Obraz zawierający Grafika, Czcionka, logo, design

Opis wygenerowany automatycznie

Studium Magisterskie

Kierunek Big Data

Imię i nazwisko autora

Antoni Ziółkowski

Nr albumu 83133

Lora

Praca magisterska

pod kierunkiem naukowym

Dr Sebastiana Zająca

Instytut

Ekonometrii

Warszawa 2024

**Spis treści**

[Wstęp 5](#_Toc162202857)

[Rozdział I. Ogólna charakterystyka dużych modeli językowych 6](#_Toc162202858)

[I.1. Architektura transformera 6](#_Toc162202859)

[I.1.1. Opis architektury 6](#_Toc162202860)

[I.2. Transformer a generowanie języka naturalnego 10](#_Toc162202861)

[I.3. Proces uczenia dużych modeli językowych 14](#_Toc162202862)

[I.3.1. Metodologie uczenia 15](#_Toc162202863)

[Rozdział II. LoRA: adaptacja LLMów macierzami niskiego rzędu 16](#_Toc162202864)

[II.1. Pojęcie adaptacji dużych modeli językowych 16](#_Toc162202865)

[II.1.1. LoRA 16](#_Toc162202866)

[II.2. Praktyczne zastosowanie w modelach 16](#_Toc162202867)

[II.3. Korzyści wynikające z zastosowania „LoRy” 16](#_Toc162202868)

[II.3.1. Oszczędność czasu i zasobów pieniężnych 16](#_Toc162202869)

[II.3.2. Dostępność dla mniejszych podmiotów 16](#_Toc162202870)

[II.4. Wyzwania i ograniczenia (zobaczę na bieżąco czy opłaca się pisać o downside’ach czy czymś innym) 16](#_Toc162202871)

[Rozdział III. Implementacja praktyczna i ocena techniki adaptacji LoRA 17](#_Toc162202872)

[III.1. Projektowanie eksperymentów 17](#_Toc162202873)

[III.2. Metodologia 17](#_Toc162202874)

[III.3. Wyniki i analiza 17](#_Toc162202875)

[III.4. Dyskusja 17](#_Toc162202876)

[Rozdział IV. Przyszłość i rozwój dużych modeli językowych 18](#_Toc162202877)

[IV.1. Nowe trendy i innowacje w LLM 18](#_Toc162202878)

[IV.2. Rola adaptacji macierzami niskiego rzędu i innych narzędzi w rozwoju LLM-ów 18](#_Toc162202879)

[IV.3. Wpływ społeczny i etyczny 18](#_Toc162202880)

[Zakończenie 19](#_Toc162202881)

[Bibliografia 20](#_Toc162202882)

[Spis tabel 22](#_Toc162202883)

[Spis wykresów 23](#_Toc162202884)

[Spis rysunków 24](#_Toc162202885)

[Streszczenie 25](#_Toc162202886)

Wstęp

Wrzucić na pewno krótką wspominkę/zarys historyczny NLP

1. Ogólna charakterystyka dużych modeli językowych
   1. Architektura transformera

Architektura transformera miała skromne początki, przedstawiona została bowiem przez jej autorów głównie jako alternatywa do dotychczasowych sieci neuronowych stosowanych w tłumaczeniu maszynowym. Szybko okazało się jednak, że jej potencjalne zastosowania są znacznie szersze, zmieniła ona całkowicie krajobraz uczenia maszynowego. Cechuje się bowiem bardzo dobrą możliwością adaptacji do różnych zadań; można stosować ją także w scenariuszach niepowiązanych z procesowaniem języka naturalnego, na przykład składaniu białek, rozpoznawaniu obrazów, rozpoznawania mowy, czy uczenia ze wzmocnieniem[[1]](#footnote-1),[[2]](#footnote-2),[[3]](#footnote-3),[[4]](#footnote-4). Najistotniejszym zastosowaniem transformera w bieżących czasach są jednak generatywne modele językowe, bazują na niej produkty takie jak ChatGPT, Llama, czy Bard.

