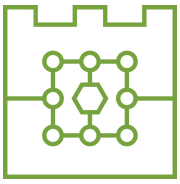
Politechnika Krakowska
im. Tadeusza Kościuszki

Wydział Informatyki i Telekomunikacji



Tomasz Dudzik

Numer albumu: 129007

Zastosowanie dużych modeli językowych do ekstrakcji informacji w domenie finansowej

Application of Large Language Models for Information Extraction in Financial Domain

Praca magisterska
na kierunku Informatyka
specjalność Data Science

Praca wykonana pod kierunkiem:
dr Radosław Kycia

Kraków, 2024

Streszczenie

W niniejszej pracy magisterskiej empirycznie zbadano efektywność dużych modeli językowych w ekstrakcji informacji z dokumentów finansowych. Skoncentrowano się na ocenie zdolności modelu GPT-4 Turbo do przetwarzania i analizy danych finansowych poprzez serię eksperymentów, wykorzystując metody takie jak manualne wstrzykiwanie wiedzy do kontekstu, generacja rozszerzona wyszukiwaniem oraz uczenie na niewielkiej liczbie przykładów. Badania przeprowadzono na danych z raportów finansowych kilku firm notowanych na Giełdzie Papierów Wartościowych. Wyniki wskazują, że uczenie na niewielkiej liczbie przykładów znacząco poprawia efektywność ekstrakcji informacji oraz że generacja rozszerzona wyszukiwaniem jest znacznie mniej efektywna niż manualne wstrzykiwanie wiedzy do kontekstu. Rezultaty te podkreślają praktyczny potencjał dużych modeli językowych oraz wskazują na konieczność dalszego badania ich skutecznych zastosowań.

Abstract

In this master's thesis, the effectiveness of large language models in extracting information from financial documents was empirically examined. The focus was on evaluating the capabilities of the GPT-4 Turbo model in processing and analyzing financial data through a series of experiments, utilizing methods such as manual knowledge injection into the context, retrieval-augmented generation (RAG), and few-shot learning. The research was conducted using data from financial reports of several companies listed on the Stock Exchange. The results indicate that few-shot learning significantly improves the efficiency of information extraction, and that retrieval-augmented generation is much less effective than manual knowledge injection into the context. These findings highlight the practical potential of large language models and point to the need for further research into their effective applications.

Spis treści

1 Wstęp **8**

1.1 Motywacja 8

1.2 Cel pracy 9

1.3 Zakres pracy 9

1.4 Metodyka 10

I Część teoretyczna **11**

2 Przegląd literatury **12**

2.1 Ciągłe pre-trenowanie 12

2.2 Pre-trenowanie specyficzne dla domeny od podstaw 12

2.3 Pre-trenowanie wielodomenowe 13

2.4 Pre-trenowanie wielodomenowe z wykorzystaniem inżynierii podpowiedzi 13

2.5 LLM dostrojony instrukcyjnie z wykorzystaniem inżynierii podpowiedzi . 13

2.6 Metoda zero-shot/few-shot learning 14

3 Podstawy dużych modeli językowych **15**

3.1 Historia i rozwój modeli językowych 15

3.1.1 Modele językowe n-gram 15

3.1.2 Model word2vec 16

3.1.3 Wczesne pre-trenowane neuronowe modele językowe 17

3.1.4 Modele językowe oparte na transformerach 18

3.1.5 Duże modele językowe 19

3.2 Rodzina modeli GPT: Od GPT-1 do GPT-4 20

3.2.1 GPT-3 20

3.2.2 InstructGPT 20

3.2.3 GPT-4 21

4 Duże modele językowe specyficzne dla domeny **22**

4.1 Zero-shot learning 22

4.2 Few-shot learning 22

4.3 Generacja wspomagana narzędziami i dostrajanie . . . 23
4.4 Pre-trenowanie od postaw . . . 23
4.5 Podsumowanie . . . 24

5 Duże modele językowe w analizie sprawozdań finansowych 26
5.1 Obowiązek publikacji sprawozdań finansowych . . . 26
5.2 Formaty sprawozdań finansowych . . . 27
5.2.1 PDF . . . 27
5.2.2 HTML . . . 27
5.2.3 XBRL . . . 27

6 Ekstrakcja informacji 29
6.1 Odpowiadanie na pytania . . . 29

7 Wstrzykiwanie wiedzy do dużych modeli językowych 31
7.1 Przyczyny błędów w odpowiedziach na pytania oparte na faktach . . . 31
7.2 Dostrajanie . . . 33
7.2.1 Dostrajanie nadzorowane . . . 33
7.2.2 Fine-Tuning z wykorzystaniem uczenia ze wzmocnieniem . . . 33
7.2.3 Fine-Tuning nienadzorowany . . . 33
7.3 Generacja rozszerzona wyszukiwaniem . . . 34

II Część badawcza 35

8 Zbiór danych 36
8.1 Baza wiedzy . . . 36
8.2 Pytania i odpowiedzi . . . 37
8.3 Zbiór testowy i treningowy . . . 38

9 Eksperymenty 39
9.1 Wybór modelu . . . 39
9.2 Metryki . . . 40
9.3 Eksperyment 1 - odpowiadanie na pytania bez wstrzykiwania wiedzy . . . 41
9.3.1 Opis . . . 41

9.3.2 Wyniki 41
9.4 Eksperyment 2 - odpowiadanie na pytania z manualnym wstrzykiwaniem wiedzy do kontekstu 42
9.4.1 Opis 42
9.4.2 Wyniki 42
9.5 Eksperyment 3 - odpowiadanie na pytania z manualnym wstrzykiwaniem wiedzy do kontekstu oraz metodą few-shot learning 43
9.5.1 Opis 43
9.5.2 Wyniki 43
9.6 Eksperyment 4 - odpowiadanie na pytania z generacją rozszerzoną wyszukiwaniem 44
9.6.1 Opis 44
9.6.2 Wyniki 44
9.7 Podsumowanie 45

10 Zakończenie **47**
10.1 Wnioski 47
10.2 Potencjalne kierunki rozwoju 47

11 Załączniki **49**
11.1 Kod źródłowy 49

Literatura **55**

1 Wstęp

1.1 Motywacja

W ostatnich latach obserwuje się intensywny rozwój sztucznej inteligencji oraz przetwarzania języka naturalnego, w szczególności dużych modeli językowych (ang. *Large Language Models*, LLMs), takich jak Generative Pre-trained Transformer (GPT). Napędzany jest on przede wszystkim rozwojem technik głębokiego uczenia maszynowego, architektury transformerów zastosowanej w dużej skali oraz ogromnymi ilościami danych i mocy obliczeniowej. Modele takie jak GPT-4 [1] czy Gemini 1.5 [2], zademonstrowały wyjątkowe zdolności w wykonywaniu licznych zadań. Modele te znacznie przewyższają skutecznością swoich poprzedników oraz mają potencjał na zastosowanie w wielu domenach, takich jak programowanie, prawo, medycyna, rolnictwo czy finanse. Co więcej, w wielu zadaniach wydajność tych modeli jest coraz bardziej zbliżona do wydajności osiąganej wcześniej tylko przez ludzi [3].

Sektor finansowy, charakteryzujący się ogromnymi zasobami danych, jest idealnym obszarem do zastosowania dużych modeli językowych. Automatyzacja takich zadań jak przetwarzanie raportów finansowych, analiza ryzyka inwestycyjnego czy monitorowanie rynku w czasie rzeczywistym nie tylko zwiększa efektywność operacyjną, ale także pozwala na skupienie się na innowacjach i rozwoju strategicznym, co jest kluczowe w dynamicznie zmieniającym się środowisku finansowym.

Jednym z zadań przetwarzania języka naturalnego, które znajduje zastosowanie w dziedzinach takich jak prawo czy finanse, jest ekstrakcja informacji (ang. *Information Extraction*, IE). Zadanie to polega na automatycznym wydobywaniu strukturyzowanych informacji z dokumentów niestrukturyzowanych lub półstrukturyzowanych. Odpowiadanie na pytania (ang. *Question Answering*, QA) jest specyficzną formą ekstrakcji informacji, której celem jest tworzenie systemów zdolnych do automatycznego udzielania odpowiedzi na pytania zadane przez użytkownika.

Istnieje rozległa literatura naukowa poświęcona badaniom nad dużymi modelami językowymi (LLM) i ich zdolnością do ekstrakcji informacji [4]. Często jednak badania te koncentrują się na uproszczonych przypadkach użycia, które, choć przydatne w kontekście akademickim, rzadko odpowiadają złożoności rzeczywistych zastosowań praktycznych.

W literaturze brakuje szczegółowych danych na temat efektywności LLM-ów w obszarach, gdzie warunki są mniej kontrolowane, a dane mogą być bardziej zróżnicowane i nieuporządkowane. Wartościowe byłoby więc zbadanie wykorzystania tych technologii do analizy rzeczywistych danych finansowych, których format raportowania może stanowić wyzwanie dla współczesnych dużych modeli językowych.

1.2 Cel pracy

Celem niniejszej pracy jest empiryczne zbadanie, w jaki sposób modele językowe sprawdzają się w realnych zastosowaniach. Przez analizę efektywności dużych modeli językowych w ekstrakcji i przetwarzaniu złożonych danych, niniejsza praca ma na celu nie tylko potwierdzenie ich teoretycznej wartości, ale również ocenę ich praktycznej użyteczności, co może rzucić światło na ich potencjalne ograniczenia i obszary do dalszego rozwoju.

1.3 Zakres pracy

Możliwości modeli zostaną zbadane za pomocą specjalnie przygotowanych eksperymentów skoncentrowanych na ekstrakcji informacji z corocznych raportów publikowanych przez spółki notowane na Giełdzie Papierów Wartościowych (GPW). Ekstrakcja informacji obejmuje kilka etapów, takich jak przygotowanie zbiorów danych, efektywne wstrzykiwanie wiedzy (ang. *Knowledge Injection*), zastosowanie generacji rozszerzonej wyszukiwaniem (ang. *Retrieval-Augmented Generation*, RAG) oraz odpowiednie formułowanie podpowiedzi (ang. *prompts*). Każdy z tych etapów jest kluczowy w celu zapewnienia, że model może skutecznie interpretować i analizować złożone dane finansowe. Przygotowanie danych polega nie tylko na ich selekcji, ale również na transformacji surowych, często niestrukturyzowanych danych do formatu, który może być efektywnie przetwarzany przez modele językowe. Wstrzykiwanie wiedzy i zastosowanie technik takich jak RAG pozwalają na wzbogacenie odpowiedzi modelu o dodatkowe, zewnętrzne informacje, co znacznie poprawia jakość i trafność wyników. Te metody, połączone z precyzyjnym formułowaniem poleceń, umożliwiają modelom głębsze zrozumienie kontekstu finansowego. Dodatkowo, opracowanie odpowiednich poleceń do interakcji z modelem jest równie ważne, gdyż pozwala kierować proces ekstrakcji informacji w sposób bardziej celowany i efektywny. Zrozumienie, jak najlepiej formułować te polecenia, aby uzyskać najbardziej trafne i użyteczne informacje, stanowi istotną część procesu badawczego.

1.4 Metodyka

W celu przeprowadzenia eksperymentów przygotowane zostaną dwa zbiory danych: zbiór testowy, który posłuży do porównywania wyników eksperymentów, oraz zbiór treningowy, wykorzystywany w zadaniach mających na celu poprawę efektywności modelu. Proces ten umożliwi ocenę zdolności modeli do interpretacji danych finansowych oraz testowanie sposobów na poprawę wydajności modelu, wykorzystując historyczne dane. Badania zostaną przeprowadzone za pomocą serii eksperymentów skupionych na różnych aspektach ekstrakcji informacji z corocznych raportów finansowych. Pierwszy eksperyment zbada podstawowe zdolności modelu do odpowiedzi na pytania na podstawie samych danych, na których został wytrenowany. Następnie eksperymenty zostaną rozszerzone o manualne wstrzykiwanie dodatkowej wiedzy, a także zastosowanie metody few-shot learning, co pozwoli ocenić, jak model adaptuje się do nowych, nieznanych wcześniej zapytań i jak techniki te wpływają na poprawę dokładności odpowiedzi. Na końcu zostanie zbadane jak zastosowanie generacji rozszerzonej wyszukiwaniem wpływa na efektywność modelu. Wyniki tych eksperymentów posłużą do głębszej analizy i dyskusji w dalszej części pracy, wskazując zarówno potencjał jak i ograniczenia stosowania modeli językowych w przetwarzaniu skomplikowanych danych finansowych.

