The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

Cybersecurity and Law

2026 Nr 1 (15)

DOI: 10.35467/cal/243543

Duże generatywne modele językowe jako wsparcie pracy badawczej nad tekstem

Large Language Models as Support for Research Work on Text

Tomasz WALKOWIAK
Politechnika Wrocławska
ORCID: 0000-0002-7749-4251

Streszczenie

W artykule przedstawiono zastosowanie generatywnych modeli językowych (GMJ) w badaniach nad tekstem, omawiając ich działanie, metody przygotowania pobudzeń oraz ograniczenia, w tym halucynacje. Zaproponowano iteracyjne doskonalenie pobudzeń, automatyzację analizy korpusów oraz wykorzystanie metody „model jako sędzia". GMJ mogą znacząco przyspieszyć analizę i ograniczyć pracę manualną, jednak w wyspecjalizowanych zadaniach ustępują dedykowanym narzędziom NLP. Ich skuteczne wykorzystanie wymaga weryfikacji źródeł, oceny eksperckiej i integracji z klasycznymi metodami badawczymi.

Słowa kluczowe

przetwarzanie języka naturalnego, generatywne modele językowe, inżynieria pobudzeń, analiza korpusów tekstowych

Abstract

This article presents the use of large language models (LLMs) in text-based research, discussing their operation, prompt design methods, and limitations, including hallucinations. It proposes iterative prompt refinement, automated corpus processing, and the "model-as-a-judge" approach. LLMs can significantly accelerate analysis and reduce manual effort, but dedicated NLP tools still outperform them in specialized tasks. Their effective use requires source verification, expert evaluation, and integration with traditional research methods.

Keywords

natural language processing, large language models, prompt engineering, text corpus analysis

Praca została sfinansowana w ramach inwestycji „CLARIN ERIC – European Research Infrastructure Consortium: Common Language Resources and Technology Infrastructure" (okres realizacji: 2024–2026), finansowanej przez Ministerstwo Nauki i Szkolnictwa Wyższego w ramach programu „Wsparcie udziału polskich zespołów naukowych w międzynarodowych projektach infrastruktury badawczej", umowa nr 2024/WK/01.

The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

Wprowadzenie

Tekst w języku naturalnym stanowi przedmiot badań w wielu dziedzinach nauk humanistycznych, jak i społecznych. Automatyczne przetwarzanie języka stanowi kluczowy element umożliwiający efektywną analizę, interpretację i porównywanie dużych korpusów tekstowych. Technologia przetwarzania języka naturalnego (ang. Natural Language Processing, NLP) ma swoją historię sięgającą połowy XX wieku. Początkowo badania w tym obszarze koncentrowały się na metodach regułowych analizy składni i tłumaczenia maszynowego, opartych na formalnych modelach gramatycznych. W latach 80. i 90. XX wieku nastąpił zwrot ku podejściom statystycznym, wykorzystującym analizę dużych korpusów tekstowych oraz metody probabilistyczne, co umożliwiło bardziej elastyczne modelowanie zjawisk językowych.

Dynamiczny rozwój sieci neuronowych i uczenia głębokiego w drugiej dekadzie XXI wieku zapoczątkował erę modeli reprezentacji języka, takich jak word2vec[1] czy BERT[2], które umożliwiły maszynom przetwarzanie znaczenia w sposób kontekstowy. Powstały narzędzia pozwalające na automatyczną analizę tekstów pod kątem emocji, tematów, struktur gramatycznych oraz relacji semantycznych. W naukach humanistycznych i społecznych metody NLP znajdują zastosowanie m.in. w analizie dyskursu, badaniach narracyjnych, studiach nad emocjami, rekonstrukcji sieci pojęć oraz identyfikacji trendów ideowych w długich okresach historycznych. Skuteczne wykorzystanie tych narzędzi wymaga jednak zarówno odpowiedniego zaplecza technologicznego jak i wiedzy dotyczącej ich możliwości oraz ograniczeń.

Pomimo rosnącej dostępności narzędzi językowych, nie wszystkie problemy badawcze związane z tekstem mogą zostać rozwiązane. Wynika to zarówno z dużej różnorodności działań badawczych, jak i z wysokich kosztów tworzenia nowych narzędzi. Tradycyjne podejście do zastosowania sztucznej inteligencji (SI) w zadaniach przetwarzania języka naturalnego wymaga bowiem dużych zbiorów treningowych – często liczących kilkanaście tysięcy przykładów – do wytrenowania nowego modelu. Proces ten jest kosztowny, gdyż angażuje badaczy z dziedziny nauk społecznych i humanistycznych do przygotowania korpusów treningowo-testowych, specjalistów w zakresie uczenia maszynowego do opracowania modeli SI oraz informatyków do udostępnienia ich w formie użytecznej dla badaczy.

Sytuacja ta uległa istotnej zmianie pod koniec 2022 roku, wraz z pojawieniem się dużych generatywnych modeli językowych (GMJ, ang. Large Language Models), które otworzyły nowe możliwości w zakresie rozumienia, przetwarzania i generowania tekstu. Należy zwrócić uwagę na obserwacje wielu badaczy, że modele językowe potrafią wiele, jednak w przypadku konkretnych zadań bardzo rzadko przewyższają dedykowane modele SI[3]. Innymi słowy, jeśli problem badawczy dotyczy zagadnienia, dla którego już opracowano specjalistyczne narzędzie NLP (np. analizę emocjonalną tekstu), to w większości zastosowań takie dedykowane narzędzie daje lepsze rezultaty jakościowe niż wykorzystanie dużych modeli generatywnych językowych.

