Question Answering w NLP

Extractive QA · Generative QA · RAG

Cel materiału:

Zrozumienie koncepcji, historii i zastosowań systemów odpowiadania na pytania przed przystąpieniem do ćwiczeń laboratoryjnych

1. Wprowadzenie – Przetwarzanie Języka Naturalnego

Przetwarzanie Języka Naturalnego (ang. Natural Language Processing, NLP) to dziedzina sztucznej inteligencji zajmująca się umożliwieniem komputerom rozumienia, interpretacji i generowania ludzkiego języka. NLP łączy w sobie elementy lingwistyki, informatyki i uczenia maszynowego, stawiając sobie za cel budowanie systemów zdolnych do komunikacji z człowiekiem w sposób naturalny i intuicyjny.

Język naturalny jest z natury niejednoznaczny, kontekstowy i pełen niuansów kulturowych. Zdanie "Zamknij okno" może oznaczać prośbę o przymknięcie okna w pokoju lub polecenie zamknięcia okna aplikacji komputerowej. Właśnie ta wieloznaczność sprawia, że NLP jest dziedziną wyjątkowo trudną i fascynującą zarazem.

1.1. Kluczowe zadania w NLP

NLP obejmuje szeroki wachlarz zadań badawczych i praktycznych. Do najważniejszych należą:

- Klasyfikacja tekstu – przypisywanie dokumentu do jednej lub wielu kategorii (np. analiza sentymentu, wykrywanie spamu).
- Rozpoznawanie bytów nazwanych (NER) – identyfikacja w tekście nazw osób, miejsc, organizacji, dat itp.
- Tłumaczenie maszynowe – automatyczne przekładanie tekstu z jednego języka na inny.
- Streszczanie tekstu – generowanie skróconej wersji dłuższego dokumentu.
- Question Answering (QA) – automatyczne udzielanie odpowiedzi na pytania zadane w języku naturalnym.
- Generowanie tekstu – tworzenie spójnego i sensownego tekstu na podstawie podanego początku lub opisu tematu.

Dlaczego QA jest ważne?
Question Answering zajmuje szczególne miejsce wśród zadań NLP, ponieważ stanowi swego rodzaju "miarę" ogólnego rozumienia języka przez maszynę. System zdolny do poprawnego odpowiadania na dowolne pytania musi rozumieć słownictwo, gramatykę, logikę i fakty o świecie. Z tego powodu benchmarki QA – jak SQuAD, TriviaQA czy Natural Questions – są powszechnie używane do oceny postępu modeli językowych.
Praktyczne zastosowania QA są ogromne: od wyszukiwarek internetowych, przez asystentów głosowych (Siri, Alexa, Google Assistant), po systemy obsługi klienta i narzędzia wspomagające pracę lekarzy czy prawników.

2. Czym jest Question Answering (QA)?

Question Answering (QA) to zadanie NLP polegające na automatycznym udzielaniu odpowiedzi na pytania sformułowane w języku naturalnym. System QA przyjmuje pytanie (ang. query) – a

często także kontekst (ang. context lub passage) – i zwraca odpowiedź w formie czytelnej dla człowieka.

"Kiedy wynaleziono internet?" "w latach 60. i 70. XX wieku"

2.1. Historia i ewolucja QA

Systemy odpowiadania na pytania mają długą historię w badaniach nad sztuczną inteligencją. Pierwsze próby sięgają lat 60. XX wieku – system BASEBALL (1961) potrafił odpowiadać na pytania dotyczące wyników meczów baseballowych, a LUNAR (1971) był w stanie odpowiadać na pytania o próbki skał przywiezionych z Księżyca. Były to jednak systemy wąsko-specjalistyczne, oparte na regułach i bazach danych.

