Your Paper

You

8. března 2023

1 Dopředná neuronová síť

Dopředná neuronová síť (Feedforward Neural Network) je typ umělé neuronové sítě, která se skládá z několika vrstev neuronů. První vrstva se nazývá vstupní, poslední výstupní a vrstvy mezi nimi jsou označovány jako skryté. Každý neuron je spojen se všemi neurony v následující vrstvě [1].

V dopředné neuronové síti je signál šířen pouze ve směru od vstupní vrstvy přes skryté vrstvy až k výstupní vrstvě. Matematicky lze dopřednou neuronovou síť popsat jako zobrazení vstupního signálu na výstupní signál. Obecně platí, že velikost vstupní vrstvy je dána rozměrem vstupních dat a velikost výstupní vrstvy je za předpokladu klasifikačního problému určena počtem tříd [2].

Nechť má neuronová síť L vrstev, kde $l = 1, 2, ..., L$ označuje vrstvu a $n^{(l)}$ označuje počet neuronů v l-té vrstvě. x je vektor vstupních dat, který je přiveden na vstup sítě. Hodnoty neuronů ve vrstvě l jsou označeny jako vektor $a^{(l)}$ [3].

Každý neuron ve vrstvě l zpracovává vstup $a^{(l-1)}$ z předchozí vrstvy sítě pomocí lineární transformace:

$$\mathbf{z}^{(l)} = \mathbf{W}^{(l)}\mathbf{a}^{(l-1)} + \mathbf{b}^{(l)}$$

kde $W^{(l)}$ je matice vah neuronů mezi $l-1$ a l vrstvou, $b^{(l)}$ je bias vektor pro vrstvu l a $a^{(0)} = x$ [3].

Následně se na lineární transformaci aplikuje nelineární aktivační funkce σ (obecně může být různá v každé vrstvě sítě):

$$\mathbf{a}^{(l)} = \sigma(\mathbf{z}^{(\mathbf{l})})$$

Tento proces se opakuje pro každou vrstvu a výstup poslední vrstvy neuronů je výstupem celé sítě.

Nejčastěji používanými aktivačními funkcemi jsou ReLU (Rectified Linear Unit), sigmoid, tanh a softmax.

ReLU je definována jako [4]:

$$\sigma(z) = \max(0, \mathbf{z})$$

Sigmoidní aktivační funkce má tvar [4]:

$$\sigma(z) = \frac{1}{1 + e^{-\mathbf{z}}}$$

Tanh aktivační funkce má tvar [4]:

$$\sigma(z) = \frac{e^{\mathbf{z}} - e^{-\mathbf{z}}}{e^{\mathbf{z}} + e^{-\mathbf{z}}}$$

Funkce Softmax může být na rozdíl od sigmoidních funkcí, které se používají pro binární klasifikaci, použita pro problémy s klasifikací více tříd. Pokud má výstupní vrstva K neuronů, potom má Softmax funkce tvar [4]:

$$\mathbf{a}^{(L)}j = \frac{e^{\mathbf{z}_j^{(L)}}}{\sum k = 1^K e^{\mathbf{z}_k^{(L)}}}$$

Cílem učení neuronové sítě je nalézt takové hodnoty vah a biasu neuronů, aby síť byla schopna klasifikovat nebo regresovat vstupní data s co největší správností. K tomuto účelu jsou používány různé učící algoritmy, jako například backpropagation. Při zpětném šíření je vypočítána chyba sítě, což je rozdíl mezi očekávaným výstupem sítě a jejím skutečným výstupem. Tato chyba je dále propagována zpět sítí a jsou vypočítány gradienty chyby vzhledem k váhám a biasům. Gradient chyby vzhledem k váhám a biasům je vypočítán pomocí řetězového pravidla derivace. Tento gradient je následně použit k aktualizaci hodnot vah a biasů pomocí gradientního sestupu, což je iterativní metoda pro nalezení minima funkce. Při použití gradientního sestupu se váhy a biasy upravují tak, aby minimalizovaly chybu sítě [1, 2].

2 ROS (Robot Operating System)

ROS (Robot Operating System) je open-source softwarový framework určený pro vývoj robotických aplikací. Poprvé byl uveden v roce 2007 na Univerzitě v Stanfordu. Poskytuje knihovny a nástroje umožňující snadno vytvářet a sdílet software pro řízení robotů [5, 6].

