno]. Adresa: http://arxiv.org/abs/2405.14860

[36] K. Li, A. K. Hopkins, D. Bau, F. Viégas, H. Pfister, i M. Wattenberg, “Emergent World Representations: Exploring a Sequence Model Trained on a Synthetic Task”, lipanj 2024., arXiv:2210.13382 [cs]. [Mrežno]. Adresa: http://arxiv.org/abs/2210.13382

[37] “Actually, Othello-GPT Has A Linear Emergent World Representation — Neel Nanda”. [Mrežno]. Adresa: https://www.neelnanda.io/mechanistic-interpretability/othello

[38] “Hybrid Machine Learning in Data Science | Domo”. [Mrežno]. Adresa: https://www.domo.com/glossary/what-is-hybrid-machine-learning

[39] “What is Hybrid Machine Learning?”. [Mrežno]. Adresa: https://www.polymersearch.com/glossary/hybrid-machine-learning

[40] “Hybrid Machine Learning Explained in Nontechnical Terms”, veljača 2020. [Mrežno]. Adresa: https://jpt.spe.org/hybrid-machine-learning-explained-nontechnical-terms

[41] N. Nanda, L. Chan, T. Lieberum, J. Smith, i J. Steinhardt, “Progress measures for grokking via mechanistic interpretability”, listopad 2023., arXiv:2301.05217 [cs]. [Mrežno]. Adresa: http://arxiv.org/abs/2301.05217

Sažetak

Analiza i hibridizacija jednostavnog modela transformera za modularno zbrajanje koristeći mehanističku interpretabilnost

Filip Nekić

Ovaj diplomski rad istražuje analizu i hibridizaciju jednostavnog modela transformera za modularno zbrajanje koristeći mehanističku interpretabilnost.Na početku opisujemo povijest, arhitekturu, način rada, glavne komponente te prednosti i nedostatke transformera. Zatim se bavimo mehanističkom interpretabilnosti, fokusirajući se na ključne obrasce i metode za razumijevanje ponašanja modela. Analiza otkriva da model koristi Fourierov algoritam za množenje, što nam daje uvid u unutrašnji rad modela. Nadalje, hibridizacijom određenih neuralnih komponenti postigli smo poboljšanje performansi modela. Ova saznanja doprinose boljem razumijevanju transformera i pokazuju kako ciljane modifikacije mogu optimizirati i performanse i interpretabilnost.

Ključne riječi: transformer, mehanistička, interpretabilnost, modularno, zbrajanje, hibridizacija

Abstract

Analysis and hybridization of a simple transformer model for modular addition using mechanistic interpretability

Filip Nekić

This thesis explores the analysis and hybridization of a simple transformer model for modular addition using mechanistic interpretability. Initially, we provide an overview of the transformer history, architecture, key components, types and advantages and disadvantages. We then talk about mechanistic interpretability, focusing on key patterns and methods for understanding the behavior of the model. Additionally, we briefly explain hybridization in machine learning. Our analysis reveals that the model employs Fourier's multiplication algorithm, offering insights into its internal workings. Furthermore, by hybridizing certain neural components, we achieved improved model performance. These findings enhance our understanding of transformers and demonstrate how targeted modifications can optimize both performance and interpretability.

Keywords: transformer, mechanistic, interpretability, modular, addition, hybridization