---

[1] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, J. Dean, Distributed Representations of Words and Phrases and Their Compositionality, in Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2 (NIPS '13), Vol. 2, Curran Associates Inc., Red Hook, NY, USA 2013, s. 3111–3119.

[2] J. Devlin, M-W. Chang, K. Lee, K. Toutanova, BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, in Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1, Minneapolis, Minnesota, Association for Computational Linguistics 2019, s. 4171–4186.

[3] J. Kocoń, et. al. ChatGPT: Jack of all trades, master of none, Information Fusion, 99, C, Nov 2023.

The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

Niniejszy artykuł poświęcony jest zagadnieniu wykorzystania dużych generatywnych modeli językowych do wspierania badań nad tekstem. Celem niniejszej pracy nie jest obszar tworzenia i wspomagania pisania, czyli generowania wstępnych wersji artykułów, raportów, podpowiedzi literaturowych, korekty gramatyki, spójności i jasności tekstów akademickich ani parafrazowania i tłumaczenia przydatnego w badaniach porównawczych tekstów z różnych języków, lecz wykorzystanie GMJ w analizie tekstów, w tym ekstrakcja informacji czy kodowanie danych jakościowych (przypisaniu etykiety, tzw. „kodu", która odzwierciedla np. znaczenie lub temat tekstu).

Praca składa się z pięciu rozdziałów. Pierwsza część wprowadza w tematykę przetwarzania języka naturalnego oraz omawia rozwój narzędzi i technologii stosowanych w naukach humanistycznych i społecznych. Druga część poświęcona jest dużym generatywnym modelom językowym, w tym zasadzie ich działania, tokenizacji, osadzaniu tokenów, architekturze warstw sieci neuronowych, generowaniu odpowiedzi oraz procesowi uczenia. Trzecia część omawia zasady przygotowywania zapytań do modeli językowych, metody konstruowania pobudzeń, techniki minimalizacji halucynacji oraz przetwarzanie większych korpusów tekstowych z wymuszeniem określonego formatu danych wyjściowych. Czwarta część zawiera rekomendowane podejście do pracy z dużymi modelami językowymi, omawia ograniczenia ich zastosowania, iteracyjny proces doskonalenia pobudzeń, metody oceny wyników, w tym podejście „model jako sędzia". W ostatniej zawarto podsumowanie.

Generatywne modele językowe – zasady działania

Współczesne generatywne modele językowe są złożonymi, wielowarstwowymi sieciami neuronowymi. Ich działanie określane jest przez architekturę (budowę warstw, liczbę parametrów – wag oraz liczbę warstw) oraz przez wartości wag ustalane w procesie uczenia. Sieć o tej samej architekturze i liczbie wag, np. Llama 3.1-70B[4] (sieć o 70 miliardach wag – wolnych parametrach ustalanych w procesie uczenia), ale uczona na innych danych lub w innej kolejności, będzie dawała inne wyniki.

Działanie modelu polega na przewidywaniu prawdopodobieństwa następnego fragmentu tekstu na podstawie aktualnego zapytania.[5] Sieć nie operuje bezpośrednio na słowach ani pojedynczych literach – na jej wejście podawany jest ciąg tzw. tokenów, które mogą być pojedynczymi znakami, fragmentami słów lub całymi słowami, w zależności od użytego tokenizatora. Po wyznaczeniu prawdopodobieństwa kolejnych tokenów wybierany jest jeden z nich (metody wyboru następnego tokenu opisano poniżej) i dodawany na koniec danych wejściowych. Proces ten powtarza się wielokrotnie, aż model stwierdzi koniec wypowiedzi (zwracając dedykowany token końca) lub zostanie osiągnięta maksymalna liczba tokenów wyjściowych (parametr zapytania do sieci). Jak widać, proces jest autoregresywny: każde wygenerowane wyjście staje się częścią danych wejściowych w kolejnej iteracji, a model generuje tekst krok po kroku. Takie rozwiązanie powoduje, że koszt działania modelu (czas generowania i zarazem zużycie energii elektrycznej) jest wprost proporcjonalny do długości wygenerowanego tekstu.

Jak już wspomniano sieć nie operuje na literach czy słowach, ale na fragmentach słów. Przykład tokenizacji tekstu przez model Mixtral-8x7B[6] pokazany jest na Rys. 1. Jak można zauważyć, tokeny wejściowe nie składają się z całych słów, lecz często z ich

[4] A. Grattafiori, et al., The LLaMA 3 Herd of Models, CoRR, 2024, s. 2407.
[5] A. Vaswani, et.al., Attention Is All You Need, in Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS '17), 6000–6010, Curran Associates Inc., Red Hook, NY 2017.
[6] Jiang, A. Q., et al., Mixtral of Experts, CoRR, 2024.

The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

fragmentów. Ponadto mogą pojawiać się specjalne tokeny, oznaczające np. początek (token <s> w przykładzie) lub koniec tekstu.

Rys. 1 Tokenizacja tekstu: „Współcześni kronikarze wspominają, iż z wiosny szarańcza w niesłychanej ilości wyroiła się z Dzikich Pól" przez model Mixtral-8x7B.

'<s>', '▁W', 'sp', 'ół', 'c', 'ześ', 'ni', '▁k', 'ron', 'ik', 'ar', 'ze', '▁wsp', 'omin', 'ają', ',', '▁i', 'ż', '▁z', '▁w', 'ios', 'ny', '▁sz', 'ara', 'ń', 'c', 'za', '▁w', '▁n', 'ies', 'ł', 'ych', 'ane', 'j', '▁il', 'o', 'ści', '▁wy', 'ro', 'i', 'ła', '▁się', '▁z', '▁D', 'z', 'ik', 'ich', '▁P', 'ól'

Tokeny zaczynające się od _ oznaczają początek wyrazu. Natomiast token <s> informuje o początku tekstu.