Część I

Część teoretyczna

2 Przegląd literatury

Duże modele językowe (LLM) wykazały niezwykłe zdolności w szerokim zakresie zadań przetwarzania języka naturalnego (NLP) i przyciągnęły uwagę z wielu dziedzin, w tym usług finansowych. Użycie tego narzędzia w sektorze finansowym może otworzyć nowe możliwości dla analizy danych, zarządzania ryzykiem, automatyzacji procesów oraz poprawy jakości interakcji z klientem. Sektor finansowy, ze swoją złożonością i dynamicznym charakterem, wymaga narzędzi, które mogą efektywnie przetwarzać i interpretować ogromne ilości niestrukturyzowanych danych tekstowych, takich jak raporty finansowe, wiadomości o rynkach, komentarze analityczne czy komunikaty regulacyjne. Niniejszy przegląd literatury ma na celu zgłębienie obecnych badań i zastosowań LLM w finansach, analizując kluczowe prace, ich metodologie oraz wyniki. W literaturze opisano wiele technik stosowanych do budowania dużych modeli językowych przeznaczonych dla domeny finansowej, poniżej opisane zostanie kilka z nich [5] [6].

2.1 Ciągłe pre-trenowanie

Ciągłe pre-trenowanie (ang. *Continual pre-training*) dużych modeli językowych ma na celu trenowanie istniejącego ogólnego modelu językowego przy użyciu nowych danych specyficznych dla danej dziedziny, w ramach przyrostowej sekwencji zadań.

FinBERT-19 [7] to pierwszy model FinBERT opublikowany do analizy sentymentu finansowego. Implementuje on trzy etapy: 1) inicjalizację ogólnego modelu językowego BERT PLM (3,3 miliarda tokenów), 2) ciągłe pre-trenowanie na korpusie z dziedziny finansowej, oraz 3) dostrojenie do zadań NLP specyficznych dla sektora finansowego.

2.2 Pre-trenowanie specyficzne dla domeny od podstaw

Podejście polegające na pre-trenowaniu specyficznym dla danej dziedziny obejmuje trenowanie modelu wyłącznie na nieoznakowanym korpusie specyficznym dla tej dziedziny, przy jednoczesnym zachowaniu oryginalnej architektury i celów treningowych.

FinBERT-20 [8], jest modelem typu BERT, specyficznym dla domeny finansowej, który został wytrenowany od podstaw na korpusie komunikacji finansowej obejmującym 4,9 miliarda tokenów.

2.3 Pre-trenowanie wielodomenowe

Podejście polegające na pre-trenowaniu wielodomenowym obejmuje trenowanie modelu zarówno na korpusie ogólnym, jak i na korpusie specyficznym dla danej domen. Zakłada się, że tekst z domeny ogólnej pozostaje istotny, podczas gdy dane z domeny finansowej dostarczają wiedzy i umożliwiają adaptację w trakcie procesu pre-trenowania.

FinBERT-21 [9] to kolejny model oparty na BERT przeznaczony do analizy tekstów finansowych, został wytrenowany równolegle na ogólnym korpusie oraz na korpusie specyficznym dla dziedziny finansowej. W modelu tym zastosowano uczenie wielozadaniowe na sześciu zadaniach pre-treningowych typu self-supervised, co umożliwiło mu efektywne przyswajanie wiedzy językowej oraz informacji semantycznych.

2.4 Pre-trenowanie wielodomenowe z wykorzystaniem inżynierii podpowiedzi

Wielodomenowe LLM są trenowane zarówno na obszernym korpusie ogólnym, jak i na dużym korpusie specyficznym dla danej domeny. Następnie użytkownicy opisują zadanie, opcjonalnie dostarczając zestaw przykładów w języku ludzkim. Ta technika, znana jako inżynieria podpowiedzi (ang. *Prompt Engineering*), wykorzystuje ten sam zamrożony model LLM do wielu zadań podrzędnych bez aktualizacji wag.

BloombergGPT [10], wykorzystujący model BLOOM, został wytrenowany na dużym korpusie ogólnym (345 miliardów tokenów) oraz na dużym korpusie finansowym (363 miliardy tokenów). Korpus finansowy, FinPile, zawiera dane zebrane z internetu, wiadomości, dokumentacji, prasy oraz danych własnych Bloomberga.

2.5 LLM dostrojony instrukcyjnie z wykorzystaniem inżynierii podpowiedzi

Dostrajanie instrukcyjne (ang. *Instruction fine-tuning*) to dodatkowe trenowanie LLM przy użyciu wyraźnych instrukcji tekstowych, aby zwiększyć zdolności i możliwość sterowania modelem. Badania w tej dziedzinie można podzielić na dwie główne kategorie [11]: 1) tworzenie zbiorów danych instrukcyjnych i 2) generowanie dostrojonych LLM przy użyciu tych zbiorów danych. W sektorze finansowym badacze zaczęli przekształ-

cać istniejące zbiory danych finansowych w zbiory danych instrukcyjnych, a następnie wykorzystywać te zbiory do dostrajania dużych modeli językowych.

FinMA [12] składa się z dwóch, dostrojonych modeli LLaMA (7B i 30B) [13], które wykorzystują finansowe zbiory danych instrukcyjnych do zadań finansowych. Jest on zbudowany z dużego, wielozadaniowego zbioru danych instrukcyjnych o nazwie Financial Instruction Tuning (FIT, 136k próbek), zawierającego dziewięć publicznie dostępnych zbiorów danych finansowych używanych w pięciu różnych zadaniach. Oprócz pięciu zadań benchmarkowych FLUE zawiera zadanie prognozowania ruchów akcji.

InvestLM [14] to dostrojony model LLaMA65B, wykorzystujący ręcznie przygotowany zbiór danych instrukcyjnych z dziedziny finansów, obejmujący pytania z egzaminów Chartered Financial Analyst (CFA), dokumenty SEC, dyskusje na temat finansów ilościowych ze Stackexchange i zadania NLP związane z finansami.

FinGPT [15] jest otwartoźródłowym i zorientowanym na dane frameworkiem, który oferuje zestaw API do źródeł danych finansowych, zbiór danych instrukcyjnych do zadań finansowych oraz kilka dopasowanych instrukcyjnie finansowych LLM. Zespół FinGPT opublikował kilka podobnych prac opisujących framework oraz pracę eksperymentalną [16] dotyczącą dopasowanych instrukcyjnie FinLLMów przy użyciu sześciu otwartoźródłowych LLMów z metodą adaptacji niskiego rzędu (LoRA).

2.6 Metoda zero-shot/few-shot learning

Poza dostrajaniem istniejących otwartych modeli LLM oraz trenowaniem nowych od podstaw, w aplikacjach finansowych można również korzystać z usługi API od dostawców takich jak OpenAI[1] czy Google[2]. Usługi te nie tylko dostarczają podstawowe zdolności dużych modeli językowych, ale również dodatkowe funkcje dopasowane do poszczególnych zastosowań.

Z modelami udostępnianymi za pomocą API można zastosować metody zero-shot i few-shot learning. W przypadku niektórych zadań techniki te mogą nie przynieść optymalnych wyników. W takich sytuacjach konieczne jest dostrojenie modelu przy użyciu oznakowanych danych, wiedzy specjalistycznej oraz zasobów obliczeniowych, aby osiągnąć zadowalające rezultaty.

[1] `https://openai.com/api` (data dostępu: 5.06.2024)
[2] `https://ai.google.dev/gemini-api` (data dostępu: 5.06.2024)

3 Podstawy dużych modeli językowych

Od momentu premiery ChatGPT[3] w listopadzie 2022 roku, duże modele językowe zyskały ogromne zainteresowanie, stając się przedmiotem intensywnych badań ze względu na ich imponujące osiągnięcia w różnorodnych zadaniach związanych z przetwarzaniem języka naturalnego. Zdolność tych modeli do rozumienia i generowania języka naturalnego wynika z trenowania miliardów parametrów na obszernych zbiorach danych tekstowych. Choć badania nad dużymi modelami językowymi są stosunkowo nowym obszarem nauki, rozwijają się one z niezwykłą dynamiką, co wpływa na postęp w dziedzinie sztucznej inteligencji oraz na praktyczne zastosowania w przemyśle i technologii.

W niniejszym rozdziale przedstawione zostaną wybrane modele językowe, które odegrały znaczącą rolę w historii i rozwoju przetwarzania języka naturalnego. Przedstawiona zostanie ewolucja od prostych modeli opartych na n-gramach, które zainicjowały systematyczne badania nad strukturą języka, poprzez bardziej zaawansowane techniki, takie jak model word2vec, który zrewolucjonizował podejście do reprezentacji semantycznej słów, aż po najbardziej zaawansowane modele oparte na architekturze transformerów. Modele takie jak BERT i GPT zrewolucjonizowały dziedzinę dzięki swojej zdolności do rozumienia kontekstu w tekście na niespotykaną dotąd skalę, ustanawiając nowe standardy w przetwarzaniu języka naturalnego.

3.1 Historia i rozwój modeli językowych

3.1.1 Modele językowe n-gram

Modelowanie języka to temat badawczy, którego początki sięgają lat 50. XX wieku. To właśnie wtedy Claude Shannon zastosował teorię informacji do języka angielskiego, mierząc skuteczność prostych modeli językowych n-gram do przewidywania i kompresji tekstu [17].

Aby zrozumieć pojęcie n-gramów, można rozpatrzyć prostą frazę: „Data science to interdyscyplinarna dziedzina”. N-gramy to sekwencje n kolejnych elementów (często słów) z tekstu, które pomagają modelom językowym przewidywać kontekst i strukturę języka.

[3] https://chatgpt.com (data dostępu: 5.06.2024)

- Unigramy (1-gramy) - są to pojedyncze słowa:
  - „Data”
  - „science”
  - „to”
  - „interdyscyplinarna”
  - „dziedzina”
- - Bigramy (2-gramy) - sekwencje dwóch kolejnych słów, które pokazują, jakie słowo zwykle występuje po innym słowie, dając podstawowy kontekst:
  - „Data science”
  - „science to”
  - „to interdyscyplinarna”
  - „interdyscyplinarna dziedzina”
- Trigramy (3-gramy) sekwencje trzech kolejnych słów, które dostarczają głębszego zrozumienia typowych połączeń w zdaniu:
  - „Data science to”
  - „science to interdyscyplinarna”
  - „to interdyscyplinarna dziedzina”

Każdy kolejny n-gram, od unigramów do trigramów i dalej, oferuje coraz bardziej złożony i kontekstowo bogaty wgląd w strukturę zdania, co jest wykorzystywane w analizie tekstów. Przechodząc od prostych unigramów do bardziej skomplikowanych trigramów, można zauważyć, jak znaczenie rośnie wraz z ilością słów w sekwencji, umożliwiając dokładniejsze przewidywanie i modelowanie języka.

3.1.2 Model word2vec

Model word2vec stanowi znaczący postęp w dziedzinie reprezentacji słów, wprowadzając znaczące ulepszenia w metodach, jakimi algorytmy przetwarzania języka naturalnego analizują kontekst i znaczenie słów w tekście. Został opracowany i opublikowany w 2013 roku przez grupę badawczą Google. Model ten korzysta z architektury sieci neuronowej

do nauki wektorowych reprezentacji słów, określanych mianem osadzeń słów (ang. *word embeddings*), na podstawie analizy dużych zbiorów tekstowych.

Word2vec generuje wektory słów, wykorzystując metodę przewidywania kontekstu danego słowa w zdaniu lub przewidywania słowa na podstawie otaczającego je kontekstu. Używa do tego jednej z dwóch architektur: Continuous Bag of Words (CBOW), która przewiduje obecne słowo na podstawie kontekstu (otaczających słów) lub Skip-Gram, która przewiduje słowa kontekstowe na podstawie obecnego słowa [18].

Word2vec nie tylko umożliwia efektywniejsze przewidywanie słów w tekstach, ale także pozwala na analizę semantyczną i syntaktyczną języka poprzez badanie wzorców w przestrzeni wektorowej. Na przykład, podobieństwo i analogie między słowami mogą być odkrywane przez operacje wektorowe.

3.1.3 Wczesne pre-trenowane neuronowe modele językowe

Pierwsze prace dotyczące modelowania języka za pomocą sieci neuronowych sięgają lat 80. i 90. XX wieku. Pionierskie badania w tej dziedzinie przeprowadzili Rumelhart, Hinton i Williams, którzy w swoim artykule z 1985 roku [19] przedstawili koncepcję propagacji błędu, co stanowiło podstawę do rozwoju algorytmów uczenia dla sieci neuronowych. Następnie, Jeffrey L. Elman w 1990 roku [20] wprowadził model rekurencyjnych sieci neuronowych (RNN), który umożliwił analizę sekwencji czasowych i struktur językowych, co było dużym krokiem naprzód w zrozumieniu strukturalnych aspektów języka.

W kolejnych latach Matthew V. Mahoney w swojej pracy [21] pokazał, jak techniki neuronowe mogą być wykorzystywane do kompresji tekstu, co również miało zastosowanie w modelowaniu języka poprzez efektywne kodowanie i dekodowanie informacji językowych.

Jednak przełom nastąpił na początku XXI wieku, kiedy to Yoshua Bengio i współpracownicy [22] opracowali jeden z pierwszych modeli językowych opartych na sieciach neuronowych, który był w stanie konkurować z modelami n-gramowymi. Ich „Neural Probabilistic Language Model" wykorzystywał gęste wektorowe reprezentacje słów oraz architektury głębokich sieci neuronowych do efektywnego modelowania języka, co otworzyło drogę do bardziej zaawansowanych technologii w dziedzinie przetwarzania języka naturalnego.