Przełom nastąpił wraz z pojawieniem się uczenia maszynowego i głębokich sieci neuronowych w XXI wieku. Kluczowe etapy tej ewolucji to:

- **2016:** Publikacja zbioru danych SQuAD (Stanford Question Answering Dataset) przez Rajpurkar i in. – stało się to punktem odniesienia dla badań nad Extractive QA.
- **2018:** Opublikowanie architektury BERT (Bidirectional Encoder Representations from Transformers) przez Google – rewolucja w jakości rozumienia tekstu.
- **2019–2020:** Modele GPT-2 i T5 pokazują, że modele generatywne mogą udzielać przekonujących odpowiedzi bez wskazanego kontekstu.
- **2020:** Facebook AI Research publikuje FAISS i RAG (Retrieval-Augmented Generation) – nowe podejście łączące wyszukiwanie i generowanie.
- **2022–2025:** Era dużych modeli językowych (LLM): GPT-4, Claude, Gemini, LLaMA – QA staje się jedną z podstawowych funkcji asystentów AI.

2.2. Ogólna architektura systemu QA

Niezależnie od wariantu, każdy system QA składa się z kilku podstawowych komponentów:

- Moduł wejściowy – przyjmuje pytanie użytkownika, opcjonalnie przetwarza je (tokenizacja, lematyzacja).
- Moduł rozumienia pytania – analizuje intencję pytania i wymagany typ odpowiedzi (osoba, data, liczba, opis itd.).
- Moduł wyszukiwania lub rozumowania – pobiera relewantne informacje ze źródła (kontekst, baza danych, internet) lub korzysta z wiedzy zapisanej w parametrach modelu.
- Moduł generowania odpowiedzi – formułuje finalną odpowiedź w formie fragmentu tekstu lub nowo wygenerowanego zdania.

3. Extractive QA – Wydobywanie Odpowiedzi z Kontekstu

Extractive QA (ekstrakcyjne odpowiadanie na pytania) to podejście, w którym model nie tworzy nowej treści, lecz lokalizuje odpowiedź jako fragment (ang. span) w podanym mu tekście. Innymi słowy: system wskazuje, w którym miejscu kontekstu znajduje się odpowiedź i zwraca ten fragment dosłownie.

3.1. Zasada działania

W Extractive QA model otrzymuje na wejściu dwa elementy:

- **Pytanie (question):** np. "Kto był pierwszym prezydentem USA?"
- **Kontekst (context/passage):** fragment tekstu (np. artykuł Wikipedii), w którym odpowiedź powinna się znajdować.

Zadaniem modelu jest wskazanie pozycji startowej i końcowej odpowiedzi w kontekście (ang. start token i end token). W efekcie odpowiedź jest dosłownym cytatem z tekstu, bez żadnych modyfikacji. Jeśli odpowiedź nie istnieje w kontekście, model powinien zgłosić brak odpowiedzi (ang. "no answer" lub "unanswerable").

Przykład działania Extractive QA
Pytanie: "W którym roku powstała organizacja ONZ?"
Kontekst: "Organizacja Narodów Zjednoczonych (ONZ) została założona 24 października 1945 roku, po zakończeniu II wojny światowej, z inicjatywy 51 państw."
Odpowiedź modelu: "24 października 1945 roku" (wyodrębniony span z kontekstu)

3.2. Kluczowe modele i architektury

BERT – Bidirectional Encoder Representations from Transformers

BERT, opublikowany przez Google w 2018 roku, jest fundamentalnym modelem dla Extractive QA. Oparty na architekturze Transformera, BERT przetwarza tekst dwukierunkowo – analizuje każde słowo w kontekście zarówno słów poprzedzających, jak i następujących po nim. Dzięki temu rozumie kontekst znacznie lepiej niż wcześniejsze modele sekwencyjne.

BERT jest wstępnie trenowany (ang. pre-trained) na ogromnych zbiorach tekstu (np. cała Wikipedia po angielsku), a następnie dostrajany (ang. fine-tuned) na konkretnym zadaniu – np. na zbiorze SQuAD. Taka strategia transfer learningu pozwala uzyskać doskonałe wyniki przy stosunkowo niewielkim zbiorze danych treningowych dla docelowego zadania.

SQuAD – Stanford Question Answering Dataset

SQuAD to zbiór danych stworzony na Uniwersytecie Stanforda, który zrewolucjonizował badania nad Extractive QA. Wersja 1.1 (2016) zawiera ponad 100 000 par pytanie-odpowiedź opartych na

artykułach Wikipedii, gdzie każda odpowiedź jest fragmentem kontekstu. Wersja SQuAD 2.0 (2018) dodała pytania bez odpowiedzi, testując zdolność modelu do rozpoznania, że odpowiedzi nie ma w tekście.