Sieci neuronowe operują na liczbach, dlatego podanie tokenów (czyli ciągów znaków lub słów) na wejście sieci wymaga ich zamiany na wielowymiarowe wektory. Proces ten nazywamy osadzaniem (z ang. embedding). Przypisanie konkretnego wektora do danego tokenu jest zwykle wynikiem procesu uczenia sieci - wektory te stanowią część jej parametrów i są optymalizowane razem z resztą modelu. W efekcie otrzymujemy macierz osadzeń. Każdy wiersz tej macierzy odpowiada jednemu tokenowi ze słownika modelu, a indeks tokenu służy do pobrania jego wektora. Dodatkowo, aby zachować informację o pozycji tokenu w tekście wejściowym, osadzenie tokenu jest sumowane z osadzeniem pozycyjnym. Tak powstała reprezentacja numeryczna stanowi dane wejściowe dla sieci neuronowej.

Sieć składa się z wielu warstw działających według tej samej zasady (na przykład 32 warstwy w modelu Mixtral 8×7B lub 80 w modelu Llama 3.1 70B). Każda warstwa otrzymuje na wejściu ciąg osadzeń (czyli wielowymiarowych wektorów reprezentujących tokeny) i generuje ciąg nowych osadzeń o tej samej długości.

Proces przetwarzania w każdej warstwie można opisać w trzech etapach[7]:

1. Obliczanie uwag (ang. attention): dla każdego tokenu wyznaczana jest ważona suma wektorów osadzeń innych tokenów. Wagi tej sumy są określane przez tzw. mechanizm uwagi (ang. attention mechanism), oparty na wyuczonych macierzach (wagach sieci).
2. Przekształcenie nieliniowe: wyniki działania mechanizmu uwagi są przetwarzane przez wielowarstwową sieć neuronową (ang. feed-forward network), której parametry również są wyuczone. Te same wagi są stosowane dla wszystkich tokenów w sekwencji.
3. Normalizacja: na końcu uzyskane wektory są poddawane normalizacji w taki sposób, aby każdy z nich miał średnią równą 0 i odchylenie standardowe równe 1; operacja ta wykonywana jest osobno dla każdego osadzenia.

W ten sposób każda warstwa stopniowo przekształca reprezentacje tokenów wejściowych w coraz bardziej złożone reprezentacje, które odzwierciedlają nie tylko znaczenie pojedynczych tokenów, lecz także relacje i zależności między nimi w kontekście całego tekstu.

Ostatnie osadzenie (reprezentujące ostatni token z tekstu wejściowego), z ostatniej warstwy sieci jest wykorzystywane do wyznaczania prawdopodobieństwa wystąpienia kolejnego tokenu (czyli kolejnej części odpowiedzi generowanej przez sieć) przy użyciu pojedynczej warstwy liniowej i dedykowanej funkcji aktywacji, pozwalającej na uzyskanie odpowiedzi mającej cechy prawdopodobieństwa (wartości z zakresu 0-1 i suma równa 1).

[7] Zob. A. Vaswani, op.cit., s. 5.

The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

Pozostaje pytanie, który token wybrać. Teoretycznie, aby uzyskać najbardziej prawdopodobną odpowiedź, należałoby przeanalizować wszystkie możliwe sekwencje tokenów i wybrać tę o największym łącznym prawdopodobieństwie. W praktyce jest to jednak nieosiągalne, ponieważ liczba analizowanych sekwencji rośnie wykładniczo wraz z długością generowanej odpowiedzi. Dlatego w praktyce stosuje się heurystyki wyboru tokenów. Najprostszą metodą jest wybór tokenu o najwyższym prawdopodobieństwie na każdym etapie generowania. Wadą tego podejścia jest to, że dla danego zapytania sieć zawsze wygeneruje tę samą odpowiedź. W zadaniach, w których pożądana jest różnorodność (np. parafraza tekstu), takie zachowanie jest niepożądane. Aby wprowadzić różnorodność w tekście wyjściowym, stosuje się generowanie z elementem losowości, wykorzystując pseudolosowe generatory liczb dostępne w systemach komputerowych.

W tym przypadku token jest losowany z uwzględnieniem prawdopodobieństw wyznaczonych przez sieć: tokeny bardziej prawdopodobne mają większą szansę na pojawienie się w odpowiedzi.

Poziom losowości można regulować przede wszystkim dwoma parametrami, temperaturą i top-p[8]:

1. Top-p określa, jaką część najbardziej prawdopodobnych tokenów uwzględniamy w losowaniu. Na przykład top-p = 0,5 oznacza, że losujemy token spośród tych, których łączne prawdopodobieństwo wynosi 0,5, pomijając pozostałe mniej prawdopodobne.
2. Temperatura modyfikuje same prawdopodobieństwa tokenów. Wartość 1 pozostawia je bez zmian. Wartości mniejsze niż 1 zwiększają prawdopodobieństwo bardziej prawdopodobnych tokenów (aż do sytuacji temperatury 0, gdzie zawsze wybierany jest tylko jeden token). Zwiększanie temperatury powyżej 1 wyrównuje prawdopodobieństwa, co sprawia, że mniej prawdopodobne tokeny stają się bardziej możliwe do wyboru. W praktyce większość systemów ogranicza maksymalną temperaturę do 2.