Głównym celem autorów było stworzenie sieci, która cechuje się lepszą pamięcią długoterminową i umiejętnością identyfikacji globalnych zależności w danych treningowych, w porównaniu do dotychczas używanych sieci rekurencyjnych i ich rozwinięć, takich jak LSTM czy GRU. Te bowiem, pomimo lat rozwoju, nadal nie radziły sobie zbyt dobrze z długimi sekwencjami danych wejściowych, a ponadto stwarzały trudności w procesie uczenia – rekurencję ciężko było zrównoleglić obliczeniowo.

* + 1. Opis architektury

Transformer jest modelem seq2seq, zaproponowanym w prace „Attention Is All You Need”[[5]](#footnote-5), a jego najważniejszą cechą jest silne skupienie na i wykorzystanie mechanizmu Multi-Head Attention. Mechanizm uwagi w innych postaciach rozwijał się jeszcze przed transformerem, a za jego początek przyjmuje się pracę bowiem rozwijał się już w latach poprzedzających pracę „*Neural Machine Translation by Jointly Learning to Align and Translate”* z 2014[[6]](#footnote-6). Oprócz zwiększonej ogólnej efektywności modelu w zadaniach tłumaczenia maszynowego czy generowania języka naturalnego, cechuje się on również dotychczas niespotykaną możliwością paralelizacji obliczeń. Stanowiło to swojego rodzaju mieszankę wybuchową dla rynku kart graficznych – kapitalizacja rynkowa lidera – korporacji NVIDIA, od 2017 do 2023 zwiększyła się z 117.26 miliardów dolarów do 2.272 bilionów dolarów[[7]](#footnote-7).

Rysunek 1: Architektura transformera

Obraz zawierający tekst, diagram, zrzut ekranu, Plan

Opis wygenerowany automatycznie

**Źródło**: A. Vaswani i in., *Attention Is All You Need*, „Advances in Neural Information Processing Systems” t. 2017-December (2017), https://arxiv.org/abs/1706.03762v7.

Danymi wejściowymi do modelu transformera są tokeny osadzone w *d*model wymiarowej przestrzeni wektorowej. Są one sumą wcześniej nauczonych osadzeń wejściowej sekwencji znaków i wektorów pozycyjnych. Jak bowiem zauważają autorzy, transformer nie wykorzystuje rekurencji ani konwolucji, zatem nie byłby w stanie nauczyć się zależności pozycyjnych między słowami bez dodania informacji pozycyjnych do ciągów wejściowych. Kodowanie pozycyjne zastosowane w „Attention Is All You Need” wyrażone jest następująco:

gdzie *pos* – pozycja

*i* –wymiar.

**Źródło:** A. Vaswani i in., *Attention Is All You Need*, „Advances in Neural Information Processing Systems” t. 2017-December (2017), https://arxiv.org/abs/1706.03762v7.

Transformer składa się z dwóch bloków: enkodera i dekodera. Budowa ta jest typowa i znana z poprzednich iteracji modelów seq2seq, skupiających się na tłumaczeniu maszynowym. Enkoder koduje ciągi znaków w języku obcym, a dekoder ma za zadanie wygenerowanie ciągu znaków w języku docelowym. W pierwotnej pracy obydwie warstwy ustawione są szeregowo i jest ich N=6.

Przygotowane sekwencje przekazywane są do enkodera, gdzie w pierwszym kroku przetwarzane są przy użyciu mechanizmu Multi-Head Attention. Pojedynczy składnik uwagi – „Scaled Dot-Product Attention” wyrażony jest następująco:

gdzie *Q* – macierz zapytań (queries)

*K* – macierz kluczy (keys)

*V* – macierz wartości (values)

*dk* – wymiar macierzy kluczy.

**Źródło:** A. Vaswani i in., *Attention Is All You Need*, „Advances in Neural Information Processing Systems” t. 2017-December (2017), https://arxiv.org/abs/1706.03762v7.

Mechanizm ten pozwala na kalkulację jak „ważne” są dla siebie nawzajem poszczególne sekwencje znaków; otrzymana wartość końcowa stanowi ważoną sumę wartości „value” z wagą, mierzoną jako zgodność zapytania i klucza. Uwaga w takiej postaci wykonywana jest na zestawach danych - V, K i Q są matrycami odpowiednich wartości. Pełna Multi-Head Attention natomiast, polega na rzucie liniowym wartości V, K i Q do przestrzeni o wymiarach dk, dv, dq przy pomocy warstwy liniowej. Obliczenia uwagi są następnie wykonywane równolege, a ich wyniki są łączone i ponownie rzutowane warstwą liniową.

gdzie

gdzie projekcje są macierzami parametrów:

i .