Inne ważne modele

- RoBERTa (Facebook, 2019) – ulepszona wersja BERT z bardziej optymalnym procesem trenowania, osiągająca lepsze wyniki na SQuAD.
- ALBERT (Google, 2019) – lżejsza wersja BERT z mechanizmem dzielenia parametrów, redukująca liczbę parametrów przy zachowaniu wydajności.
- DistilBERT (Hugging Face, 2019) – skompresowany BERT (40% mniej parametrów), 60% szybszy w działaniu, przy zachowaniu 97% jakości.
- DeBERTa (Microsoft, 2020) – model z ulepszonym mechanizmem uwagi, osiągający stan wiedzy na wielu benchmarkach QA.

3.3. Zastosowania Extractive QA

Extractive QA sprawdza się szczególnie dobrze w następujących scenariuszach:

- **Bazy wiedzy i FAQ:** systemy automatycznie odnajdują odpowiedzi w dokumentacji firmowej, regulaminach lub bazach FAQ, odpowiadając na pytania klientów.
- **Medycyna i prawo:** wyszukiwanie konkretnych informacji w artykułach naukowych, orzeczeniach sądowych lub dokumentach medycznych.
- **Wyszukiwarki semantyczne:** zamiast listy linków, użytkownik otrzymuje bezpośrednią odpowiedź wyodrębnioną z treści stron.
- **Asystenci czytający:** narzędzia wspomagające naukę, które odpowiadają na pytania studenta na podstawie podanego materiału edukacyjnego.

3.4. Ograniczenia Extractive QA

- Model nie może odpowiedzieć, jeśli odpowiedź nie jest dosłownie zawarta w kontekście – nie potrafi wnioskować ani parafrazować.
- Wymaga dostarczenia odpowiedniego kontekstu z góry – ktoś lub coś musi wcześniej wskazać właściwy fragment tekstu.
- Odpowiedzi są ograniczone do fragmentów istniejących dokumentów, co utrudnia udzielanie złożonych, wielozdaniowych wyjaśnień.

4. Generative QA – Generowanie Swobodnych Odpowiedzi

Generative QA (generatywne odpowiadanie na pytania) to podejście, w którym model samodzielnie konstruuje odpowiedź, zamiast ją wyodrębniać z podanego tekstu. Model może parafrazować, wnioskować, syntetyzować wiedzę z wielu źródeł i tworzyć płynne, pełne zdania – nawet jeśli odpowiedź nie jest dosłownie zapisana w żadnym dokumencie.

4.1. Zasada działania

W klasycznym Generative QA model (zazwyczaj tzw. model językowy, ang. Language Model) na wejściu otrzymuje pytanie, a na wyjściu generuje odpowiedź token po tokenie. Proces ten opiera się na prawdopodobieństwach warunkowych: model oblicza, jakie słowo (token) powinno pojawić się jako następne, biorąc pod uwagę wszystko, co zostało wygenerowane dotychczas.

Kluczową różnicą względem Extractive QA jest to, że odpowiedź powstaje de novo – jest nowo wygenerowanym tekstem, nie fragmentem istniejącego dokumentu. Oznacza to, że system posiada wiedzę "zakodowaną" w swoich parametrach podczas trenowania na ogromnych zbiorach danych.

Przykład działania Generative QA

Pytanie: "Wyjaśnij, czym jest entropia w termodynamice."

Odpowiedź modelu: "Entropia to miara nieuporządkowania lub losowości układu. W termodynamice opisuje kierunek, w jakim spontanicznie przebiegają procesy – układy dążą do stanów o wyższej entropii. Formalnie wyrażona jest wzorem $S = k \cdot \ln(W)$, gdzie k to stała Boltzmanna, a W liczba możliwych mikrostanów."