Następnie, modelowanie języka za pomocą rekurencyjnych sieci neuronowych

(RNN) i ich odmian, takich jak Long Short-Term Memory (LSTM) [23] i Gated Recurrent Unit (GRU) [24], zyskało popularność w wielu aplikacjach związanych z językiem naturalnym, w tym w tłumaczeniach maszynowych, generowaniu tekstu i klasyfikacji tekstu. Te wczesne eksploracje modelowania języka za pomocą sieci neuronowych stanowiły fundament dla rozwoju nowoczesnych dużych modeli językowych, takich jak te oparte na architekturach transformerów, które są obecnie standardem w przetwarzaniu języka naturalnego.

3.1.4 Modele językowe oparte na transformerach

Wynalezienie architektury transformerów stanowi kolejny kamień milowy w rozwoju neuronowych modeli językowych. Dzięki zastosowaniu mechanizmu samouwagi, który pozwala równolegle obliczać dla każdego słowa w zdaniu lub dokumencie „wynik uwagi", modelując wpływ, jaki każde słowo wywiera na inne, architektura ta umożliwia znacznie większą równoległość niż modele RNN [25]. Dzięki temu możliwe jest efektywne wstępne trenowanie bardzo dużych modeli językowych na dużych ilościach danych na GPU. Te pre-trenowane modele językowe (PLMs) mogą być następnie dostosowywane do wielu zadań. Modele oparte na transformerach można podzielić na trzy główne kategorie: Modele PLM typu „encoder-only", typu „decoder-only" oraz typu „encoder-decoder".

Modele PLM typu „encoder-only" składają się wyłącznie z sieci kodującej (encoder). Modele te zostały pierwotnie opracowane do zadań związanych ze zrozumieniem języka, takich jak klasyfikacja tekstu, gdzie modele muszą przewidywać etykietę klasy dla tekstu wejściowego. Reprezentatywne modele wyłącznie z enkoderem to BERT oraz jego warianty.

BERT (Bidirectional Encoder Representations from Transformers) jest jednym z najczęściej stosowanych modeli językowych typu „encoder-only". Składa się z trzech głównych modułów: (1) modułu osadzeń, który przekształca tekst wejściowy w sekwencję wektorów osadzeń, (2) stosu enkoderów transformer, które przekształcają wektory osadzeń w kontekstowe wektory reprezentacji, oraz (3) warstwy w pełni połączonej, która przekształca wektory reprezentacji (na ostatniej warstwie) w wektory typu one-hot. BERT jest wstępnie trenowany, używając dwóch celów: modelowania języka z maskowaniem (MLM) oraz przewidywania następnego zdania. Wstępnie wytrenowany model BERT może być dostosowywany poprzez dodanie warstwy klasyfikacyjnej do wielu za-

dań związanych ze zrozumieniem języka, począwszy od klasyfikacji tekstu, przez odpowiadanie na pytania, aż po wnioskowanie językowe.

Modele PLM typu „decoder-only” składają się wyłącznie z sieci dekodującej (decoder). Dwa z najczęściej używanych modeli tego typu to GPT-1 i GPT-2, opracowane przez firmę OpenAI. Te modele położyły podwaliny pod bardziej zaawansowane duże modele językowe, takie jak GPT-3 i GPT-4. GPT-1 jako pierwszy wykazał, że dobre wyniki w szerokim zakresie zadań związanych z przetwarzaniem języka naturalnego można osiągnąć dzięki generatywnemu wstępnemu trenowaniu (GPT) modelu transformera typu „decoder-only” [26]. GPT-2 natomiast pokazał, że modele językowe są zdolne do nauki wykonywania specyficznych zadań językowych bez wyraźnego nadzoru [27].

Modele PLM typu „encoder-decoder” składają się zarówno z sieci enkodującej jak i dekodującej. Dowiedziono, że niemal wszystkie zadania związane z przetwarzaniem języka naturalnego (NLP) można przedstawić jako zadania generacji sekwencji do sekwencji [28]. Dlatego model językowy typu encoder-decoder, z założenia, jest modelem uniwersalnym, zdolnym do realizacji wszelkich zadań związanych ze zrozumieniem i generowaniem języka naturalnego. Jednym z reprezentatywnych modeli językowych tego typu jest BART. BART, wykorzystujący standardową architekturę modelu tłumaczenia sekwencja-do-sekwencji, jest wstępnie trenowany poprzez zniekształcanie tekstu za pomocą dowolnej funkcji dodającej szum, a następnie ucząc się odtworzyć oryginalny tekst. Ten proces pre-treningu pozwala modelowi na efektywną naukę odkrywania i poprawiania błędów w tekście, co czyni go szczególnie użytecznym w zadaniach takich jak poprawa gramatyczna, parafrazowanie czy nawet generowanie spójnego tekstu.

3.1.5 Duże modele językowe

Duże modele językowe to głównie modele oparte na architekturze transformerów, które zawierają dziesiątki do setek miliardów parametrów. W porównaniu do wcześniej omawianych modeli PLM, LLMy są znacznie większe pod względem rozmiaru modelu, ale także wykazują silniejsze zdolności do zrozumienia i generowania języka oraz posiadają zdolności emergentne, których nie można zaobserwować w modelach o mniejszej skali.

3.2 Rodzina modeli GPT: Od GPT-1 do GPT-4

Generative Pre-trained Transformers (GPT) są serią modeli językowych opartych wyłącznie na dekoderach, opracowanych przez firmę OpenAI. W rodzinie tej można wyróżnić modele takie jak GPT-1, GPT-2, GPT-3 oraz GPT-4. Wczesne wersje tych modeli są open-source, jednak nowsze, takie jak GPT-3 i GPT-4, są dostępne wyłącznie za pośrednictwem API.

3.2.1 GPT-3

GPT-3 [29] to pre-trenowany autoregresyjny model językowy posiadający 175 miliardów parametrów. Uważany jest on za pierwszy model z kategorii dużych modeli językowych nie tylko dlatego, że jest znacznie większy niż inne PLMy, ale także dlatego, że jako pierwszy zademonstrował emergentne zdolności takie jak uczenie kontekstowe (ang. *in-context learning*). Pozwala ono na realizację zadań bez potrzeby dodatkowego dostosowywania modelu — wystarczy interakcja tekstowa, która określa zadania i demonstruje przykładowe wykonania.

3.2.2 InstructGPT

InstructGPT [30] został zaproponowany, aby umożliwić dużym modelom językowym podążanie za instrukcjami człowieka, poprzez dostosowanie modeli językowych do intencji użytkowników w różnorodnych zadaniach za pomocą dostrajania przy użyciu informacji zwrotnej. Początkowo zbiera się zestaw poleceń napisanych przez osoby zajmujące się etykietowaniem danych oraz przesłanych przez API OpenAI, tworząc zbiór danych z demonstracjami pożądanych zachowań modelu. Następnie GPT-3 jest dostrajany na podstawie tego zbioru. Kolejnym krokiem jest zebranie zbioru danych z ocenami ludzkimi wyników modelu, które służą do dalszego dostrajania modelu za pomocą uczenia ze wzmocnieniem. Metoda ta, znana jako uczenie ze wzmocnieniem na podstawie informacji zwrotnej od ludzi (ang. *Reinforcement Learning from Human Feedback*, RLHF), pozwoliła modelom InstructGPT osiągnąć lepszą prawdomówność i zmniejszyć generowanie toksycznych treści, przy jednoczesnym minimalnym wpływie na wydajność w publicznych zbiorach NLP.

3.2.3 GPT-4

GPT-4 [1], to najnowszy i najbardziej zaawansowany model z serii. Został on uruchomiony w marcu 2023 roku i jako multimodalny duży model językowy, GPT-4 może przetwarzać zarówno tekst, jak i obrazy, generując odpowiedzi tekstowe. Mimo że wciąż ustępuje ludziom w niektórych skomplikowanych realnych scenariuszach, wykazuje wydajność na poziomie ludzkim w różnych profesjonalnych i akademickich benchmarkach, w tym zdając symulowany egzamin prawniczy z wynikami plasującymi go w czołówce uczestników. GPT-4, podobnie jak wcześniejsze modele, był najpierw wstępnie trenowany do przewidywania kolejnych tokenów na dużych korpusach tekstowych, a następnie dostosowywany metodą RLHF, aby lepiej odpowiadać na ludzkie oczekiwania.

4 Duże modele językowe specyficzne dla domeny

Duże modele językowe specyficzne dla domeny stanowią ewolucję w dziedzinie przetwarzania języka naturalnego, dostarczając narzędzi bardziej dostosowanych do specyficznych potrzeb i wymagań poszczególnych sektorów lub dziedzin wiedzy. Rozwój takich modeli jest możliwy dzięki różnorodnym podejściom metodologicznym, które obejmują pre-trenowanie od postaw, dostrajanie oraz wykorzystanie gotowych modeli za pośrednictwem interfejsów API lub rozwiązań open-source.

4.1 Zero-shot learning

Pierwszą decyzją, jaką należy podjąć to wybór pomiędzy wykorzystaniem istniejących usług LLM lub modeli open-source. Jeśli dane wejściowe zawierają informacje poufne, konieczny może być samodzielne hostowanie modelu open-source, takiego jak np. Llama [13].

Jeżeli prywatność danych nie jest problemem, korzystne może być wybranie usług LLM takich jak GPT4 od firmy OpenAI. Ta opcja umożliwia stosunkowo proste eksperymenty oraz wstępną ocenę wydajności bez znaczących kosztów początkowych. Jedynymi kosztami są opłaty związane z każdym wywołaniem API, które zazwyczaj zależą od długości wejścia oraz liczby tokenów generowanych przez model.

4.2 Few-shot learning

Jeżeli zastosowanie metody zero-shot learning nie daje oczekiwanych wyników, można zastosować metodą few-shot learning, jeśli dostępnych jest kilka przykładowych pytań wraz z odpowiedziami. Metoda ta okazuje się skuteczna, co wykazały publikacje takie jak [29] i [31]. Kluczowym pomysłem tej metody jest dostarczenie zestawu przykładowych pytań wraz z odpowiedziami jako kontekst, wraz z faktycznie zdawanym pytaniem. Koszty związane z metodą few-shot learning są podobne do kosztów metody zero-shot learning, z wyjątkiem wymogu dostarczania przykładów za każdym razem. Osiągnięcie dobrej wydajności może wymagać użycia od 1 do kilku przykładów. Te przykłady mogą być takie same dla różnych pytań lub wybrane w zależności od konkretnego pytania. Wyzwanie polega na wyborze odpowiednich przykładów oraz optymalnej ich liczby. Proces

ten obejmuje eksperymentowanie i testowanie aż do osiągnięcia pożądanej granicy wydajności.

4.3 Generacja wspomagana narzędziami i dostrajanie

Jeśli zadanie jest na tyle skomplikowane, że uczenie kontekstowe nie przynosi zadowalających wyników, kolejnym krokiem może być wykorzystanie zewnętrznych narzędzi lub wtyczek współpracujących z dużymi modelami językowymi (LLM). Na przykład, prosty kalkulator może pomóc w zadaniach związanych z arytmetyką, podczas gdy wyszukiwarka może być niezbędna do zadań wymagających dużej ilości wiedzy.

Integracja narzędzi z LLMami może być realizowana poprzez dostarczenie opisów narzędzi. Koszty związane z tym podejściem są zazwyczaj wyższe niż w przypadku metody few-shot learning, ze względu na rozwój narzędzi i dłuższą sekwencję wejściową wymaganą jako kontekst. Koszt wdrożenia zazwyczaj obejmuje koszt używania LLMów oraz koszt używania narzędzi.

Jeśli powyższe opcje nie przynoszą zadowalających wyników, można spróbować dostrajania LLMów. Etap ten wymaga rozsądnej ilości adnotowanych danych, zasobów obliczeniowych (GPU, CPU itp.) oraz ekspertyzy w zakresie strojenia modeli językowych.

4.4 Pre-trenowanie od postaw

Pre-trenowanie dużych modeli językowych to najbardziej zaawansowany sposób tworzenia narzędzi przetwarzania języka naturalnego dostosowanych do szczególnych potrzeb określonej domeny. Proces ten pozwala na budowanie modelu, który od podstaw jest dostosowany do wymagań i charakterystycznych cech danej domeny. Może on zawierać wykorzystanie danych wyłącznie specyficznych dla danej domeny lub ich kombinację z bardziej ogólnymi zestawami danych, co umożliwia osiągnięcie lepszej ogólności i odporności modelu.

Pre-trenowanie od postaw umożliwia dogłębne zrozumienie specyfiki języka i terminologii danej dziedziny, co jest niezbędne dla aplikacji wymagających wysokiej precyzji. Jest to jednak proces wymagający znaczących nakładów czasowych i finansowych. Koszty obliczeniowe mogą sięgać milionów dolarów, a trening wymaga zestawu danych liczącego biliony tokenów. Cały proces może trwać od kilku miesięcy do nawet lat pracy zespołu specjalistów.