Proces uczenia modeli generatywnych, ze względu na ich bardzo dużą liczbę parametrów, jest złożony i wymaga ogromnej ilości danych oraz znacznych zasobów obliczeniowych. W uproszczeniu można go podzielić na trzy główne etapy:

1. Uczenie wstępne (ang. pre-training). Na tym etapie model uczy się ogólnego rozumienia języka. Sieci przedstawia się ogromną liczbę fragmentów tekstu i trenuje ją w zadaniu przewidywania kolejnego tokenu - czyli kontynuacji danego tekstu. Etap ten nie wymaga danych etykietowanych, ponieważ część tekstu można potraktować jako wejście, a kolejną część jako oczekiwane wyjście. Uczenie wstępne wymaga jednak bardzo dużej ilości różnorodnych danych tekstowych. Trening modelu wyłącznie na danych w jednym języku jest w praktyce trudny z powodu zbyt małej liczby dostępnych tekstów. Dlatego stosuje się zbiory wielojęzyczne.
2. Dostrajanie[9] (ang. Supervised Fine-Tuning, SFT). Na tym etapie model jest dostrajany do wykonywania konkretnych zadań językowych. Proces polega na uczeniu modelu na parach (pytanie, odpowiedź) lub (polecenie, rezultat). Dane uczące są przygotowywane ręcznie lub półautomatycznie (z wykorzystaniem innych modeli językowych). Celem tego etapu jest nauczenie modelu reagowania

[8] A. Fan, M, Lewis, Y. Dauphin, Hierarchical Neural Story Generation, in Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, Association for Computational Linguistics 2018, s. 889–898,

[9] L. Ouyang, et.al., Training Language Models to Follow Instructions with Human Feedback, in Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS '22), Curran Associates Inc., Red Hook, NY 2022, s. 27730–27744.

The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

w sposób zgodny z oczekiwaniami i poprawny językowo. Im bardziej różnorodne są przykłady w zbiorze uczącym (np. streszczanie tekstów, odpowiadanie na pytania, tłumaczenia, analiza sentymentu), tym bardziej wszechstronny staje się model.

3. Wychowanie. Ten etap ma na celu nauczenie modelu generowania odpowiedzi nie tylko poprawnych, ale także pożądanych z punktu widzenia twórców – tj. najczęściej uprzejmych, pomocnych, zgodnych z kontekstem i niewywołujących kontrowersji. Jedną z metod jest przygotowywanie dla podanego pobudzenia dwóch odpowiedzi: pożądanej i niepożądanej, a następnie minimalna modyfikacja wag sieci w taki sposób, aby pożądana odpowiedź była oceniana przez model jako bardziej prawdopodobna niż ta niepożądana.

Efektem wszystkich trzech etapów treningu są wagi definiujące działanie generatywnego modelu językowego, pozwalające zarówno rozumieć, jak i generować tekst zbliżony do języka naturalnego, udzielać poprawnych odpowiedzi na zapytania oraz postępować zgodnie z intencjami twórców modelu.

Zasady przygotowywania zapytań

Jak wspomniano we wstępie, odpowiedź modelu językowego zależy zarówno od jego wag, jak i od tekstu podanego na wejście. Aby uzyskać pożądaną odpowiedź, można zastosować dwie grupy metod:

1. Douczyć model do własnych potrzeb (czyli zmodyfikować jego wagi)
2. Odpowiednio przygotować pobudzenie (z ang. prompt).

Douczenie modelu wymaga specjalistycznego sprzętu, dużych zbiorów danych oraz znacznych zasobów obliczeniowych, dlatego nie jest procesem dostępnym dla zwykłych użytkowników modeli językowych. W praktyce jedynym sposobem dostosowania działania modelu jest odpowiednie przygotowanie zapytania. Proces ten wymaga zrozumienia roli pobudzenia w działaniu sieci i parametrów z tym związanych. Przy ustawieniu temperatury na zero odpowiedzi modelu będą zależeć wyłącznie od pobudzenia. Należy zwrócić uwagę, że podczas użytkowania modeli przez interfejsy webowe konwersacja z modelem organizowana jest w „chaty" czy „rozmowy". Model na wejściu nie dostaje tylko aktualnego zapytania, ale także historię całej rozmowy, włącznie z własnymi odpowiedziami. Odpowiedź zależy zatem również od całej historii komunikacji. Modele mają silną tendencję do sugerowania się konwersacją (tak są uczone). W efekcie często obserwuje się „uparcie" modelu co do pewnych wcześniej przedstawionych faktów. Ponadto modele posiadają tzw. pobudzenie systemowe. Jest to tekst opracowany przez twórców modelu lub osoby udostępniające model, który zawsze dodawany jest na początek rozmowy. Zawiera on często aktualną datę oraz prośbę do modelu o odpowiedź w sposób miły i grzeczny. Bardzo często ten tekst nie jest znany użytkownikowi modelu. Literatura[10] wyróżnia dwie główne grupy metod przygotowywania pobudzeń dla modeli językowych: „uczenie bezprzykładowe" (ang. zero-shot) oraz „uczenie wieloprzykładowe" (ang. few-shot). Nazwy te mogą być nieco mylące, ponieważ w klasycznym uczeniu maszynowym „uczenie" oznacza proces modyfikacji wag sieci, podczas gdy w omawianych przypadkach wag modelu się nie zmienia - zamiast tego konstruuje się odpowiednie pobudzenie, które kieruje modelem tak, aby lepiej zrozumiał intencję pytającego i zwiększył prawdopodobieństwo udzielenia pożądanej odpowiedzi.

[10] T.B. Brown, et.al. Language Models Are Few-Shot Learners, in Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS '20), Curran Associates Inc., Red Hook, NY, Article 159, 2020, s. 1877–1901.