**Źródło:** A. Vaswani i in., *Attention Is All You Need*, „Advances in Neural Information Processing Systems” t. 2017-December (2017), https://arxiv.org/abs/1706.03762v7.

W następnym kroku każda pozycja procesowana jest w pełni połączoną siecią feed-forward – dwie liniowe transformacje z funkcją aktywacji RELU pomiędzy. Ponadto, zastosowane są połączenia rezydualne, a zatem zarówno po bloku Multi-Head Attention i FFN, wykonywane są obliczenia addycji i normalizacji danych.

Blok dekodera zbudowany jest bardzo podobnie jak enkoder, z dwiema kluczowymi zmianami. Po pierwsze, matryce Q i K, są maskowane tak, aby sekwencje nie mogły stosować mechanizmu uwagi na tokenach „z przyszłości”. Innymi słowy, tokeny w środku sekwencji mogą się „uczyć” informacji jedynie o tokenach je poprzedzających. Drugą zmianą jest wprowadzenie drugiego bloku Multi-Head Attention, przez niektórych nazywanym Cross-Attention[[8]](#footnote-8). Tym sposobem sekwencje wejściowe bloku dekodera mogą „uważać” nie tylko na siebie, ale także na informacje z bloku enkodera, co jest kluczowe podczas tłumaczenia maszynowego.

* 1. Transformer a generowanie języka naturalnego

Jak okazało się niedługo po zaproponowaniu oryginalnej architektury transformera, dobrze radzi sobie ona nie tylko z zadaniami seq2seq, ale także otwartym generowaniem tekstu, co szybko znalazło swoje miejsce w tworzeniu modeli chatbotów. Wiązało się to jednak z wprowadzeniem kilku kluczowych zmian do oryginalnej architektury, które można prześledzić na podstawie artykułów prezentujących nowe iteracje GPT (Generative Pre-trained Transformer) od OpenAI.

Pierwszą z zaproponowanych zmian, wykorzystanych w GPT-1, było całkowite porzucenie bloku enkodera[[9]](#footnote-9). Z empirycznego punktu widzenia, modele posiadające jedynie dekoder, sprawdzają się lepiej w otwartym generowaniu tekstu oraz inferencji na podstawie „promptów”. Modele składające się z obu bloków wykazują się lepszymi wynikami w zadaniach klasyfikacyjnych oraz seq2seq. Kluczowym jest również fakt, że zastosowanie jedynie bloku dekodera znacznie zmniejsza liczbę parametrów w modelach, a zatem pozytywnie wpływa na czas inferencji, koszty i skalowalność. Jak zauważają autorzy ULLP, modele enkoder-dekoder mają średnio 2x więcej parametrów od modelów wykorzystujących jedynie dekoder. Z perspektywy funkcjonalnej, w modelach enkoder-dekoder, sekwencje input procesowane są jedynie przez enkoder, natomiast dekoder zajmuje się sekwencjami target, a zatem obydwie z nich mają własne zestawy parametrów. W przypadku modeli porzucających enkoder, sekwencje input-target są łączone[[10]](#footnote-10). Pod względem konstrukcji modelu, warto zauważyć, że w modelach „decoder-only” blok dekodera posiada tylko jedną warstwę Multi-Head Self Attention, ponieważ w takiej konfiguracji warstwa Cross Attention, łącząca enkoder i dekoder nie jest wymagana.

Jako część GPT-2, zamieniono kolejność warstw normalizacji (LayerNorm) - w przeciwieństwie do pierwotnej architektury, występują teraz przed poszczególnymi warstwami dekodera. Wykorzystano także jedną dodatkową warstwę normalizacji po ostatnim bloku uwagi własnej. Ponadto, zastosowano zmodyfikowaną inicjalizację wag oraz bezstratny, odwracalny tokenizer tekstu wejściowego[[11]](#footnote-11).