ROS byl vyvíjen veřejně s použitím permisivní licence BSD a postupně se stal široce používaným v komunitě výzkumníků robotiky. Na rozdíl od klasického přístupu, kdy všichni přispěvatelé umístí svůj kód na stejné servery, byl ROS vyvíjen v několika institucích a pro různé druhy robotů. Toto se stalo jednou z největších výhod ROS ekosystému. Současně je používán desetitisíci uživateli po celém světě v oblastech od domácích projektů v rámci koníčků, po velké průmyslové automatizované systémy [6]. V současné době je ROS stále aktivně vyvíjen a jeho verze 2.0, známá jako ROS 2, je určena pro použití v průmyslových aplikacích s většími nároky na spolehlivost a bezpečnost [7].

Základními koncepty implementace ROS jsou uzly (Nodes), zprávy (Messages), témata (Topics) a služby (Services). Uzly jsou procesy provádějící výpočty, přičemž systém obvykle sestává z mnoha uzlů. Termín "uzel"je v tomto kontextu zaměnitelný se "softwarovým modulem". Použití termínu "uzel"vzniká z vizualizací systémů pomocí grafu, kde jsou procesy zobrazeny jako uzly grafu a peer-to-peer spojení jako oblouky (Arcs) [6, 8].

Uzly mezi sebou komunikují předáváním zpráv. Zpráva je striktně typovaná datová struktura. Podporovány jsou jak standardní primitivní datové typy (celé číslo, desetinné číslo, boolean, atd.), tak pole primitivních datových typů. Zprávy mohou být složeny z jiných zpráv a polí jiných zpráv, v libovolné hloubce vnoření [6, 8].

Uzel odesílá zprávu publikováním na dané téma. Uzel, který má zájem o určitý druh dat, se přihlásí k příslušnému tématu. Pro jedno téma může existovat několik souběžných vydavatelů a odběratelů a jeden uzel může publikovat a/nebo odebírat více témat. Vydavatelé a odběratelé většinou nevědí o existenci druhého. Idea publikování a odběru témat v ROS umožňuje komunikaci mezi moduly na velmi flexibilní úrovni.

"Broadcast"směrování však není vhodné pro synchronní transakce. Proto ROS poskytuje tzv. "service"(službu), která je definována názvem a dvěma přísně typovanými zprávami: jednou pro požadavek a druhou pro odpověď [6, 8].

Filozofické cíle ROS lze shrnout následovně:

- Peer-to-peer: ROS by měl být navržen tak, aby jednotlivé uzly mohly komunikovat přímo mezi sebou, aniž by bylo nutné centralizované řízení, tj. aby byl decentralizovaný [6, 8].
- Založení na nástrojích (Tools-based): ROS poskytuje mnoho nástrojů, jako jsou nástroje pro získávání a nastavování konfiguračních parametrů, vizualizaci topologie připojení peer-to-peer, měření využití šířky pásma, grafické vykreslování data zpráv, automatické generování dokumentace atd. Avšak nemá kanonické integrované vývojové a runtime prostředí, všechny úkoly jsou prováděny samostatnými programy, což podporuje vytváření nových, vylepšených implementací [6, 8].
- Mnohojazyčnost (Multi-lingual): Softwarové moduly ROS lze psát v jakémkoli programovacím jazyce, pro který byla napsána klientská knihovna. Tyto moduly mezi sebou komunikují díky jazykově neutrálnímu a jednoduchému jazyku pro definici rozhraní (IDL), které slouží k popisu zpráv odesílaných mezi moduly [6, 8].
- Thin: Konvence ROS vybízejí přispěvatele k vytváření samostatných knihoven. Tyto knihovny jsou následně zabaleny a umožňují komunikaci pomocí zpráv s jinými ROS moduly. Tato další vrstva umožňuje opětovné použití softwaru mimo ROS pro jiné aplikace [6, 8].
- Zdarma a Open-Source: Úplný zdrojový kód ROS je veřejně dostupný [6, 8].

3 Speechcloud

SpeechCloud je platforma pro zpracování řeči a analýzu hlasu vyvinutá společností Speech Tech. Umožňuje například automatický přepis diktátu do psané podoby, generování vysoce přirozené řeči ze zadaných textů, ověření a verifikaci osob na základě jedinečných charakteristik jejich hlasu a hlasovou komunikaci mezi člověkem a počítačem. Platforma podporuje několik jazyků včetně češtiny a slovenštiny. SpeechCloud je navržen tak, aby umožňoval snadné integrování s různými aplikacemi a systémy třetích stran. [9].

Základní SpeechCloud technologie lze označit jako:

- Automatické rozpoznávání řeči (ASR) - SpeechCloud využívá pokročilé ASR technologie, které umožňují převádět mluvenou řeč na text. Tato technologie používá strojové učení a hluboké neuronové sítě, aby dosáhla vysoké přesnosti a robustnosti při rozpoznávání řeči.
- Text-to-speech (TTS) - SpeechCloud také umožňuje generovat řeč z textu pomocí TTS technologie. Podporováno je několik hlasových stylů.