The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

Przykładem zadania bezprzykładowego jest polecenie w stylu: „Oceń nastawienie emocjonalne poniższego zdania.” Model wykorzystuje wówczas swoją wcześniejszą wiedzę - zdobytą w trakcie dostrajania nadzorowanego oraz ewentualnego wychowania – aby wygenerować odpowiedź zgodną z oczekiwaniami użytkownika. Takie podejście sprawdza się bardzo dobrze w typowych zadaniach przetwarzania języka naturalnego, takich jak streszczanie, ekstrakcja słów kluczowych czy rozpoznawanie nazw własnych. Dzieje się tak, ponieważ dla tego typu zadań istnieje wiele publicznie dostępnych zbiorów danych, które są powszechnie wykorzystywane podczas uczenia GMJ (w etapie dostrajania).

W przypadku zadań mniej typowych (np. wymagających specyficznej interpretacji, wiedzy) skuteczność podejścia bezprzykładowych spada. Wówczas konieczne jest zastosowanie uczenia wieloprzykładowego, czyli podanie w pobudzeniu kilku przykładów par pytanie–odpowiedź, które pokazują modelowi oczekiwany sposób formułowania odpowiedzi.

Jednym z kluczowych ograniczeń użycia dużych modeli językowych (GMJ) jest zjawisko tzw. halucynacji (ang. hallucination), polegające na generowaniu informacji pozornie wiarygodnych, lecz faktycznie nieprawdziwych lub nieistniejących w analizowanym kontekście. Halucynacje mogą przybierać różne formy – od drobnych zniekształceń faktów po całkowicie fikcyjne dane, źródła lub interpretacje. Wynikają one z probabilistycznej natury modeli językowych, które generują tekst na podstawie statystycznych zależności między słowami w danych treningowych. GMJ nie posiadają rzeczywistego rozumienia semantycznego ani dostępu do wiarygodnych źródeł w momencie generacji, przez co mogą tworzyć logicznie spójne, lecz merytorycznie błędne wypowiedzi. Zjawisko to jest szczególnie problematyczne w zastosowaniach badawczych, gdzie oczekuje się wysokiej precyzji i weryfikowalności informacji.

Ograniczenie ryzyka halucynacji, można osiągnąć stosując następujące zasady:

1. Precyzyjne formułowanie pobudzeń – doprecyzowanie poleceń i ograniczenie przestrzeni interpretacyjnej modelu (np. przez wyraźne instrukcje, zakazy „domyślania” lub wymóg cytowania źródeł).
2. Użycie kontekstu zewnętrznego – zasilanie modelu rzeczywistymi fragmentami tekstów, danymi źródłowymi lub streszczeniami, zamiast polegania wyłącznie na jego pamięci językowej.
3. Weryfikacja krzyżowa – porównywanie wyników uzyskanych z różnych modeli lub pobudzeni.
4. Ocena ekspercka – ręczna kontrola kluczowych fragmentów wyników.

Wykorzystanie generatywnych modeli językowych do analizy pojedynczych tekstów jest interesujące z punktu widzenia poznawczego, jednak w wielu przypadkach takie zadania może wykonać sam badacz. Prawdziwa zaleta modeli językowych ujawnia się dopiero przy przetwarzaniu dużych zbiorów tekstów, gdzie umożliwiają one analizę całego korpusu bez konieczności wielogodzinnej pracy człowieka.

Proces ten wymaga oczywiście odpowiedniego pobudzenia, ale także właściwego przygotowania korpusu oraz dostosowania pobudzenia w taki sposób, aby uzyskane wyniki można było łatwo przekształcić do formatu wykorzystywanego przez badacza (np. arkusza Excel lub narzędzi do analizy ilościowej).

Modele językowe potrafią przetwarzać wiele tekstów w ramach jednego zapytania – można je na przykład poprosić o wygenerowanie słów kluczowych dla kilku lub kilkunastu fragmentów naraz. Jednak jak wspomniano wcześniej, modele są silnie podatne na kontekst – treść zapytania i poprzednie interakcje mogą wpływać na wynik. Dlatego najlepszym rozwiązaniem jest analizowanie każdego tekstu osobno, w ramach oddzielnego zapytania.

The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

Współczesne modele dysponują coraz dłuższym kontekstem (maksymalną liczbą tokenów, które można podać na wejście – co też definiuje rozmiar wyjścia, bo autoregresyjność modelu sprowadza się do ciągłego przekazywania wyjścia na wejście) przekraczającym 128 tys. tokenów, dzięki czemu są w stanie przetwarzać obszerne teksty. Mimo to, jeśli zadanie nie wymaga globalnego zrozumienia treści (np. identyfikacji wszystkich postaci występujących w tekście), bardziej efektywne może być dzielenie tekstu na fragmenty i scalanie wyników. Wynika to z faktu, że przy bardzo długich kontekstach (liczących wiele tysięcy tokenów) modele mają tendencję do koncentrowania uwagi na początku i końcu tekstu, analizując środkowe części mniej dokładnie – co, swoją drogą, jest zjawiskiem typowym również dla ludzi.

Drugim wyzwaniem przy przetwarzaniu dużych korpusów jest format odpowiedzi generowanej przez model. Modele językowe są szkolone głównie do prowadzenia konwersacji, dlatego mają tendencję do dodawania elementów takich jak: „Świetnie - oto rozbudowana wersja tekstu…" na początku odpowiedzi, lub „Jeśli chcesz, mogę też…" na końcu.

W przypadku pojedynczych zapytań można takie fragmenty usunąć ręcznie, natomiast w analizie masowej należy odpowiednio zmodyfikować pobudzenie, aby model zwracał dane w maszynowo przetwarzalnym formacie. Dobrym rozwiązaniem jest dopisanie w treści instrukcji wprost takiej informacji, np.: „Nie dawaj żadnych dodatkowych komentarzy ani wyjaśnień. Zwróć wyłącznie dane w określonym formacie." Takie doprecyzowanie znacząco zwiększa spójność i jakość wyników uzyskiwanych w analizie automatycznej.