Niestety wraz z narastającą motywacją monetyzacji swoich produktów, architektury GPT-3 i GPT-4 są znacznie bardziej zamknięte od poprzedników, jednakże nawet z ograniczonym wglądem, zauważyć można kilka kluczowych innowacji. Jedną z nich jest zastosowanie architektury transformera rzadkiego[[12]](#footnote-12). „Wymagania dotyczące pamięci i obliczeń takich sieci [transformerów] rosną kwadratowo wraz z długością sekwencji, co wyklucza ich użycie w długich sekwencjach.”[[13]](#footnote-13) Autorzy wprowadzili zatem kilka rzadkich faktoryzacji macierzy atencji, co pozwoliło osiągnąć niemożliwą do tej pory długość sekwencji, jednocześnie nie tracąc przy tym wydajności modelu. Zgodnie z Rys. 2, tokeny nie są maskowane macierzami trójkątnymi, lecz rzadkimi.

Rysunek 2: Transformery rzadkie

Obraz zawierający linia, zrzut ekranu, kwadrat, Jaskrawoniebieski

Opis wygenerowany automatycznie

**Źródło**: R. Child i in., Generating Long Sequences with Sparse Transformers, (2019), https://arxiv.org/abs/1904.10509v1.

Szczegóły związane z GPT-4 są z kolei całkowicie prywatne, natomiast przecieki sugerują, że model bazuje na uczeniu grupowym (ensemble learning), a dokładnie MoE (Mixture of Experts), tj. składa się z wielu mniejszych modeli, posiadających odrębne specjalizacje w różnych dziedzinach wiedzy[[14]](#footnote-14). Znajduje to także swoje odwzorowanie w modelach open-source, np. Mixtral 8x7B[[15]](#footnote-15).

Wiele współczesnych modeli językowych wprowadza również zmiany w procesie kodowania pozycyjnego tokenów wejściowych. Przedstawiając oryginalną architekturę transformera wykorzystane zostało kodowanie w pozycjach bezwzględnych. Rok później zaproponowane zostało także kodowanie względne[[16]](#footnote-16). Najciekawszym rozwiązaniem jednakże, wydaje się osadzanie w pozycjach rotacyjnych; „RoPE koduje pozycję absolutną za pomocą macierzy rotacji, w tym samym momencie uwzględniając wyraźną zależność względnej pozycji w formułowaniu mechanizmu uwagi własnej”[[17]](#footnote-17). Mechanizm ten cechuje się kilkoma plusami względem pozostałych metod:

* Długość sekwencji wejściowych jest elastyczna,
* RoPE wprowadza zanikająca zależność między tokenami wraz ze wzrostem odległości względnych,
* Umożliwia wyposażenie liniowego mechanizmu uwagi własnej w pozycyjne kodowanie względne.

Być może najbardziej oczywistym usprawnieniem jest stopniowe zwiększanie skali tworzonych modeli. Bazowy model zaproponowany w „*Attention Is All You Need”* posiadał 65 milionów parametrów, natomiast liczebność parametrów współczesnych modeli wyrażamy w miliardach i bilionach[[18]](#footnote-18).

Wykres 1: Liczba parametrów modeli w czasie

Obraz zawierający tekst, diagram, linia, Wykres

Opis wygenerowany automatycznie

**Źródło**: A. Alvi, P. Kharya, *Using DeepSpeed and Megatron to Train Megatron-Turing NLG 530B, the World’s Largest and Most Powerful Generative Language Model - Microsoft Research* https://www.microsoft.com/en-us/research/blog/using-deepspeed-and-megatron-to-train-megatron-turing-nlg-530b-the-worlds-largest-and-most-powerful-generative-language-model/, 11 października 2021 r., dostęp 19 marca 2024 r.