Dlatego pre-trenowanie od postaw jest zalecane głównie w projektach, które wymagają niestandardowych rozwiązań lub w sytuacjach, gdy dostępne modele językowe nie spełniają specyficznych wymagań danej dziedziny. To podejście jest szczególnie wartościowe dla projektów o wysokim stopniu specjalizacji. Decyzja o pre-trenowaniu modelu powinna również uwzględniać dostępne zasoby i specyficzne potrzeby użytkownika, co pozwala na lepsze dostosowanie do ograniczeń projektu.

4.5 Podsumowanie

Tabela nr 1 przedstawia przybliżone zakresy wymagań dotyczących danych i kosztów finansowych różnych opcji dużych modeli językowych (LLM). Opublikowane dane pochodzą z publikacji [6]. Koszty wdrożenia dla rozwiązań stron trzecich oraz koszty wdrożeń open source są obliczone na podstawie aktualnych cen usług chmurowych i API dostępnych w chwili pisania niniejszej pracy. Przyjmuje się użycie GPU NVIDIA A100. Koszt generacji tokenów wynika z obliczeń dolarów za sekundę, przyjmując, że generacja 1000 tokenów trwa zwykle od 3 do 33 sekund, w zależności od rozmiaru modelu.

Opcja	**Koszt rozwoju ($)**	**Koszt danych (próbki)**	**Koszt wdrożenia ($/1k tokenów)**
Opne source + zero-shot	-	-	0.006 - 0.037
API + zero-shot	-	-	0.002 - 0.12
Open source + few-shot	-	-	0.006 - 0.037
API + few-shot	-	-	0.002 - 0.12
Open source + narzędzia	-	Koszt tworzenia narzędzi	0.006 - 0.037
API + narzędzi	-	Koszt tworzenia narzędzi	0.002 - 0.12
Open source + dostrajanie	4-360,000	10,000 - 12,000,000	0.0016 - 0.12
API + dostrajanie	30-30,000	10,000 - 12,000,000	0.002 - 0.12
Trenowanie od podstaw	5,000,000	700,000,000	0.0016 - 0.12

Tabela 1: Koszt różnych opcji LLM

Wyniki zawarte w tabeli wskazują na wysokie koszty danych i obliczeń w przypadku dostrajania modeli oraz trenowania ich od zera. Takie podejścia wymagają znaczących inwestycji w zasoby obliczeniowe i zbiory danych, co czyni je mniej dostępnymi w przypadku ograniczonego budżetu. Warto zauważyć, że alternatywne metody, takie jak wy-

korzystanie modeli korzystając z metod zero-shot czy few-shot, mogą oferować znacznie niższe koszty wdrożenia. Dodatkowo generacja wspomagana narzędziami, choć wymaga początkowych inwestycji w rozwój narzędzi, może znacząco zwiększyć możliwości i wydajność modeli LLM w niektórych zastosowaniach.

5 Duże modele językowe w analizie sprawozdań finansowych

Sprawozdania finansowe są kluczowym elementem komunikacji między firmami a ich interesariuszami, w tym inwestorami, regulatorami i rynkami finansowymi. Roczne skonsolidowane sprawozdanie finansowe, zgodnie z ustawą o rachunkowości, zawiera zintegrowane dane finansowe grupy kapitałowej, traktowanej jako jedna jednostka, co pozwala na pełniejsze zrozumienie kondycji finansowej i operacyjnej firmy. Dokumenty te są zazwyczaj dostępne w różnych formatach, takich jak PDF, HTML oraz XBRL (eXtensible Business Reporting Language).

5.1 Obowiązek publikacji sprawozdań finansowych

Zgodnie z ustawą o rachunkowości, spółki, w szczególności te zależne od jednostek dominujących, mają obowiązek sporządzania i publikowania rocznych skonsolidowanych sprawozdań finansowych [32]. Te sprawozdania są wymagane, aby zapewnić transparentność finansową i umożliwić ocenę sytuacji ekonomicznej grupy kapitałowej, traktowanej jako całość.

Skonsolidowane sprawozdanie finansowe zawiera szereg istotnych elementów finansowych, takich jak skonsolidowany bilans, rachunek zysków i strat, rachunek przepływów pieniężnych oraz zestawienie zmian w kapitale własnym. Każdy z tych dokumentów dostarcza szczegółowych informacji na temat kondycji finansowej grupy kapitałowej, w tym zysków, strat, aktywów, pasywów oraz przepływów pieniężnych.

Większość danych w sprawozdaniach finansowych prezentowana jest w formie tabel, co ułatwia ich analizę i porównywanie. Struktura sprawozdań jest często podobna między różnymi jednostkami i latami, co sprzyja spójności i możliwościom śledzenia zmian finansowych w czasie. Nie istnieje jednak jednolity, globalny standard formatowania sprawozdań, co może prowadzić do pewnych różnic w prezentacji danych finansowych, szczególnie w kontekście międzynarodowym.

5.2 Formaty sprawozdań finansowych

Różnorodność formatów stosowanych do publikowania sprawozdań finansowych stanowi wyzwanie dla analizy i ekstrakcji danych. Sprawozdania mogą być dostępne w tradycyjnych formatach takich jak PDF, bardziej strukturalnych jak HTML, czy w specjalnie zaprojektowanych do celów biznesowych formatach takich jak XBRL. Każdy z tych formatów posiada swoje specyficzne cechy, które mogą wpłynąć zarówno na dostępność danych, jak i na ich analizę.

5.2.1 PDF

Format PDF jest powszechnie stosowany do publikacji sprawozdań finansowych ze względu na jego wszechstronność i szeroką akceptację. Nie jest jednak on przystosowany do łatwej ekstrakcji danych, gdyż informacje są zamknięte w formie graficznej, co może utrudniać automatyczne przetwarzanie. PDF zazwyczaj zachowuje wygląd oryginalnego dokumentu, co jest kluczowe dla zachowania formalnego charakteru sprawozdań finansowych, ale to również sprawia, że dostęp do zawartych w nim danych jest bardziej skomplikowany. Narzędzia do ekstrakcji tekstu mogą napotykać problemy z interpretacją tabel i innych złożonych elementów graficznych.

5.2.2 HTML

Format HTML również jest często stosowany do publikacji sprawozdań finansowych. HTML, jako język znaczników, umożliwia strukturalne przedstawienie danych, co jest kluczowe dla automatycznego przetwarzania informacji. Dzięki jasno określonej strukturze, elementy takie jak tabele, nagłówki oraz paragrafy mogą być łatwo identyfikowane i ekstrahowane przez różne narzędzia. Taka struktura jest szczególnie przydatna przy wykorzystaniu dużych modeli językowych, które mogą analizować, interpretować i przetwarzać dane zawarte w sprawozdaniach.

5.2.3 XBRL

Format XBRL (eXtensible Business Reporting Language) jest specjalnie zaprojektowany do komunikacji i wymiany informacji biznesowych, co czyni go przydatnym w analizie sprawozdań finansowych. Jest to standard bazujący na XML, który umożliwia jed-

noznaczną identyfikację i ekstrakcję poszczególnych elementów danych finansowych za pomocą zdefiniowanych etykiet i relacji. Dzięki swojej strukturze XBRL pozwala na automatyczne przetwarzanie i analizę danych bez konieczności manualnego przeszukiwania dokumentów. Każdy element finansowy w dokumencie XBRL jest opatrzony unikalną etykietą, która definiuje jego znaczenie oraz kontekst, co ułatwia interpretację danych przez oprogramowanie i systemy analityczne. Na przykład, elementy takie jak przychody, koszty czy kapitał własny mają swoje specyficzne identyfikatory, co może pozwolić modelom językowym na precyzyjne przekształcanie i analizę tych danych.

6 Ekstrakcja informacji

Ekstrakcja informacji (ang. *Information Extraction*, IE) to istotna dziedzina przetwarzania języka naturalnego, która przekształca zwykły tekst w uporządkowaną wiedzę. Stanowi ona fundamentalny wymóg dla szerokiego zakresu zadań, takich jak konstrukcja grafów wiedzy, rozumowanie oparte na wiedzy oraz odpowiadanie na pytania. Typowe zadania IE obejmują Rozpoznawanie Nazwanych Bytów (ang. *Named Entity Recognition*, NER), Ekstrakcję Relacji (ang. *Relation Extraction*, RE) oraz Ekstrakcję Zdarzeń (ang. *Event Extraction*, EE).

Pojawienie się dużych modeli językowych (LLM), takich jak GPT-4 [1] czy Llama [13], znacząco przyczyniło się do rozwoju przetwarzania języka naturalnego dzięki ich zdolnościom w zakresie rozumienia tekstu, generowania i generalizacji. W związku z tym ostatnio obserwuje się wzrost zainteresowania generatywnymi metodami ekstrakcji informacji, które wykorzystują LLM do generowania informacji strukturalnych, zamiast wydobywania ich ze zwykłego tekstu.

Najnowsze prace pokazały również zdolność dużych modeli językowych (LLM) do generalizacji, umożliwiając im nie tylko uczenie się z danych szkoleniowych dotyczących ekstrakcji informacji poprzez dostrajanie, ale także wydobywanie informacji za pomocą metod few-shot learning oraz zero-shot learning, polegając wyłącznie na przykładach lub instrukcjach podanych w kontekście [33], [34].

6.1 Odpowiadanie na pytania

Odpowiadanie na pytania (ang. *Question Answering*, QA) jest specyficzną formą ekstrakcji informacji, która koncentruje się na tworzeniu systemów zdolnych do generowania odpowiedzi na pytania zadane przez użytkowników. W kontekście dużych modeli językowych QA wykorzystuje zaawansowane zdolności modeli do zrozumienia kontekstu i intencji zawartych w zapytaniu, co umożliwia precyzyjne wyszukiwanie i prezentowanie informacji zawartych w dużych zbiorach danych tekstowych.

Dzięki zdolnościom do generowania naturalnie brzmiących odpowiedzi QA jest szczególnie użyteczne w aplikacjach wymagających interakcji z użytkownikiem, takich jak wirtualni asystenci, systemy wsparcia klienta czy interaktywne bazy wiedzy. Wykorzystanie LLM w zadaniach QA umożliwia również przekształcenie nieustrukturyzowa-

nych danych tekstowych w odpowiedzi, które mogą być bezpośrednio wykorzystane do podejmowania decyzji lub dalszej analizy.

W ramach rozwoju technologii QA istotne są także techniki takie jak rozumienie pytań zależnych od kontekstu, które wymaga od modelu zdolności do utrzymania stanu dialogu lub zrozumienia złożonych zależności między serią pytań i odpowiedzi. Na przykład, w zadaniach takich jak analiza danych finansowych, model może być zapytany o specyficzne trendy czy wyniki, co wymaga od niego nie tylko zrozumienia danych, ale także zdolności do syntezowania i prezentowania wyników w sposób zrozumiały dla użytkownika.

Modele takie jak GPT-4 wykazują obiecujące wyniki w zadaniach QA, częściowo dzięki ich zdolnościom do uczenia kontekstowego, które pozwala modelowi dostosowywać odpowiedzi na podstawie dostarczonych przykładów lub instrukcji, bez konieczności dalszego dostrajania. To podejście, znane również jako uczenie zero-shot i few-shot, jest szczególnie wartościowe w sytuacjach, gdzie nie jest możliwe lub praktyczne gromadzenie dużych ilości oznakowanych danych do treningu [35] [36].

7 Wstrzykiwanie wiedzy do dużych modeli językowych

Duże modele językowe charakteryzują się zdolnością do przechowywania obszernych ilości informacji opartych na faktach [37]. Wykazują one również imponujący poziom wiedzy w różnorodnych dziedzinach, który wynika z pre-trenowania na rozległych zbiorach danych. Istnieją jednak dwa główne ograniczenia tej wiedzy: jest ona statyczna i nie aktualizuje się w czasie oraz może brakować jej szczegółowości ekspertyzy w konkretnych dziedzinach. Oba te problemy są ze sobą głęboko powiązane, ponieważ ich rozwiązanie opiera się na ulepszeniu bazy wiedzy modelu.

Ostatnio coraz częściej pojawiają się prace nad dostosowywaniem dużych modeli językowych do specyficznych dziedzin i aktualizacją ich wiedzy [38]. Zaproponowano różne modele mające na celu poprawę wiedzy opartej na faktach i zdolności w różnych obszarach, takich jak opieka zdrowotna [39], finanse [10] czy prawo [40].

Jednym ze sposobów na dodanie wiedzy do wstępnie wytrenowanego modelu jest fine-tuning. Proces ten polega na kontynuacji treningu modelu przy użyciu danych specyficznych dla zadania, co ma na celu adaptację wag modelu do konkretnej bazy wiedzy, optymalizując jego wydajność i trafność w specjalistycznych dziedzinach.