Format odpowiedzi powinien umożliwiać jej łatwą analizę za pomocą dostępnych narzędzi programistycznych. Dodatkowo długość odpowiedzi ma bezpośredni wpływ na koszt użycia modelu. Dlatego jeśli analizowane zadanie sprowadza się do zwrócenia etykiety z ustalonej listy klas (np. przypisania tekstu do jednej z kilku kategorii), dobrym rozwiązaniem jest poproszenie modelu o zwrócenie liczby lub pojedynczego symbolu reprezentującego daną klasę. W przypadku, gdy klas jest mniej niż 10, można użyć cyfr 0–9. Wówczas odpowiedź ma długość jednego tokenu, co minimalizuje koszt i ułatwia dalsze przetwarzanie.

W sytuacjach, gdy konieczne jest zwrócenie informacji ustrukturyzowanej, zaleca się użycie formatu JSON, który jest powszechnie rozumiany zarówno przez modele językowe (często pojawia się w danych uczących), jak i przez narzędzia programistyczne. Przykład pobudzenia: „Zwróć listę słów kluczowych w formacie listy JSON."

Wykorzystanie formatów o zdefiniowanej i prostej strukturze można dodatkowo wymusić poprzez zastosowanie trybu ukierunkowanego generowania (ang. guided output). Tryb ten, obsługiwany przez wiele platform udostępniających modele językowe, polega na formalnym opisie struktury wyjścia przy użyciu standardu JSON Schema[11].

Opis w formie JSON Schema działa jak prosta gramatyka - system kontrolujący generowanie nie dopuści do wygenerowania wyjścia niezgodnego z określonym formatem. Na przykład: jeśli w strukturze JSON wymagane jest oddzielanie obiektów przecinkiem, model będzie mógł wybrać tylko token przecinka, a jeśli wartość pola ma być logiczna, w odpowiedzi pojawi się wyłącznie tekst: „true" lub „false". Takie podejście zapewnia pełną gwarancję poprawności struktury odpowiedzi, co znacząco ułatwia jej późniejsze automatyczne przetwarzanie. Oczywiście, poprawność strukturalna nie oznacza poprawności merytorycznej - model może zwrócić formalnie poprawny, lecz semantycznie błędny wynik.

[11] https://json-schema.org [dostęp: 14.10.2025]

The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

Zastosowanie modeli językowych GMJ wykracza poza prosty schemat interakcji typu *zapytanie–odpowiedź*. Współczesne interfejsy użytkownika, udostępniające modele tego typu, nie ograniczają się do wykorzystania samego modelu językowego. Przykładowo, zapytania kierowane za pośrednictwem aplikacji OpenAI[12] mogą inicjować szereg zewnętrznych procesów, takich jak przeszukiwanie zasobów internetowych. W konsekwencji generowana przez model odpowiedź jest formułowana w odniesieniu do informacji uzyskanych z tych dodatkowych źródeł. Zjawisko to stanowi przejaw tzw. agentowości[13] modeli sztucznej inteligencji**,** polegającej na zdolności systemu do samodzielnego korzystania z narzędzi, planowania działań oraz wnioskowania na podstawie pozyskanych danych.

Wykorzystanie modeli o cechach agentowych w badaniach naukowych tekstów rodzi jednak istotne trudności metodologiczne. Badacz dysponuje ograniczoną kontrolą nad zakresem i sposobem pozyskiwania informacji przez system, co utrudnia zachowanie przejrzystości procesu analizy oraz replikowalności wyników[14]. Z tego względu zastosowania tego rodzaju nie mieszczą się w ramach niniejszego opracowania.

Rekomendowane podejście

Wiedząc, jak działają duże generatywne modele językowe i znając ich przydatność w analizie danych tekstowych, należy opracować procedurę postępowania w przypadku potrzeby przetwarzania większych korpusów tekstów. Trzeba jednak zwrócić uwagę na dwa istotne ograniczenia tych modeli. Po pierwsze, GMJ mają trudności z wykonywaniem zadań wymagających dokładnego zliczania elementów lub wyznaczania statystyk. Nie powinno się więc wykorzystywać modeli generatywnych do zadań analitycznych tego typu. Lepszym podejściem jest użycie modeli do wyznaczania prostych cech elementarnych, które następnie mogą być analizowane za pomocą klasycznych metod statystycznych. Przykładowo, można wykorzystać GMJ do określania emocjonalnego wydźwięku krótszych fragmentów tekstu, a następnie przeanalizować uzyskane dane za pomocą prostych skryptów w Pythonie lub popularnych narzędzi, takich jak Excel. GMJ mogą być także pomocne w generowaniu kodu wspomagającego analizę wyników uzyskanych w wyniku ich wcześniejszego działania.

Drugie ograniczenie wiąże się z rozmiarem analizowanych próbek tekstu. Wielkość próbki powinna być adekwatna do charakteru realizowanego zadania. Z jednej strony należy dążyć do jej minimalizacji, z drugiej – zapewnić wystarczający kontekst umożliwiający prawidłowe zrozumienie treści i wykonanie polecenia zdefiniowanego w zapytaniu.

Przykładowo, w przypadku zadań polegających na identyfikacji nazw własnych w tekście, wystarczający może być pojedynczy akapit, ponieważ zrozumienie krótkiego fragmentu dostarcza kontekstu niezbędnego do poprawnego wykonania tego typu analizy. W sytuacjach, gdy zrozumienie zadania wymaga szerszego kontekstu semantycznego, możliwe jest przekazanie modelowi dłuższego fragmentu tekstu, przy jednoczesnym ograniczeniu analizy do wybranego zakresu.