* 1. Proces uczenia dużych modeli językowych

Metody uczenia sieci bazujących na architekturze transformera, przewidywalnie, różni się w zależności od modelowanego problemu czy zadania jakie model ma wykonywać. W przypadku modeli stworzonych do modelowania językowego, a dokładniej generujących język naturalny, bardzo popularnych w bieżących czasach, chatbotów-asystentów, mamy do czynienia z modelami typu GPT – Generative Pre-trained Transformer. Ze względu na bardzo dużą złożoność języka naturalnego, modele tego typu cechują się ogromną liczbą parametrów – miliony, miliardy, a zatem nawet proces inferencji, tj. generowania tekstu z nauczonego już modelu, wymaga klastrów procesorów graficznych, aby obywał się on w racjonalnym czasie. Charakterystyka ta jeszcze istotniej prezentuje się jednak w procesie uczenia modeli, który zajmuje/kosztuje (tutaj przykład openai albo llamy), a zatem jest fizycznie niewykonalny przez pojedyncze jednostki bez korporacyjnego finansowania. Proces uczenia takiego asystenta zaczyna się od „pre-trainingu” t.j. trenowania parametrów na gigantycznym korpusie danych (unlabelled text); cały internet (tutaj ładniej opisać, sprawdzić w papierach, w gpt3 opisują czego i ile użyli). Oczywiście im lepsza jakość danych wejściowych, tym lepsze będą wyniki modelu, a zatem dużo uwagi w ostatnim czasie przykłada się do przygotowania/oczyszczania danych wejściowych. Dla przykładu, firma OpenAI silnie polegała na danych z platformy Reddit, zaciągając wątki mające 3 upvote’y lub więcej. Po wykonaniu uczenia wstępnego modele tego typu nie są niczym innym niż generatorami losowego tekstu i daleko im jeszcze do komercjalnie dostępnych produktów. Kluczowym etapem uczenia jest więc „fine-tuning” gdzie sekwencje wejściowe przyjmują postać konwersacji Asystent-Użytkownik o charakterze konwersacyjnym. W porównaniu do ogromnych zasobów potrzebnych do pre-trainingu, modele względnie szybko adaptują się do poszczególnych zadań używając fine-tuningu. (jakieś źródło/porównanie resourców potrzebnych do pre-train/fine-tune). Proces ten nie zapewnia jednak, że model będzie akceptowalny etycznie i moralnie, szczególnie biorąc pod uwagę skalę end-userów. Częste jest zatem, że firmy w tym momencie wykorzystują techniki takie jak RLHF, gdzie to ludzie oceniają tekst generowany przez model w celu poprawienia AI „alignmentu”. Warto w tym miejscu zwrócić uwagę, że proces zwany pre-trainingiem nie został wprowadzony wraz z genezą architektury transformera, był bowiem wykorzystywany już wcześniej i w innych architekturach: (Mikolov et al., 2013; Pennington et al., 2014; Neumann et al., 2018; Dai & Le, 2015; Howard & Ruder, 2018) – papier Unifying Language Learning Paradigms – strona 6

Również na tej samej stronie, niżej, opisane dlaczego decoder only sprawdzał się lepiej w open ended text generation

Info sprawdzić:

Generative pretraining (GP) was a long-established concept in machine learning applications.[16][17][18] It was originally used as a form of semi-supervised learning, as the model is trained first on an unlabelled dataset (pretraining step) by learning to generate datapoints in the dataset, and then it is trained to classify a labelled dataset.”

* + 1. Metodologie uczenia

1. LoRA: adaptacja LLMów macierzami niskiego rzędu
   1. Pojęcie adaptacji dużych modeli językowych
      1. LoRA

**Tabela 1. Tabela testowa**

|  |  |
| --- | --- |
| 1 | 2 |
| 3 | 4 |

Źródło: ;kljlkjlkj

**Rysunek 3. Rysunek testowy**

Obraz zawierający tekst, zrzut ekranu, diagram

Opis wygenerowany automatycznie

* 1. Praktyczne zastosowanie w modelach
  2. Korzyści wynikające z zastosowania „LoRy”
     1. Oszczędność czasu i zasobów pieniężnych
     2. Dostępność dla mniejszych podmiotów
  3. Wyzwania i ograniczenia (zobaczę na bieżąco czy opłaca się pisać o downside’ach czy czymś innym)

1. Implementacja praktyczna i ocena techniki adaptacji LoRA
   1. Projektowanie eksperymentów
   2. Metodologia
   3. Wyniki i analiza
   4. Dyskusja
2. Przyszłość i rozwój dużych modeli językowych
   1. Nowe trendy i innowacje w LLM
   2. Rola adaptacji macierzami niskiego rzędu i innych narzędzi w rozwoju LLM-ów
   3. Wpływ społeczny i etyczny

Zakończenie

Bibliografia

Bahdanau D., Cho K.H., Bengio Y., *Neural Machine Translation by Jointly Learning to Align and Translate*, „3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings” (2014), https://arxiv.org/abs/1409.0473v1.