Inną metodą wzmocnienia bazy wiedzy modelu jest wykorzystanie uczenia kontekstowego (ang. *in-context learning*, ICL). Podstawową ideą uczenia kontekstowego jest poprawa wydajności wstępnie wytrenowanych LLM w nowych zadaniach przez modyfikację zapytania wejściowego do modelu bez bezpośredniej zmiany jego wag. Jedną z form ICL jest generowanie rozszerzeone wyszukiwaniem (ang. *Retrieval Augmented Generation*, RAG) [41], które wykorzystuje techniki wyszukiwania informacji, aby umożliwić dużym modelom językowym uzyskanie odpowiednich informacji ze źródła wiedzy i włączenie ich do generowanego tekstu.

7.1 Przyczyny błędów w odpowiedziach na pytania oparte na faktach

Istnieje wiele potencjalnych przyczyn niepoprawnego odpowiadania dużych modeli językowych na pytania oparte na faktach. W publikacji [42] przedstawiono klasyfikację pięciu głównych przyczyn błędów w faktach na poziomie modelu [43]:

- **Brak wiedzy domenowej:** Model językowy może nie posiadać kompleksowej wiedzy specjalistycznej w danej dziedzinie, do której nie był wcześniej wystawiony. Na

przykład, model szkolony wyłącznie na tekstach Williama Shakespeare'a prawdopodobnie nie poradzi sobie dobrze, gdy zostanie zapytany o dzieła Marka Twaina.

- **Przestarzałe informacje:** Duże modele językowe (LLM) mają z góry określoną datę ostatniego treningu w swoim zestawie danych. W związku z tym wszystkie wydarzenia, odkrycia lub zmiany, które miały miejsce po ostatniej aktualizacji treningowej, nie będą znajdować się w zakresie wiedzy modelu bez dostępu do zewnętrznych źródeł.

- **Niezapamiętywanie:** Czasami model jest wystawiony na wiedzę podczas procesu treningu, ale jej nie zatrzymuje. Jest to szczególnie prawdziwe dla rzadkich faktów, które pojawiają się w zestawie danych treningowych tylko sporadycznie.

- **Zapominanie:** Modele językowe często przechodzą dodatkowy trening po fazie wstępnego trenowania (fine-tuning). W niektórych przypadkach może to prowadzić do zjawiska zwanego *katastroficznym zapominaniem*, gdzie modele tracą część wiedzy, którą posiadały przed procesem dostrajania.

- **Błąd w rozumowaniu:** W niektórych przypadkach model językowy może posiadać odpowiednią wiedzę na temat faktu, ale nie potrafi jej właściwie wykorzystać. Jest to szczególnie widoczne w złożonych zadaniach wymagających wieloetapowego rozumowania lub gdy model jest pytany różnymi pytaniami o ten sam fakt, co prowadzi do rozbieżnych wyników.

Proces pre-trenowania na ogólnych danych jest najczęściej niewystarczający do wykonywania przez duży model językowy zadań wymagających specjalistycznej wiedzy. Aby przeciwdziałać tym ograniczeniom, istotne jest zastosowanie dodatkowego kroku post-przetwarzania, mającego na celu wzbogacenie wiedzy wytrenowanego już modelu. Ten proces, często określany jako wstrzykiwanie wiedzy, pozwala na znaczące zwiększenie funkcjonalności modeli przez adaptację do bardziej złożonych i specyficznych zadań.

Dwie szeroko używane metody wstrzykiwania wiedzy to dostrajanie oraz generacja rozszerzona wyszukiwaniem (RAG) [43].

7.2 Dostrajanie

7.2.1 Dostrajanie nadzorowane

Dostrajanie nadzorowane (ang. *Supervised Fine-Tuning*, SFT) polega na regulacji pre-trenowanego modelu przy użyciu zestawów etykietowanych par wejście-wyjście. Jedną z najbardziej rozpowszechnionych metod SFT jest tzw. strojenie instrukcyjne (ang. *instruction tuning*), które okazało się skuteczne w poprawie wydajności modelu. W strojeniu instrukcyjnym wejściem jest opis zadania w języku naturalnym, a wyjściem – przykład pożądanego zachowania. Wiele współczesnych modeli LLM przeszło przez proces strojenia instrukcyjnego, co znacząco podniosło ich jakość, szczególnie w zakresie zdolności do rozumowania i zastosowaniach zero-shot. Jednak mimo tych zalet, strojenie instrukcyjne samo w sobie nie wprowadza nowej wiedzy do modelu, co oznacza, że nie rozwiązuje problemu wstrzykiwania wiedzy.

7.2.2 Fine-Tuning z wykorzystaniem uczenia ze wzmocnieniem

Inną formą dostosowywania modelu jest wykorzystanie strategii optymalizacji inspirowanych uczeniem ze wzmocnieniem (RL). Przykłady to uczenie ze wzmocnieniem od ludzkiego feedbacku (ang. *Reinforcement Learning from Human Feedback*, RLHF), optymalizacja bezpośrednich preferencji (ang. *Direct Preference Optimization*, DPO) oraz optymalizacja polityki zbliżonej (ang. *Proximal Policy Optimization* (PPO)). Te techniki okazały się użyteczne, zwłaszcza gdy są stosowane razem ze strojeniem instrukcyjnym. Podobnie jednak jak w przypadku strojenia instrukcyjnego, skupiają się one na ogólnej jakości odpowiedzi i oczekiwanym zachowaniu, a niekoniecznie na poszerzaniu wiedzy modelu.

7.2.3 Fine-Tuning nienadzorowany

Ostatnią omawianą strategią jest dostrajanie nienadzorowane, oznaczające brak dostępnych etykiet, z których model mógłby się uczyć. Powszechną techniką w tej kategorii jest ciągłe pre-trenowanie (ang. *Continual Pre-training*). W tej metodzie proces dostrajania traktowany jest jako bezpośrednie kontynuowanie fazy pre-trenowania. Zaczynamy od zapisanego checkpointu oryginalnego modelu i trenujemy go w sposób autoregresywny, czyli przewidując następny token. Jedną z głównych różnic w porównaniu do właściwego pre-trenowania jest tempo uczenia, które zazwyczaj powinno być znacznie niższe, aby

uniknąć katastrofalnego zapominania. Wykorzystanie nienadzorowanego dostrajania pozwala kontynuować proces wstrzykiwania wiedzy do modelu.

7.3 Generacja rozszerzona wyszukiwaniem

Generacja rozszerzona wyszukiwaniem (ang. *Retrieval Augmented Generation*, RAG) to technika, która rozszerza możliwości dużych modeli językowych, szczególnie w zadaniach wymagających intensywnej wiedzy, poprzez wykorzystanie zewnętrznych źródeł wiedzy. Chociaż pierwotnie każde zadanie wymagało dodatkowego treningu, wykazano, że wstępnie wytrenowany model osadzeń może osiągnąć lepszą wydajność bez dodatkowego treningu. Idea polega na tym, że mając do dyspozycji pomocniczą bazę wiedzy oraz zapytanie wejściowe, wykorzystujemy architekturę RAG do znalezienia w bazie wiedzy dokumentów, które odpowiadają zapytaniu wejściowemu. Te dokumenty są następnie dodawane do zapytania wejściowego, dając modelowi dodatkowy kontekst na temat tematu zapytania.

Część II

Część badawcza

8 Zbiór danych

W ramach niniejszej pracy przygotowany został zbiór danych, który dzieli się na dwie główne części: *Baza wiedzy* oraz *Pytania i odpowiedzi*. Celem *Bazy wiedzy* jest dostarczenie informacji niezbędnych do skutecznego odpowiadania na pytania, natomiast *Pytania i odpowiedzi* stanowią zestaw danych składający się z pytań używanych w eksperymentach oraz z odpowiadających im oczekiwanych odpowiedzi, które są niezbędne do oceny skuteczności modelu.

8.1 Baza wiedzy

Baza wiedzy, której celem jest dostarczenie informacji do odpowiadania na pytania, składa się z raportów finansowych trzech różnych spółek giełdowych, reprezentujących różnorodne sektory rynkowe: sektor budowlany, telewizyjny oraz energetyczny. Raporty te pochodzą z lat 2021 i 2023. Spółki giełdowe często publikują swoje raporty w kilku formatach, jednak wszystkie raporty znajdujące się w tym zbiorze danych są w formacie HTML. Format HTML jest szczególnie przydatny w kontekście raportów finansowych, ponieważ umożliwia strukturyzowanie i wyodrębnianie danych z tekstów i tabel, co może okazać się przydatne dla ekstrakcji informacji.

Poniżej przedstawiono przykładowe dane z *bazy wiedzy*, zawierające tabele z raportów dwóch różnych spółek giełdowych. Prezentują one wybrane pozycje finansowe, które można znaleźć w raporcie[4] [5].

Grupa Budimex

Skonsolidowane sprawozdanie finansowe za rok zakończony 31 grudnia 2023 roku sporządzone zgodnie z Międzynarodowymi Standardami Sprawozdawczości Finansowej

budimex

(wszystkie kwoty wyrażone są w tysiącach złotych)

Skonsolidowane sprawozdanie z sytuacji finansowej

AKTYWA	Nota	31 grudnia 2023 roku	31 grudnia 2022 roku
Aktywa trwałe (długoterminowe)			
Rzeczowe aktywa trwałe	10	717 986	640 734
Wartości niematerialne	11	131 112	145 094
Wartość firmy jednostek podporządkowanych	12	178 198	178 198
Inwestycje w jednostkach wycenianych metodą praw własności	13	2 657	2 405
Inwestycje w instrumenty kapitałowe	14.3	3 892	7 545
Kaucje z tytułu umów o budowę	28	67 631	83 393
Należności z tytułu dostaw i usług oraz pozostałe należności	16	26 718	24 441
Należności z tytułu umów koncesyjnych	15	46 266	46 511
Pozostałe aktywa finansowe	14	16 890	4 777
Aktywa z tytułu odroczonego podatku dochodowego	23	810 426	685 036
Aktywa trwałe (długoterminowe) ogółem		**2 001 776**	**1 818 134**
Aktywa obrotowe (krótkoterminowe)			

Skonsolidowane Sprawozdanie Finansowe GK PGE
za rok 2023 zgodne z MSSF UE (w mln PLN)

PGE Polska Grupa Energetyczna

SKONSOLIDOWANE SPRAWOZDANIE Z SYTUACJI FINANSOWEJ

	Nota	Stan na dzień 31 grudnia 2023	Stan na dzień 31 grudnia 2022
Rzeczowe aktywa trwałe	9	68.508	64.388
Wartości niematerialne	10	1.952	726
Prawa do użytkowania składników aktywów	11	1.852	1.311
Należności finansowe	24.1.1	254	223
Instrumenty pochodne i inne aktywa wyceniane w wartości godziwej przez wynik finansowy	24.1.2	278	608
Udziały i akcje oraz pozostałe instrumenty kapitałowe		102	117
Udziały i akcje wykazywane metodą praw własności	12	453	180
Pozostałe aktywa długoterminowe	16.1	1.147	882
Uprawnienia do emisji CO_2 na własne potrzeby	15	20	114
Aktywa z tytułu podatku odroczonego	13.1	3.774	3.183
AKTYWA TRWAŁE		**78.340**	**71.732**
Zapasy	14	3.773	4.918
Uprawnienia do emisji CO_2 na własne potrzeby	15	10.517	4.754
Należności z tytułu podatku dochodowego		967	239
Instrumenty pochodne i inne aktywa wyceniane w wartości godziwej przez wynik finansowy	24.1.2	116	927
Należności z tytułu dostaw i usług i pozostałe należności finansowe	24.1.1	10.516	9.083
Pozostałe aktywa krótkoterminowe	16.2	3.181	2.238
Środki pieniężne i ich ekwiwalenty	17	6.033	11.887
AKTYWA OBROTOWE		**35.103**	**34.046**
SUMA AKTYWÓW		**113.443**	**105.778**

Rysunek 1: Przykłady raportów giełdowych tworzących bazę wiedzy

[4] `https://inwestor.budimex.pl/raporty-okresowe` (data dostępu: 20.04.2024)
[5] `https://www.gkpge.pl/dla-inwestorow/akcje/dane-finansowe` (data dostępu: 20.04.2024)

Jak można zaobserwować, choć tabele w różnych raportach mają podobną strukturę, często różnią się układem danych, nazewnictwem pozycji finansowych oraz formatowaniem i jednostkami prezentowanych kwot, co wskazuje na ich różnorodność mimo reprezentowania analogicznych danych.

8.2 Pytania i odpowiedzi

Zbiór danych *Pytania i odpowiedzi* składa się z pytań oraz odpowiadających im oczekiwanych odpowiedzi. Jest on niezbędny do przeprowadzania eksperymentów oraz oceny skuteczności modelu. Zawiera pozycje finansowe z trzech kategorii: Rachunek zysków i strat, Bilans oraz Przepływy pieniężne. Te pozycje zostały zidentyfikowane i wyodrębnione przez ekspertów z raportów finansowych, a następnie opublikowane na portalu BiznesRadar[6]. Opisują one elementy tabel w raportach finansowych, reprezentując bezpośrednie wartości z raportów lub będąc wynikiem sumy, bądź różnicy kilku innych pozycji.