---

[12] https://chatgpt.com [dostęp: 15.10.2025]

[13] J. Wu, et.al., Agentic Reasoning: A Streamlined Framework for Enhancing LLM Reasoning with Agentic Tools, in Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*,* Vienna, Austria, Association for Computational Linguistics, 2025, s. 28489–28503.

[14] O.E. Gundersen, S. Kjensmo, State of the Art: Reproducibility in Artificial Intelligence, in Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-18), , AAAI Press 2018, s. 1644–1651.

The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

Jak omówiono w rozdziale „Zasady przygotowywania zapytań", kluczowym elementem skutecznego wykorzystania dużych modeli językowych jest opracowanie odpowiednich pobudzeń. Proces ich konstruowania ma charakter iteracyjny i obejmuje kilka etapów. W praktyce polega on na wielokrotnym formułowaniu pobudzeń, uruchamianiu modelu na wybranych przykładach, analizie uzyskanych wyników, a następnie wprowadzaniu kolejnych poprawek, aż do uzyskania satysfakcjonujących rezultatów. Jest to wykorzystanie opisywanej w literaturze metody HITL[15] (ang. human in the loop), łączące automatyczne działanie sztucznej inteligencji z aktywnym udziałem człowieka w kluczowych momentach procesu. Najistotniejszym etapem tego cyklu jest proces modyfikacji pobudzeń, który pozwala na stopniowe zwiększanie jakości i trafności odpowiedzi generowanych przez model.

Poniżej przedstawiono kilka praktyk, które mogą wspierać efektywność tego procesu:

a) Doprecyzowanie zadania – formułowanie pobudzenia w sposób jednoznaczny, z jasnym określeniem celu analizy i oczekiwanej formy odpowiedzi.

b) Spójność językowa – pobudzenie należy napisać w języku w jakim wymagana jest odpowiedź od modelu i najlepiej w tym samym języku co analizowane tekstu. Zapobiegnie to zmianą języka w odpowiedzi modelu. Model sugeruje się całym zapytaniem, a nie tylko opisem zadania i może odpowiedzieć w innym języku niż spodziewa się autor pobudzenia.

c) Dodanie przykładów typu „pobudzenie → odpowiedź" – szczególnie użyteczne w obszarach, w których model początkowo nie radzi sobie dobrze z zadaniem. Przykłady umożliwiają modelowi lepsze uchwycenie wzorców i kryteriów oceny.

d) Redefinicja zadania – ponowne sformułowanie problemu w inny sposób, w celu znalezienia opisu zadania bardziej zrozumiałego dla modelu.

e) Rozszerzenie zespołu opracowującego pobudzenia – włączenie nowych osób o różnych kompetencjach i perspektywach może prowadzić do generowania bardziej trafnych i zróżnicowanych propozycji pobudzeń.

f) Wykorzystanie dużych modeli językowych do udoskonalenia pobudzeń – możliwe jest zlecenie modelowi analizy istniejących pobudzeń i wskazania ich słabych stron, a następnie wygenerowanie ulepszonej wersji. W tym celu można wykorzystać duże (a zatem bardziej kosztowne) modele, ponieważ proces ten jest jednostkowy i nie jest wykonywany dla każdego tekstu w przetwarzanym korpusie.

Cały proces iteracyjnego doskonalenia pobudzeń można podzielić na kilka etapów:

1. Test wstępny. Na kilku przykładach należy sprawdzić, czy wybrany model GMJ radzi sobie z danym zadaniem. Na tym etapie oceny wyników dokonuje się ręcznie, bez konieczności stosowania docelowego formatu wyjściowego wymaganego do automatycznej analizy.
2. Dopracowanie pobudzenia pod kątem formatu wyjściowego. Na tych samych przykładach należy doprecyzować pobudzenie tak, aby uzyskać pożądany format odpowiedzi. Alternatywnie można zastosować podejście ukierunkowanego generowania.
3. Testy na większej próbie. W tym etapie zaleca się:

i. wybrać (najlepiej losowo) fragment analizowanego korpusu obejmujący co najmniej 100 dokumentów,

[15] S. Amershi, M. Cakmak, W.B. Knox, T. Kulesza, Power to the People: The Role of Humans in Interactive Machine Learning, AI Magazine, 35(4), Winter 2014, s. 105–112.

The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

ii. wybrać kilka modeli językowych (GMJ), biorąc pod uwagę zarówno docelowe koszty przetwarzania, jak i intuicyjne założenie, że większe modele mogą generować lepsze rezultaty,
iii. przygotować kilka wariantów pobudzeni lub zmodyfikować wcześniej opracoane pobudzenia,
iv. przeprowadzić eksperymenty porównawcze,
v. dokonać oceny uzyskanych wyników.

Kroki iii-v etapu trzeciego należy powtarzać iteracyjnie, aż do uzyskania satysfakcjonujących jakościowo rezultatów. W kolejnych iteracjach warto zwiększać liczbę analizowanych tekstów (etap i).

Ocena uzyskanych wyników powinna być możliwie najbardziej obiektywna. Jeżeli zadanie ma charakter deterministyczny, tzn. można jednoznacznie określić poprawny rezultat dla danego tekstu, należy przygotować zestaw poprawnych odpowiedzi opracowanych ręcznie lub z wykorzystaniem wyników generowanych przez model, które następnie zostaną skorygowane przez człowieka. W takich warunkach możliwe jest zastosowanie ilościowych metryk jakości. Rekomenduje się ustawienie niskiej wartości parametru temperatury, co zapewnia powtarzalność wyników eksperymentu. Przykładowo, w przypadku zadań klasyfikacyjnych – polegających na przypisywaniu tekstów do z góry określonej liczby klas – można wykorzystać miarę dokładności (ang. *accuracy*), czyli odsetek poprawnie sklasyfikowanych przypadków.