Brown T.B., Mann B., Ryder N., i in., *Language Models are Few-Shot Learners*, „Advances in Neural Information Processing Systems” t. 2020-December (2020), https://arxiv.org/abs/2005.14165v4.

Chen L., Lu K., Rajeswaran A., i in., *Decision Transformer: Reinforcement Learning via Sequence Modeling*, „Advances in Neural Information Processing Systems” t. 18 (2021), https://arxiv.org/abs/2106.01345v2.

Child R., Gray S., Radford A., Sutskever I., *Generating Long Sequences with Sparse Transformers*, (2019), https://arxiv.org/abs/1904.10509v1.

Dosovitskiy A., Beyer L., Kolesnikov A., i in., *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*, „ICLR 2021 - 9th International Conference on Learning Representations” (2020), https://arxiv.org/abs/2010.11929v2.

Gulati A., Qin J., Chiu C.C., i in., *Conformer: Convolution-augmented Transformer for Speech Recognition*, „Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH” t. 2020-October (2020), DOI: 10.21437/Interspeech.2020-3015, https://arxiv.org/abs/2005.08100v1.

Jiang A.Q., Sablayrolles A., Roux A., i in., *Mixtral of Experts*, (2024), https://arxiv.org/abs/2401.04088v1.

Jumper J., Evans R., Pritzel A., i in., *Highly accurate protein structure prediction with AlphaFold*, „Nature 2021 596:7873” t. 596 nr 7873 (2021), DOI: 10.1038/s41586-021-03819-2, https://www.nature.com/articles/s41586-021-03819-2.

Kosar V., *Cross-Attention in Transformer Architecture* [na:] https://vaclavkosar.com/ml/cross-attention-in-transformer-architecture, 30 grudnia 2022 r., dostęp 18 marca 2024 r.

Lubbad M., *GPT-4 Parameters: Unlimited guide NLP’s Game-Changer* [na:] https://medium.com/@mlubbad/the-ultimate-guide-to-gpt-4-parameters-everything-you-need-to-know-about-nlps-game-changer-109b8767855a, 19 marca 2023 r., dostęp 22 marca 2024 r.

*NVIDIA (NVDA) - Market capitalization* [na:] https://companiesmarketcap.com/nvidia/marketcap/, dostęp 18 marca 2024 r.

Openai A.R., Openai K.N., Openai T.S., Openai I.S., *Improving Language Understanding by Generative Pre-Training*, (2018), https://gluebenchmark.com/leaderboard.

Patel D., Gerald W., *GPT-4 Architecture, Infrastructure, Training Dataset, Costs, Vision, MoE* [na:] https://www.semianalysis.com/p/gpt-4-architecture-infrastructure, 10 lipca 2023 r., dostęp 24 marca 2024 r.

Radford A., Wu J., Child R., i in., *Language Models are Unsupervised Multitask Learners*, (2019), https://github.com/codelucas/newspaper.

Shaw P., Uszkoreit J., Vaswani A., *Self-Attention with Relative Position Representations*, „NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference” t. 2 (2018), DOI: 10.18653/v1/n18-2074, https://arxiv.org/abs/1803.02155v2.

Su J., Ahmed M., Lu Y., i in., *RoFormer: Enhanced Transformer with Rotary Position Embedding*, „Neurocomputing” t. 568 (2021), DOI: 10.1016/j.neucom.2023.127063, https://arxiv.org/abs/2104.09864v5.

Tay Y., Dehghani M., Tran V.Q., i in., *UL2: Unifying Language Learning Paradigms*, (2022), https://arxiv.org/abs/2205.05131v3.

Vaswani A., Shazeer N., Parmar N., i in., *Attention Is All You Need*, „Advances in Neural Information Processing Systems” t. 2017-December (2017), https://arxiv.org/abs/1706.03762v7.