W tabeli 2 przedstawiono przykład pozycji finansowych, aby zilustrować strukturę i typ informacji, które są analizowane w ramach badań. Zbiór danych składa się łącznie ze 145 pozycji finansowych trzech spółek giełdowych. Kolumna „Symbol" zawiera symbol identyfikujący spółkę, kolumna „Pozycja" zawiera nazwy pozycji finansowych, a kolumny „2021" oraz „2023" wartości, jakie można przypisać pozycjom kolejno w latach 2021 oraz 2023.

Symbol	**Pozycja**	**2021**	**2023**
BDX	Przychody ze sprzedaży	7 911 192	9 801 515
BDX	Techniczny koszt wytworzenia produkcji sprzedanej	7 077 395	8 676 934
BDX	Koszty sprzedaży	11 733	13 516
CYFRPLSAT	Aktywa trwałe	24 159 700	28 149 200
CYFRPLSAT	Aktywa trwałe - Wartości niematerialne i prawne	16 251 400	18 400 700
CYFRPLSAT	Aktywa obrotowe	8 077 300	9 027 500
PGE	Przepływy pieniężne z działalności operacyjnej	7 456 000	3 269 000
PGE	Amortyzacja	4 412 000	13 455 000
PGE	Przepływy pieniężne z działalności inwestycyjnej	-4 367 000	-11 451 000

Tabela 2: Przykład pozycji finansowych tworzących zbiór pytań i odpowiedzi

[6] `https://biznesradar.pl` (data dostępu: 15.05.2024)

W zbiorze *pytań i odpowiedzi* uwzględniono również powiązanie poszczególnych pozycji finansowych z ich dokładnym miejscem występowania w raportach. Ta informacja jest kluczowa podczas przeprowadzania eksperymentów wymagających identyfikacji źródeł, z których pochodzą odpowiedzi na zadawane pytania.

8.3 Zbiór testowy i treningowy

Obydwa wyżej opisane zbiory można podzielić również na *zbiór treningowy* oraz *zbiór testowy*. *Zbiór treningowy* może być wykorzystywany do nauki i kalibracji modelu, dostarczając danych, które pozwalają modelowi na adaptację i optymalizację w celu efektywnej analizy danych. Zbiór ten stanowią pochodzące z raportów finansowych dane z roku 2021, a także odpowiadające im pytania i odpowiedzi, umożliwiając modelowi naukę na realnych, historycznych przykładach.

Z kolei *zbiór testowy* został zaprojektowany z myślą o umożliwieniu weryfikacji efektywności modeli oraz porównania wyników różnych eksperymentów. Zawiera on raporty z roku 2023, które w czasie pisania tej pracy nie były częścią danych, na których trenowany był rozważany model. Dzięki temu zbiór testowy umożliwia ocenę, jak dobrze model radzi sobie z nieznanymi wcześniej informacjami i w jakim stopniu jest w stanie generalizować nauczone wzorce na nowe dane.

Raporty finansowe, mimo że publikowane są co roku, często zachowują spójną strukturę, co ułatwia proces treningu i testowania modeli. Wykorzystanie tej spójności strukturalnej pozwala modelom lepiej rozumieć i przetwarzać dane, niezależnie od okresu, z którego pochodzą.

9 Eksperymenty

Celem przeprowadzanych eksperymentów jest ocena zdolności modelu do skutecznego odpowiadania na pytania z użyciem różnych technik wstrzykiwania wiedzy oraz możliwości generalizacji na nowe dane. Eksperymenty te mają kluczowe znaczenie dla weryfikacji, jak model radzi sobie z zadaniami odpowiadania na pytania oparte na danych finansowych, które nie były bezpośrednio dostępne podczas jego treningu. Porównywane są różne metody dostarczania informacji do modelu, aby zrozumieć, które z nich najefektywniej wspomagają model w precyzyjnym odpowiadaniu. Przykłady kodu wykorzystanego w eksperymentach umieszczone zostały w załączniku.

9.1 Wybór modelu

Wybór odpowiedniego modelu językowego stanowi pierwszą i zarazem najistotniejszą decyzję w procesie przygotowania do eksperymentów. Odpowiedni dobór modelu jest kluczowy, gdyż wpływa na efektywność oraz zdolność modelu do generalizacji wiedzy domenowej. Aby podjąć tę decyzję, konieczne jest rozważenie dostępnych metod uczenia modelu, które obejmują opisane wcześniej uczenie kontekstowe, dostrajanie modelu oraz pre-trenowanie od podstaw.

Biorąc pod uwagę ograniczenia związane z dostępnością danych oraz mocą obliczeniową, zdecydowano o rezygnacji z metod pre-trenowania i dostrajania modelu. W konsekwencji wybór padł na uczenie kontekstowe. Decyzję tę dodatkowo umocnił fakt, że najbardziej zaawansowane modele językowe dostępne przez API nie oferują możliwości ich dostrajania. Takie podejście pozwala na efektywne wykorzystanie modeli w kontekście dostępnych zasobów oraz skupienie się na istniejących już pre-trenowanych modeli do nauki i adaptacji w kontekście specyficznych dla danej domeny danych.

Analizując dostępne w literaturze benchmarki, szczególnie te dotyczące zadań typu Question Answering, zauważalna jest najwyższa efektywność modelu GPT-4 opracowanego przez OpenAI. Wyniki te wskazują, że GPT-4 przewyższa inne dostępne modele pod względem zdolności do rozumienia i generowania odpowiedzi na pytania [44] [45]. W związku z tym zdecydowano o wyborze modelu GPT-4 Turbo (gpt−4−turbo−2024−04−09) jako głównego narzędzia w przeprowadzanych eksperymentach. Model ten oferuje zaawansowane przetwarzanie języka naturalnego z maksymalną długością kontekstu wyno-

szącą 128 000 tokenów, co jest istotne w kontekście obsługi długich i złożonych zapytań charakterystycznych dla danych finansowych. Dodatkowo model ten został wyposażony w możliwości przetwarzania wizualnego oraz obsługę JSON i wywoływania funkcji, co znacząco rozszerza zakres jego zastosowania, umożliwiając integrację z bardziej złożonymi systemami i formatami danych. Model ten został wytrenowany na danych z okresu do grudnia 2023 roku, co jest istotnym faktem w kontekście tworzenia zbioru danych testowych, na których model nie był trenowany.

9.2 Metryki

Ocena skuteczności modelu w eksperymentach zostanie oceniona poprzez mierzenie procentu poprawnych odpowiedzi, co jest główną metryką używaną do analizy wyników. Dopuszczalne były drobne błędy, takie jak odpowiedzi podane w niewłaściwych jednostkach (np. milionach zamiast tysiącach), jednak oceniana była również zdolność modelu do precyzyjnego odpowiadania bez błędów:

- **Procent poprawnych odpowiedzi:** Podstawowa miara efektywności, która ocenia, ile procent zadanych pytań zostało odpowiedzianych poprawnie przez model.
- **Procent odpowiedzi ”Brak danych”:** Ocena, jaki procent razy model stwierdził, że nie posiada danych niezbędnych do odpowiedzi, co jest ważne w kontekście zapobiegania generowaniu błędnych odpowiedzi (halucynacji).
- **Procent niepoprawnych odpowiedzi:** ocena, jaki procent razy model odpowiedział nieprawidłowo, co jest sytuacją najbardziej niepożądaną.

Dodatkowo rozpatrywano skuteczność odpowiedzi według poszczególnych firm, co pozwalało zrozumieć, czy model równie dobrze radzi sobie z danymi pochodzącymi z różnych raportów. Ważnym elementem analizy była również identyfikacja przyczyn błędnych odpowiedzi, co obejmowało rozróżnienie czy błędy wynikały z niewystarczającej zdolności modelu do wykonania obliczeń (np. suma/różnica kilku pozycji) czy z niezrozumienia pytania spowodowanego różnicami w nazewnictwie pozycji finansowych.

9.3 Eksperyment 1 - odpowiadanie na pytania bez wstrzykiwania wiedzy

9.3.1 Opis

Celem eksperymentu nr 1 jest zbadanie skuteczności modelu w odpowiadaniu na pytania bez wstrzykiwania dodatkowej wiedzy. Model, który podlega badaniu, został wytrenowany na danych dostępnych do grudnia 2023 roku, a raporty finansowe, których dotyczą pytania, zostały opublikowane w 2024 roku, zatem model nie powinien być w stanie odpowiedzieć na żadne pytanie dotyczące tych raportów.

9.3.2 Wyniki

Tabela nr 3 przedstawia tabelę wyniki eksperymentu nr 1, w którym model nie miał dostępu do dodatkowej wiedzy:

Firma	Poprawne	Niepoprawne	Brak danych	Błędy w jednostkach
BDX	0,00%	0,00%	100,00%	N/A
CYFRPLSAT	0,00%	0,00%	100,00%	N/A
PGE	0,00%	0,00%	100,00%	N/A
Razem	0,00%	0,00%	100,00%	N/A

Tabela 3: Wyniki eksperymentu nr 1

Analiza wyników pokazuje, że model konsekwentnie odpowiadał „Brak danych” na wszystkie pytania, co było spodziewanym rezultatem, biorąc pod uwagę, że raporty finansowe z 2024 roku, o które pytano, nie były dostępne w trakcie treningu modelu. Model nie generował błędnych odpowiedzi, co potwierdza jego zdolność do poprawnego identyfikowania braków w dostępnej wiedzy, zamiast podania nieprawidłowej informacji (halucynacji).

9.4 Eksperyment 2 - odpowiadanie na pytania z manualnym wstrzykiwaniem wiedzy do kontekstu

9.4.1 Opis

Celem eksperymentu nr 2 jest zbadanie skuteczności modelu w odpowiadaniu na pytania przy manualnym wstrzykiwaniu dodatkowej wiedzy do kontekstu. W ramach tego eksperymentu, przed zadaniem pytania, do kontekstu zostały dodane niezbędne informacje, które model powinien wykorzystać do udzielenia odpowiedzi. Te dodatkowe informacje obejmują szczegóły z raportów finansowych opublikowanych w 2024 roku. Eksperyment ma na celu sprawdzenie, jak dobrze model potrafi wykorzystać te manualnie dodane dane do udzielenia precyzyjnych i trafnych odpowiedzi na zadane pytania.

9.4.2 Wyniki

Tabela nr 4 zawiera wyniki eksperymentu nr 2. Ilustruje ona efektywność modelu w odpowiedziach na pytania z manualnym wstrzykiwaniem wiedzy:

Firma	**Poprawne**	**Niepoprawne**	**Brak danych**	**Błędy w jednostkach**
BDX	68,75%	14,58%	16,67%	0,00%
CYFRPLSAT	49,02%	21,57%	29,41%	24,00%
PGE	65,22%	15,22%	19,57%	33,33%
Razem	60,69%	17,24%	22,07%	18,18%

Tabela 4: Wyniki eksperymentu nr 2

Wyniki wskazują, że model różnie radzi sobie z pytaniami zależnie od spółki, co sugeruje potrzebę dalszego dostosowywania i kalibracji modelu do specyfiki danych. Błędy w jednostkach miały miejsce głównie w przypadku spółki PGE, co wskazuje na potencjalne obszary do poprawy w zakresie rozumienia kontekstu finansowego.

Analiza przyczyn błędów wykazała, że 80% odpowiedzi uznanych za nieprawidłowe oraz 37,5% odpowiedzi ”Brak danych”, było spowodowanych trudnościami modelu z przeprowadzeniem odpowiednich obliczeń.

9.5 Eksperyment 3 - odpowiadanie na pytania z manualnym wstrzykiwaniem wiedzy do kontekstu oraz metodą few-shot learning

9.5.1 Opis

Celem eksperymentu nr 3 jest zbadanie skuteczności modelu w odpowiadaniu na pytania przy manualnym wstrzykiwaniu dodatkowej wiedzy do kontekstu oraz uczeniu modelu na podstawie niewielkiej liczby przykładów (ang. *few-shot learning*). W tym eksperymencie, oprócz dodawania niezbędnych informacji do kontekstu, model będzie również otrzymywał kilka przykładów pytań i odpowiedzi, które mają pomóc mu lepiej zrozumieć strukturę i oczekiwania dotyczące odpowiedzi. Przykłady będą pochodzić z raportów finansowych znajdujących się w zbiorze danych treningowych. Literatura sugeruje, że nawet podanie jednego przykładu może pomóc dużym modelom językowym w analizie danych znajdujących się w tabelach [46]. Eksperyment ma na celu ocenę, jak dobrze model potrafi wykorzystać podane przykłady do udzielenia precyzyjnych i trafnych odpowiedzi.

9.5.2 Wyniki

Tabela nr 5 zawiera wyniki eksperymentu nr 3, w którym model był wspomagany manualnie dodaną wiedzą oraz dodatkowymi przykładami:

Firma	Poprawne	Niepoprawne	Brak danych	Błędy w jednostkach
BDX	77,08%	6,25%	16,67%	0,00%
CYFRPLSAT	64,71%	25,49%	9,80%	6,06%
PGE	71,74%	15,22%	13,04%	27,27%
Razem	71,03%	15,86%	13,10%	10,68%

Tabela 5: Wyniki eksperymentu nr 3

Analiza wyników eksperymentu nr 3 pokazuje, że manualne wstrzykiwanie wiedzy oraz nauka oparta na niewielkiej liczbie przykładów znacząco poprawiły zdolność modelu do udzielania poprawnych odpowiedzi na pytania. Najwyższy procent poprawnych odpowiedzi odnotowano dla firmy BDX, co może świadczyć o lepszej dostępności lub jakości danych, które zostały użyte do treningu modelu w kontekście tej spółki.