W przypadku zadań o charakterze generatywnym, takich jak konwersja tekstu do innej postaci, ocena wyników zwykle wymaga oceny eksperckiej lub ręcznej weryfikacji. W sytuacjach, gdy liczba analizowanych przykładów jest bardzo duża (rzędu tysięcy lub więcej), można rozważyć zastosowanie dużego modelu językowego pełniącego funkcję sędziego[16] (ang. *model as a judge*). Metoda ta polega na wykorzystaniu modelu językowego (najczęściej innego niż ten użyty do uzyskania odpowiedzi) do automatycznej oceny rezultatów uzyskanych w badanym zadaniu. W praktyce jest to zadanie zbliżone do opisywanego w tym rozdziale procesu – wymaga bowiem opracowania odpowiedniego pobudzenia, które tym razem służy jednak nie analizie danych, lecz ocenie jakości wyników uzyskanych przez inny model.

Zastosowanie podejścia „model jako sędzia" pozwala znacząco przyspieszyć proces ewaluacji oraz ograniczyć koszty związane z ręczną oceną dużych zbiorów wyników. W licznych badaniach wykazano, że duże GMJ (takie jak GPT-5 czy Claude 3) osiągają wysoką zgodność ocen z ocenami ludzkimi, zwłaszcza w zadaniach wymagających interpretacji semantycznej, takich jak streszczanie tekstu, ocena spójności wypowiedzi czy analiza tonu emocjonalnego.

Należy jednak podkreślić, że metoda ta nie powinna całkowicie zastępować oceny ludzkiej, ponieważ konieczna jest weryfikacja jej poprawności w kontekście konkretnego zadania. Ocena ta powinna zostać przeprowadzona ręcznie, aby uniknąć tworzenia kaskady ocen, w której działanie modeli i pobudzeń byłoby weryfikowane wyłącznie przez kolejne modele językowe.

Zjawisko halucynacji podkreśla konieczność traktowania GMJ nie jako autonomicznych narzędzi analitycznych, lecz jako systemów wspomagających proces badawczy. Działanie modelu w kontekście, tzn. dodawanie do pobudzenia analizowanego i pytania dotyczącego owego tekstu zmniejsza prawdopodobieństwo halucynacji, ale zawsze należy weryfikować efekt działania modeli. Przy analizie statystycznej uzyskanych z GMJ wyników nie jest konieczne ręczne sprawdzanie wszystkich rezultatów, ale zbadanie skuteczności działania modeli na wybranej (najlepiej

[16] J. Gu, et.al., A Survey on LLM-as-a-Judge, arXiv preprint arXiv:2411.15594, 2024.

The article is licensed under a Creative Commons Attribution-ShareAlike 4.0 Unported (CC BY-SA)

losowo) próbie. W pracy z tekstami źródłowymi model może stanowić efektywne narzędzie eksploracyjne, jednak wymaga integracji z klasycznymi metodami naukowymi – krytyczną analizą, weryfikacją źródeł i interpretacją ekspercką.

Podsumowanie

W artykule omówiono działanie, uczenie i ograniczenia generatywnych modeli językowych (GMJ) oraz metody przygotowywania zapytań. Zaproponowano schemat analizy korpusów tekstowych oparty na iteracyjnym doskonaleniu pobudzeń i automatycznej ocenie wyników. Pokazano potencjał GMJ w badaniach humanistycznych i społecznych, podkreślając potrzebę weryfikacji wyników i łączenia ich z tradycyjnymi metodami. Integracja GMJ z narzędziami analizy statystycznej może przyspieszyć badania, ograniczyć pracę manualną i umożliwić rozwój interaktywnych asystentów badawczych.

W procesie redakcji artykułu wykorzystano model GMJ, a konkretnie GPT-5. Jego rola ograniczała się wyłącznie do poprawy językowej i stylistycznej tekstu – nie generowano przy jego użyciu nowych treści. Po korekcie wykonanej przez GMJ tekst był następnie przeglądany i modyfikowany przez autora w celu usunięcia ewentualnych nieścisłości lub błędnych sformułowań wprowadzonych przez model.

Bibliografia

Amershi S. et.al., Power to the People: The Role of Humans in Interactive Machine Learning, AI Magazine 35, 4 (Winter 2014).

Brown T.B., et.al, Language models are few-shot learners, In Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS '20), Curran Associates Inc., Red Hook, NY, USA.

Devlin J., et.al., BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding, In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, Minnesota, Association for Computational Linguistics.

Fan A., et.al., Hierarchical Neural Story Generation, In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Melbourne, Australia, Association for Computational Linguistics 2018.

Gu J. et.al., A Survey on LLM-as-a-Judge, arXiv preprint arXiv:2411.15594, 2024.

Kocoń J., et.al., ChatGPT: Jack of all trades, master of none, Information Fusion 99, C (Nov 2023).

Mikolov T., et.al., Distributed representations of words and phrases and their compositionality, Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2 (NIPS'13), Vol. 2, Curran Associates Inc., Red Hook, NY 2013.

Ouyang L., et.al., Training language models to follow instructions with human feedback, Proceedings of the 36th International Conference on Neural Information Processing Systems (NIPS '22), Curran Associates Inc., Red Hook, NY 2022.

Wu J., et.al., Agentic Reasoning: A Streamlined Framework for Enhancing LLM Reasoning with Agentic Tools, Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vienna, Austria, Association for Computational Linguistics, 2025.