Spis tabel

**Tabela 1. Tabela testowa** 16

Spis wykresów

Wykres 1: Liczba parametrów modeli w czasie 14

Spis rysunków

[Rysunek 1: Architektura transformera 8](#_Toc162202970)

[Rysunek 2: Transformery rzadkie 12](#_Toc162202971)

[**Rysunek 3. Rysunek testowy** 16](#_Toc162202972)

Streszczenie

1. A. Dosovitskiy i in., *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*, „ICLR 2021 - 9th International Conference on Learning Representations” (2020), https://arxiv.org/abs/2010.11929v2. [↑](#footnote-ref-1)
2. J. Jumper i in., *Highly accurate protein structure prediction with AlphaFold*, „Nature 2021 596:7873” t. 596 nr 7873 (2021), DOI: 10.1038/s41586-021-03819-2, https://www.nature.com/articles/s41586-021-03819-2. [↑](#footnote-ref-2)
3. A. Gulati i in., *Conformer: Convolution-augmented Transformer for Speech Recognition*, „Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH” t. 2020-October (2020), DOI: 10.21437/Interspeech.2020-3015, https://arxiv.org/abs/2005.08100v1. [↑](#footnote-ref-3)
4. L. Chen i in., *Decision Transformer: Reinforcement Learning via Sequence Modeling*, „Advances in Neural Information Processing Systems” t. 18 (2021), https://arxiv.org/abs/2106.01345v2. [↑](#footnote-ref-4)
5. A. Vaswani i in., *Attention Is All You Need*, „Advances in Neural Information Processing Systems” t. 2017-December (2017), https://arxiv.org/abs/1706.03762v7. [↑](#footnote-ref-5)
6. D. Bahdanau, K.H. Cho, Y. Bengio, *Neural Machine Translation by Jointly Learning to Align and Translate*, „3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings” (2014), https://arxiv.org/abs/1409.0473v1. [↑](#footnote-ref-6)
7. *NVIDIA (NVDA) - Market capitalization* [na:] https://companiesmarketcap.com/nvidia/marketcap/, dostęp 18 marca 2024 r. [↑](#footnote-ref-7)
8. V. Kosar, Cross-Attention in Transformer Architecture https://vaclavkosar.com/ml/cross-attention-in-transformer-architecture, 30 grudnia 2022 r., dostęp 18 marca 2024 r. [↑](#footnote-ref-8)
9. A.R. Openai i in., *Improving Language Understanding by Generative Pre-Training*, (2018), https://gluebenchmark.com/leaderboard. [↑](#footnote-ref-9)
10. Y. Tay i in., *UL2: Unifying Language Learning Paradigms*, (2022), https://arxiv.org/abs/2205.05131v3. [↑](#footnote-ref-10)
11. A. Radford i in., *Language Models are Unsupervised Multitask Learners*, (2019), https://github.com/codelucas/newspaper. [↑](#footnote-ref-11)
12. T.B. Brown i in., *Language Models are Few-Shot Learners*, „Advances in Neural Information Processing Systems” t. 2020-December (2020), https://arxiv.org/abs/2005.14165v4. [↑](#footnote-ref-12)
13. R. Child i in., *Generating Long Sequences with Sparse Transformers*, (2019), https://arxiv.org/abs/1904.10509v1. [↑](#footnote-ref-13)
14. D. Patel, W. Gerald, *GPT-4 Architecture, Infrastructure, Training Dataset, Costs, Vision, MoE* [na:] https://www.semianalysis.com/p/gpt-4-architecture-infrastructure, 10 lipca 2023 r., dostęp 24 marca 2024 r. [↑](#footnote-ref-14)
15. A.Q. Jiang i in., *Mixtral of Experts*, (2024), https://arxiv.org/abs/2401.04088v1. [↑](#footnote-ref-15)
16. P. Shaw, J. Uszkoreit, A. Vaswani, *Self-Attention with Relative Position Representations*, „NAACL HLT 2018 - 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference” t. 2 (2018), DOI: 10.18653/v1/n18-2074, https://arxiv.org/abs/1803.02155v2. [↑](#footnote-ref-16)
17. J. Su i in., *RoFormer: Enhanced Transformer with Rotary Position Embedding*, „Neurocomputing” t. 568 (2021), DOI: 10.1016/j.neucom.2023.127063, https://arxiv.org/abs/2104.09864v5. [↑](#footnote-ref-17)
18. M. Lubbad, GPT-4 Parameters: Unlimited guide NLP’s Game-Changer https://medium.com/@mlubbad/the-ultimate-guide-to-gpt-4-parameters-everything-you-need-to-know-about-nlps-game-changer-109b8767855a, 19 marca 2023 r., dostęp 22 marca 2024 r. [↑](#footnote-ref-18)