Niepoprawne odpowiedzi oraz błędy w jednostkach wskazują na obszary, które

wymagają dalszej kalibracji i optymalizacji modelu. Szczególnie błędy jednostkowe dla PGE (27,27%) sugerują potrzebę dodatkowego skupienia się na nauce rozpoznawania i przeliczania wartości finansowych w odpowiednich jednostkach. Znaczna poprawa w przypadku modelu, który otrzymał wsparcie w formie przykładów, potwierdza wartość tej metody w kontekście zwiększania precyzji odpowiedzi modelu.

Analiza przyczyn błędów wykazała, że tym razem 69,57% błędnych odpowiedzi oraz 42,11% odpowiedzi ”Brak danych” było spowodowane trudnościami modelu z przeprowadzaniem obliczeń.

9.6 Eksperyment 4 - odpowiadanie na pytania z generacją rozszerzoną wyszukiwaniem

9.6.1 Opis

Celem eksperymentu nr 4 jest zbadanie skuteczności modelu w odpowiadaniu na pytania przy użyciu generacji rozszerzonej wyszukiwaniem (ang. *Retrieval Augmented Generation*, RAG) oraz uczeniu modelu na podstawie niewielkiej liczby przykładów (ang. *few-shot learning*). Ten eksperyment różni się od eksperymentu nr 2 tym, że informacje wstrzykiwane do kontekstu nie są dodawane manualnie, lecz są automatycznie pozyskiwane za pomocą metody RAG. W tej metodzie wykorzystywane są wektorowe osadzenia (ang. *vector embeddings*), liczone na podstawie stron sprawozdania finansowego, przy użyciu modelu OpenAI $\text{text}-\text{embedding}-3-\text{large}$, który obsługuje maksymalnie 8191 tokenów na wejściu. Takie podejście pozwala na automatyczne wyszukiwanie semantyczne i integrowanie odpowiednich informacji z zewnętrznych zbiorów danych bezpośrednio do kontekstu modelu. Eksperyment ten ma na celu ocenić, jak skutecznie model radzi sobie w pełni zautomatyzowany sposób, korzystając z danych uzyskanych poprzez zastosowanie techniki RAG.

9.6.2 Wyniki

Tabela nr 6 przedstawia wyniki eksperymentu nr 4, w którym model używał techniki generacji rozszerzonej wyszukiwaniem oraz uczenia na podstawie niewielkiej liczby przykładów:

Firma	Poprawne	Niepoprawne	Brak danych	Błędy w jednostkach
BDX	37,50%	12,50%	50,00%	0,00%
CYFRPLSAT	25,49%	11,76%	62,75%	3,92%
PGE	45,65%	15,22%	39,13%	17,39%
Razem	35,86%	13,10%	51,03%	6,90%

Tabela 6: Wyniki eksperymentu nr 4

Analiza wyników eksperymentu nr 4 pokazuje, że użycie techniki RAG przyniosło raczej nieoptymalne wyniki. Chociaż firma PGE osiągnęła stosunkowo wysoki wskaźnik poprawnych odpowiedzi (45,65%), ogólna skuteczność modelu pozostała poniżej oczekiwań, z łącznym wynikiem na poziomie 35,86% poprawnych odpowiedzi. Skuteczność modelu w poszczególnych firmach waha się, co sugeruje, że metoda RAG może być bardziej lub mniej skuteczna w zależności od specyfiki danych.

Niski procent błędów w jednostkach mierzalnych (6,90%) może niekoniecznie odzwierciedlać efektywność metody RAG w eliminowaniu tego typu problemów. Zamiast tego, niski wskaźnik może wynikać z faktu, że RAG przypadkowo odnalazł te dane, w których problem z jednostkami nie występował. Wyniki wskazują na potrzebę dalszego doskonalenia procesu wyszukiwania oraz integracji z technikami uczenia maszynowego, aby zwiększyć precyzję i niezawodność modelu w praktycznych zastosowaniach finansowych.

9.7 Podsumowanie

Przeprowadzone eksperymenty miały na celu ocenę zdolności modelu językowego do odpowiedzi na pytania oparte na danych finansowych.

Eksperyment 1, który testował model bez dodatkowej wiedzy, pokazał, że model efektywnie identyfikuje braki w swojej wiedzy, konsekwentnie odpowiadając ”Brak danych” na wszystkie pytania dotyczące danych poza zakresem jego treningu. To podkreśla zdolność modelu do unikania błędnych odpowiedzi, gdy nie dysponuje potrzebnymi informacjami.

Eksperyment 2 z ręcznym wstrzykiwaniem wiedzy wykazał, że manualne dodanie informacji pozwala modelowi na osiągnięcie wyższej precyzji w odpowiedziach, jednak różnice w efektywności między firmami sugerują potrzebę dalszego dostosowywania mo-

delu do specyfik danych. Wciąż obserwowano błędy, zwłaszcza w jednostkach, co wskazuje na potencjalne obszary do poprawy.

Eksperyment 3, łączący manualne wstrzykiwanie wiedzy z metodą few-shot learning, przyniósł największe ulepszenia w zdolnościach modelu do generowania poprawnych odpowiedzi, co pokazuje skuteczność tego podejścia. Zastosowanie przykładów w treningu modelu znacząco poprawiło jego efektywność, choć niektóre błędy wciąż wymagały rozwiązania.

Natomiast Eksperyment 4, z użyciem metody RAG oraz uczenia na podstawie niewielkiej liczby przykładów, dał mieszane wyniki, wskazujące na to, że pełna automatyzacja procesu wstrzykiwania wiedzy może być mniej efektywna w niektórych przypadkach. Choć niektóre firmy osiągnęły przyzwoite wyniki, ogólna skuteczność była niższa niż w poprzednich eksperymentach.

Podsumowując, różne metody wstrzykiwania wiedzy i techniki uczenia modelu mają istotny wpływ na jego zdolność odpowiedzi na pytania. Najlepsze wyniki osiągnięto łącząc ręczne wstrzykiwanie wiedzy z metodą few-shot learning, co znacząco poprawiło precyzję odpowiedzi modelu. Inne podejścia, takie jak RAG, wymagają dalszych usprawnień.

10 Zakończenie

10.1 Wnioski

Podsumowując, przeprowadzone badania empiryczne w niniejszej pracy magisterskiej wykazały, że duże modele językowe, takie jak GPT-4, mają znaczący potencjał w ekstrakcji informacji ze złożonych dokumentów finansowych. Ich skuteczność znacząco zależy jednak od metody dostarczania i wstrzykiwania wiedzy. Eksperymenty wyraźnie pokazały, że metody manualnego wstrzykiwania wiedzy, a także uczenie na podstawie niewielkiej liczby przykładów (few-shot learning), znacząco poprawiają zdolność modeli do generowania precyzyjnych odpowiedzi na zadane pytania. Z kolei metody automatyczne, takie jak generacja rozszerzona wyszukiwaniem (RAG), choć obiecujące, wymagają dalszego rozwoju i dostosowania do specyficznych wymagań i charakterystyk danych finansowych.

10.2 Potencjalne kierunki rozwoju

Na podstawie wyników badań oraz analizy istniejących ograniczeń dużych modeli językowych w domenie finansowej, można zidentyfikować kilka kluczowych kierunków dla przyszłych badań i rozwoju technologii:

- **Udoskonalanie metod wstrzykiwania wiedzy:** Badania wykazały, że metody wstrzykiwania wiedzy mają krytyczne znaczenie dla efektywności modeli w zadaniach ekstrakcji informacji. Rozwój bardziej zaawansowanych technik, które mogą automatycznie dostosowywać i integrować specyficzną wiedzę domenową, będzie kluczowy. Istotnym kierunkiem może być zbadanie ulepszeń w metodach takich jak Retrieval-Augmented Generation (RAG), które pozwalają na dynamiczne wyszukiwanie i integrację potrzebnych informacji w trakcie generowania odpowiedzi przez model. Efektywne wykorzystanie RAG może wymagać zastosowania bardziej precyzyjnych modeli do obliczania wektorów osadzeń (embeddings), które mogą być liczone na różnych poziomach granularności — od pojedynczych słów po całe akapity lub zdania. Dodatkowo, może być korzystne zastosowanie uproszczonych metod identyfikacji kluczowych sekcji dokumentów, jak spisy treści, które umożliwią szybkie lokalizowanie i integrację najbardziej relewantnych informacji

bez konieczności przetwarzania całych dokumentów.

- **Badanie potencjału modeli o rozszerzonym kontekście:** Modele używane w badaniach, takie jak GPT-4, posiadają ograniczony zakres kontekstu, co może wpływać na ich zdolności do przetwarzania i analizy długich dokumentów finansowych. Nowe modele, takie jak Gemini 1.5 Pro, które oferują możliwość przetwarzania kontekstu do 1 miliona tokenów[7], otwierają nowe perspektywy dla przyszłych badań. Nieznany jest jeszcze wpływ tak dużej możliwości kontekstowej na skuteczność modelu, w szczególności w metodach takich jak few-shot learning, gdzie model może wymagać znacznie więcej przykładów dla efektywnej nauki. Jednakże większe okno kontekstowe może zmniejszyć konieczność stosowania metod takich jak RAG, które w przeprowadzonych badaniach okazały się mniej efektywne. Potencjał ograniczenia zależności od zewnętrznych systemów wyszukiwania informacji to istotna korzyść, która może znacząco zwiększyć precyzję i szybkość przetwarzania danych.

- **Optymalizacja rozmiaru kontekstu dla efektywności kosztowej:** W kontekście rosnących kosztów operacyjnych i wymagań obliczeniowych związanych z dużymi modelami językowymi, badania nad minimalizacją rozmiaru kontekstu, przy jednoczesnym zachowaniu efektywności, stają się kluczowe. Optymalizacja ta ma na celu redukcję czasu przetwarzania i zasobów potrzebnych do działania modelu, co jest szczególnie ważne w analizie złożonych dokumentów, takich jak raporty finansowe w formacie HTML. Istotne będzie zbadanie, które elementy strukturalne tych dokumentów, takie jak tagi HTML czy metadane, mogą być pominięte bez negatywnego wpływu na zdolność modelu do dokładnej ekstrakcji i analizy treści.

- **Badanie wpływu formatu dokumentów na skuteczność modeli:** Przeprowadzone badania skoncentrowane były na analizie dokumentów finansowych w formacie HTML. Warto jednak zbadać skuteczność modeli w przetwarzaniu innych formatów, takich jak PDF oraz XBRL, który coraz częściej jest stosowany do publikacji raportów finansowych. XBRL (eXtensible Business Reporting Language) umożliwia strukturalne przedstawienie danych finansowych, co może znacząco wpłynąć na skuteczność modeli w ekstrakcji i analizie danych.