4 Sentence transformery

Sentence Transformer je model strojového učení, který dokáže transformovat vstupní textové sekvence (např. věty) do vektorů s vysokou dimenzionalitou. Tyto vektory pak

mohou být využity například pro kategorizaci, porovnávání podobnosti nebo další úlohy zpracování přirozeného jazyka.

Architektura zvaná transformer byla poprvé představena v roce 2017 v článku "Attention Is All You Need". Tato architektura dokáže na rozdíl od tradičních rekurentních nebo konvolučních neuronových sítí pracovat zcela bez opakování a konvolucí, navíc dosahují tyto modely lepší kvality, jsou více paralelizovatelné a vyžadují podstatně méně času na trénování. Klíčovou součástí transformeru je mechanismus pozornosti (Attention), který umožňuje modelu se zaměřit na důležité části vstupní sekvence a ignorovat méně významné informace [10].

Vzorec pro výpočet pozornosti vypadá následovně [10]:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}$$

kde:

- $\mathbf{Q} \in \mathbb{R}^{n \times d_k}$ je matice dotazů (Query Matrix)
- $\mathbf{K} \in \mathbb{R}^{m \times d_k}$ je matice klíčů (Key Matrix)
- $\mathbf{V} \in \mathbb{R}^{m \times d_v}$ je matice hodnot (Value Matrix)
- d_k je počet dimenzí vektoru klíče (Key Vector)
- n a m jsou počty dotazů a klíčů/hodnot
- $\sqrt{d_k}$ slouží k normalizaci skalárního součinu a zlepšuje stabilitu gradientů v trénování

Hlavním cílem Sentence Transformeru je naučit se reprezentovat věty tak, aby výsledné vektory zachovávaly sémantickou podobnost mezi větami. To umožňuje využití těchto vektorů pro různé úlohy, jako například hledání podobností mezi větami, kategorizaci textů, vyhledávání odpovědí atd. Pro výpočet vektorové reprezentace vět se často používají předtrénované modely, jako jsou BERT nebo GPT, které jsou široce dostupné pro použití v různých aplikacích.

4.1 FERNET-C5

FERNET-C5 je jednojazyčný BERT model, který byl od úplného počátku trénován na datech českého korpusu Colossal Clean Crawled Corpus (C5) - obdoba anglického datasetu C4 pro češtinu. Trénovací data obsahují téměř 13 miliard slov. Model má stejnou architekturu jako původní BERT model, tedy 12 transformačních bloků, 12 pozornostních hlav (Attention Heads) a skrytou velikost 768 neuronů. Na rozdíl od BERT modelů od Googlu byla použita tokenizace SentencePiece místo interní tokenizace WordPiece od Googlu [11].

Model je veřejně k dispozici online a umožňuje integraci s různými aplikacemi, jako jsou chatboty, analýza sentimentu, nebo klasifikace textu.

Reference

[1] Daniel Svozil, Vladimir Kvasnicka, and Jiri Pospíchal. Introduction to multi-layer feed-forward neural networks. *Chemometrics and Intelligent Laboratory Systems*, 39:43–62, 1997.

[2] Martin Bulín. *On Using Multi-Agent Technologies to Build Neural Networks*. PhD thesis, University of West Bohemia in Pilsen, Univerzitní 2732/8, 301 00 Pilsen 3, Czech Republic, 2021.

[3] Niranjan Kumar. Deep learning: Feedforward neural networks explained, Apr 2019.

[4] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. Activation functions in neural networks. *Towards Data Sci*, 6(12):310–316, 2017.

[5] Evan Ackerman and Erico Guizzo. Wizards of ros: Willow garage and the making of the robot operating system, Nov 2022.

[6] Morgan Quigley, Brian Gerkey, and William D. Smart. *Programming Robots with ROS: A Practical Introduction to the Robot Operating System*. O'Reilly Media, Inc., 1st edition, 2015.

[7] Jongkil Kim, Jonathon M. Smereka, Calvin Cheung, Surya Nepal, and Marthie Grobler. Security and performance considerations in ROS 2: A balancing act. *CoRR*, abs/1809.09566, 2018.

[8] Morgan Quigley, Ken Conley, Brian P. Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[9] SpeechTech — speechtech.cz. https://www.speechtech.cz/. [Accessed 25-Feb-2023].

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[11] Jan Lehecka and Jan Svec. Comparison of czech transformers on text classification tasks. *CoRR*, abs/2107.10042, 2021.