[7] `https://blog.google/technology/ai/long-context-window-ai-models/` (data dostępu: 8.06.2024)

11 Załączniki

11.1 Kod źródłowy

```
from textwrap import dedent

experiment1_prompt = {
    'System message': dedent('''
        Jesteś pomocnym asystentem.
        Odpowiadaj na podstawie danych pozyskanych ze skonsolidowanego
        ↪ sprawozdania finansowego lub jednostkowego, jeśli spółka tylko takie
        ↪ publikuje.
        Odpowiadaj na pytania po polsku.
    ''').strip(),
    'User message': dedent('''
        Jakie były wartości (kwoty) w spółce [company] w roku [year] dla poniżej
        ↪ wymienionych pozycji finansowych?
        Wartości dla pozycji mogą być bezpośrednio umieszczone w sprawozdaniu pod
        ↪ tą samą nazwą, pod inną (analogiczną) nazwą lub mogą być kombinacją
        ↪ (np. sumą lub różnicą) kilku pozycji.
        Twoja odpowiedź musi być w następującym formacie:

        Pozycja,Wartość
        <<Pozycja 1>>,<<Wartość 1>>
        <<Pozycja 2>>,<<Wartość 2>>
        ...

        Jeżeli nie masz dostępu do danych, w miejce brakującej wartości wpisz
        ↪ "Brak danych". Odpowiedź powinna składać się tylko z odpowiedzi
        ↪ sformatowanej jak przedstawiono powyżej.

        Oto pozycje:
        [positions]
    ''').lstrip()
}
```

Listing 1: Prompt dla eksperymentu nr 1

```
from textwrap import dedent

experiment2_prompt = {
    'System message': dedent('''
        Jesteś pomocnym asystentem.
        Odpowiadaj na podstawie danych pozyskanych ze skonsolidowanego
        ↪ sprawozdania finansowego lub jednostkowego, jeśli spółka tylko takie
        ↪ publikuje.
        Odpowiadaj na pytania po polsku.
    ''').strip(),
    'User message': dedent('''
        Zapoznaj się z poniższymi danymi pochodzącymi ze skonsolidowanego
        ↪ sprawozdania finansowegeo spółki [company] w roku [year], a następnie
        ↪ bazując na nich udziel odpowiedzi na pytanie.

        [source]

        Jakie były wartości (kwoty) w spółce [company] w roku [year] dla poniżej
        ↪ wymienionych pozycji finansowych?
        Wartości dla pozycji mogą być bezpośrednio umieszczone w sprawozdaniu pod
        ↪ tą samą nazwą, pod inną (analogiczną) nazwą lub mogą być kombinacją
        ↪ (np. sumą lub różnicą) kilku pozycji.
        Wartości muszą być podawane w tys. PLN! Jeśli raport podaje dane w innej
        ↪ jednostce, skonwertuj do tys. PLN.
        Twoja odpowiedź musi być w następującym formacie:

        Pozycja,Wartość
        <<Pozycja 1>>,<<Wartość 1>>
        <<Pozycja 2>>,<<Wartość 2>>
        ...
```

```
        Jeżeli nie masz dostępu do danych, w miejce brakującej wartości wpisz
        ↪ "Brak danych". Odpowiedź powinna składać się tylko z odpowiedzi
        ↪ sformatowanej jak przedstawiono powyżej.

        Oto pozycje:
        [positions]
    ''').lstrip()
}
```

Listing 2: Prompt dla eksperymentu nr 2 wykorzystujący wstrzykiwanie wiedzy

```
from textwrap import dedent

experiment3_prompt = {
    'System message': dedent('''
        Jesteś pomocnym asystentem.
        Odpowiadaj na podstawie danych pozyskanych ze skonsolidowanego
        ↪ sprawozdania finansowego lub jednostkowego, jeśli spółka tylko takie
        ↪ publikuje.
        Odpowiadaj na pytania po polsku.
    ''').strip(),
    'User message': dedent('''
        Zapoznaj się z poniższymi danymi pochodzącymi ze skonsolidowanego
        ↪ sprawozdania finansowego spółki [company] w roku [example_year].

        [example_source]

        Na podstawie tych danych można wywnioskować następujące wartości pozycji
        ↪ finansowych.
        Wartości podane są w tys. PLN:

        Pozycja,Wartość
        [example_position_values]
```

```
        Zapoznaj się z poniższymi danymi pochodzącymi ze skonsolidowanego
        ↪ sprawozdania finansowegeo spółki [company] w roku [year], a następnie
        ↪ bazując na nich oraz na przykładzie podanym wcześniej, udziel
        ↪ odpowiedzi na pytanie.

        [source]
        Jakie były wartości (kwoty) w spółce [company] w roku [year] dla poniżej
        ↪ wymienionych pozycji finansowych?
        Wartości dla pozycji mogą być bezpośrednio umieszczone w sprawozdaniu pod
        ↪ tą samą nazwą, pod inną (analogiczną) nazwą lub mogą być kombinacją
        ↪ (np. sumą lub różnicą) kilku pozycji.
        Wartości muszą być podawane w tys. PLN! Jeśli raport podaje dane w innej
        ↪ jednostce, skonwertuj do tys. PLN.
        Twoja odpowiedź musi być w następującym formacie:

        Pozycja,Wartość
        <<Pozycja 1>>,<<Wartość 1>>
        <<Pozycja 2>>,<<Wartość 2>>
        ...

        Jeżeli nie masz dostępu do danych, w miejce brakującej wartości wpisz
        ↪ "Brak danych". Odpowiedź powinna składać się tylko z odpowiedzi
        ↪ sformatowanej jak przedstawiono powyżej.

        Oto pozycje:
        [positions]
    ''').lstrip()
}
```

Listing 3: Prompt dla eksperymentu 3 wykorzystujący wstrzykiwanie wiedzy i metodę few-shot learning

```
from html import unescape
import re

def clean_text(text):
    text = unescape(text)
    text = re.sub(r'\\s+', ' ', text)
    text = text.replace(u'\\xa0', ' ')
    text = text.strip()
    return text

from bs4 import BeautifulSoup, NavigableString

def clean_html(html_content):
    soup = BeautifulSoup(html_content, 'html.parser')

    # Remove all <img> tags
    for img in soup.findAll('img'):
        img.decompose()

    # Remove 'style' and 'class' attributes from all tags
    for tag in soup.findAll():
        if tag.has_attr('style'):
            del tag['style']
        if tag.has_attr('class'):
            del tag['class']

    def remove_unnecessary_tags(tag):
        if isinstance(tag, NavigableString):
            return

        for child in list(tag.children):
            remove_unnecessary_tags(child)
```

```
        # Remove tags that start with 'ix' or are <span> tags
        if tag.name and (tag.name.startswith('ix') or tag.name == 'span'):
            contents_to_reinsert = list(tag.contents)
            parent = tag.parent
            if parent:
                for content in reversed(contents_to_reinsert):
                    tag.insert_before(content)
                tag.decompose()

    remove_unnecessary_tags(soup.body)

    return soup.prettify()
```

Listing 4: Czyszczenie treści HTML i tekstowych przy użyciu BeautifulSoup i wyrażeń regularnych na potrzeby eksperymentu nr 4

```
from openai import OpenAI

OPEN_API_KEY='<<OPEN_API_KEY>>'
client = OpenAI(api_key=OPEN_API_KEY)

def get_embedding(text, model='text-embedding-3-large'):
   text = text.replace("\\n", " ")
   return client.embeddings.create(input=[text], model=model).data[0].embedding
```

Listing 5: Generowanie osadzeń za pomocą API OpenAI z możliwością wyboru modelu na potrzeby eksperymentu nr 4

Literatura

[1] OpenAI *et al.*, *Gpt-4 technical report*, 2024. arXiv: 2303.08774 [cs.CL].

[2] G. Team *et al.*, *Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context*, 2024. arXiv: 2403.05530 [cs.CL].

[3] S. Bubeck *et al.*, *Sparks of artificial general intelligence: Early experiments with gpt-4*, 2023. arXiv: 2303.12712 [cs.CL].

[4] D. Xu *et al.*, *Large language models for generative information extraction: A survey*, 2023. arXiv: 2312.17617 [cs.CL].

[5] J. Lee, N. Stevens, S. C. Han, and M. Song, *A survey of large language models in finance (finllms)*, 2024. arXiv: 2402.02315 [cs.CL].

[6] Y. Li, S. Wang, H. Ding, and H. Chen, *Large language models in finance: A survey*, 2023. arXiv: 2311.10723 [q-fin.GN].

[7] D. Araci, *Finbert: Financial sentiment analysis with pre-trained language models*, 2019. arXiv: 1908.10063 [cs.CL].

[8] Y. Yang, M. C. S. UY, and A. Huang, *Finbert: A pretrained language model for financial communications*, 2020. arXiv: 2006.08097 [cs.CL].

[9] Z. Liu, D. Huang, K. Huang, Z. Li, and J. Zhao, "Finbert: A pre-trained financial language representation model for financial text mining," in *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence*, ser. IJCAI'20, Yokohama, Yokohama, Japan, 2021, ISBN: 9780999241165.

[10] S. Wu *et al.*, *Bloomberggpt: A large language model for finance*, 2023. arXiv: 2303.17564 [cs.LG].

[11] S. Zhang *et al.*, *Instruction tuning for large language models: A survey*, 2024. arXiv: 2308.10792 [cs.CL].

[12] Q. Xie *et al.*, *Pixiu: A large language model, instruction data and evaluation benchmark for finance*, 2023. arXiv: 2306.05443 [cs.CL].

[13] H. Touvron *et al.*, *Llama: Open and efficient foundation language models*, 2023. arXiv: 2302.13971 [cs.CL].

[14] Y. Yang, Y. Tang, and K. Y. Tam, *Investlm: A large language model for investment using financial domain instruction tuning*, 2023. arXiv: 2309.13064 [q-fin.GN].

[15] H. Yang, X.-Y. Liu, and C. D. Wang, *Fingpt: Open-source financial large language models*, 2023. arXiv: 2306.06031 [q-fin.ST].

[16] N. Wang, H. Yang, and C. D. Wang, *Fingpt: Instruction tuning benchmark for open-source large language models in financial datasets*, 2023. arXiv: 2310.04793 [cs.CL].

[17] C. E. Shannon, "Prediction and entropy of printed english," *Bell System Technical Journal*, vol. 30, no. 1, pp. 50–64, 1951.

[18] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient estimation of word representations in vector space*, 2013. arXiv: 1301.3781 [cs.CL].

[19] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," 1985.

[20] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[21] M. V. Mahoney, "Fast text compression with neural networks," in *FLAIRS Conference*, 2000, pp. 230–234.

[22] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," in *Advances in Neural Information Processing Systems*, vol. 13, 2000.

[23] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems*, vol. 27, 2014.

[24] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.

[25] A. Vaswani *et al.*, *Attention is all you need*, 2023. arXiv: 1706.03762 [cs.CL].

[26] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, *Improving language understanding by generative pre-training*, 2018.

[27] A. Radford *et al.*, "Language models are unsupervised multitask learners," *OpenAI blog*, vol. 1, no. 8, p. 9, 2019.

[28] C. Raffel *et al.*, *Exploring the limits of transfer learning with a unified text-to-text transformer*, 2023. arXiv: 1910.10683 [cs.LG].

[29] T. B. Brown *et al.*, *Language models are few-shot learners*, 2020. arXiv: 2005.14165 [cs.CL].

[30] L. Ouyang *et al.*, *Training language models to follow instructions with human feedback*, 2022. arXiv: 2203.02155 [cs.CL].

[31] Y. Wang, Q. Yao, J. Kwok, and L. M. Ni, *Generalizing from a few examples: A survey on few-shot learning*, 2020. arXiv: 1904.05046 [cs.LG].

[32] Ministerstwo Finansów Polski, *Ustawa z dnia 29 września 1994 r. o rachunkowości*, Skonsolidowane sprawozdania finansowe grupy kapitałowej, Dziennik Ustaw, Tekst jednolity Dz.U.2023.0.120, 2023.

[33] X. Wei *et al.*, *Chatie: Zero-shot information extraction via chatting with chatgpt*, 2024. arXiv: 2302.10205 [cs.CL].

[34] X. Wang, S. Li, and H. Ji, *Code4struct: Code generation for few-shot event structure prediction*, 2023. arXiv: 2210.12810 [cs.CL].

[35] K. S. Phogat, C. Harsha, S. Dasaratha, S. Ramakrishna, and S. A. Puranam, *Zero-shot question answering over financial documents using large language models*, 2023. arXiv: 2311.14722 [cs.CL].

[36] J. Saad-Falcon *et al.*, *Pdftriage: Question answering over long, structured documents*, 2023. arXiv: 2309.08872 [cs.CL].

[37] F. Petroni *et al.*, *Language models as knowledge bases?* 2019. arXiv: 1909.01066 [cs.CL].

[38] W. Yu *et al.*, "A survey of knowledge-enhanced text generation," *ACM Computing Surveys*, vol. 54, no. 11s, pp. 1–38, Jan. 2022, ISSN: 1557-7341. DOI: 10.1145/3512467. [Online]. Available: http://dx.doi.org/10.1145/3512467.

[39] K. Singhal *et al.*, "Large language models encode clinical knowledge," *Nature*, vol. 620, no. 7972, pp. 172–180, 2023.

[40] Q. Huang *et al.*, *Lawyer llama technical report*, 2023. arXiv: 2305.15062 [cs.CL].

[41] P. Lewis *et al.*, *Retrieval-augmented generation for knowledge-intensive nlp tasks*, 2021. arXiv: 2005.11401 [cs.CL].

[42] C. Wang *et al.*, *Survey on factuality in large language models: Knowledge, retrieval and domain-specificity*, 2023. arXiv: 2310.07521 [cs.CL].

[43] O. Ovadia, M. Brief, M. Mishaeli, and O. Elisha, *Fine-tuning or retrieval? comparing knowledge injection in llms*, 2024. arXiv: 2312.05934 [cs.AI].

[44] U. Shaham, M. Ivgi, A. Efrat, J. Berant, and O. Levy, *Zeroscrolls: A zero-shot benchmark for long text understanding*, 2023. arXiv: 2305.14196 [cs.CL].

[45] C. Wang *et al.*, *Novelqa: A benchmark for long-range novel question answering*, 2024. arXiv: 2403.12766 [cs.CL].

[46] W. Chen, *Large language models are few(1)-shot table reasoners*, 2023. arXiv: 2210.06710 [cs.CL].

Spis rysunków

1 Przykłady raportów giełdowych tworzących bazę wiedzy 36

Spis tabel

1 Koszt różnych opcji LLM 24
2 Przykład pozycji finansowych tworzących zbiór pytań i odpowiedzi . . . 37
3 Wyniki eksperymentu nr 1 41
4 Wyniki eksperymentu nr 2 42
5 Wyniki eksperymentu nr 3 43
6 Wyniki eksperymentu nr 4 45