(Odpowiedź wygenerowana – nie wyodrębniona z konkretnego dokumentu)

4.2. Kluczowe modele

GPT – Generative Pre-trained Transformer

Seria modeli GPT opracowana przez OpenAI (GPT-2 w 2019, GPT-3 w 2020, GPT-4 w 2023) to ikony Generative QA. GPT oparty jest na architekturze dekodera Transformera i trenowany jest na zadaniu przewidywania następnego tokenu. Modele te wykazują zdolność do "few-shot learningu" – potrafią nauczyć się nowych zadań jedynie na podstawie kilku przykładów podanych w prompcie, bez dodatkowego trenowania.

T5 – Text-to-Text Transfer Transformer

T5, opracowany przez Google Research w 2019 roku, przyjmuje ujednolicone podejście: każde zadanie NLP jest traktowane jako problem zamiany jednego tekstu na inny. Odpowiadanie na pytania staje się więc po prostu transformacją: wejście to pytanie (i ewentualnie kontekst), wyjście to odpowiedź. T5 był trenowany na zbiorze C4 (Colossal Clean Crawled Corpus) o rozmiarze 750 GB.

FLAN-T5 – Fine-tuned Language Net T5

FLAN-T5 (Google, 2022) to wersja T5 dostrojona za pomocą techniki "instruction tuning" – model był trenowany na tysiącach różnorodnych zadań NLP, z wyraźnymi instrukcjami tekstowymi opisującymi każde zadanie. Dzięki temu FLAN-T5 znacznie lepiej generalizuje na nowe, niewidziane wcześniej zadania i jest jednym z najlepszych publicznie dostępnych modeli dla Open-Domain QA.

4.3. Open-Domain QA a Closed-Domain QA

Generatywne systemy QA można podzielić na dwa warianty ze względu na zakres dziedziny wiedzy:

Open-Domain QA	Closed-Domain QA
Pytania z dowolnej dziedziny wiedzy, bez ograniczeń tematycznych	Pytania ograniczone do konkretnej dziedziny (np. medycyna, prawo, finanse)
Przykłady: GPT-4, Claude, FLAN-T5	Przykłady: modele medyczne (BioGPT), prawnicze, finansowe

4.4. Zjawisko halucynacji

Kluczowym wyzwaniem Generative QA jest tzw. halucynacja (ang. hallucination) – tendencja modeli do generowania odpowiedzi, które brzmią przekonująco, ale są faktycznie nieprawdziwe. Model "nie wie, czego nie wie" i zamiast przyznać się do braku wiedzy, generuje plausible-sounding, ale błędne informacje.

Halucynacje wynikają z samej natury modeli językowych: ich celem jest generowanie prawdopodobnych sekwencji tokenów, nie weryfikacja faktów. Z tego powodu Generative QA wymaga ostrożności w zastosowaniach wymagających wysokiej dokładności faktograficznej.

5. Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) to architektura łącząca dwa paradygmaty: wyszukiwanie informacji (ang. retrieval) i generowanie tekstu (ang. generation). Zaproponowana przez Lewisa i współpracowników z Facebook AI Research w 2020 roku, RAG stanowi odpowiedź na dwa kluczowe ograniczenia: halucynacje modeli generatywnych oraz statyczną wiedzę zakodowaną w parametrach modelu.

5.1. Motywacja i geneza

Wyobraź sobie eksperta, który przed udzieleniem odpowiedzi najpierw sięga do biblioteki, czyta odpowiednie fragmenty i dopiero potem formułuje odpowiedź. RAG działa na podobnej zasadzie: zamiast polegać wyłącznie na wiedzy "zapamiętanej" podczas trenowania, model najpierw wyszukuje aktualne, relewantne dokumenty, a następnie generuje odpowiedź na ich podstawie.

Geneza RAG wywodzi się z obserwacji, że czyste modele generatywne mają "datę ważności" wiedzy (ang. knowledge cutoff) – nie wiedzą nic o zdarzeniach po dacie zakończenia trenowania. RAG rozwiązuje ten problem, pozwalając modelowi korzystać z aktualnych, zewnętrznych baz wiedzy.

5.2. Architektura RAG – trzy fazy

Działanie systemu RAG przebiega w trzech wyraźnie odróżnialnych fazach:

FAZA 1	**Retrieval – Pobieranie dokumentów** Pytanie użytkownika jest przekształcane w wektor (embedding) i porównywane z wektorami dokumentów w bazie wiedzy. Najlepiej pasujące fragmenty (np. top-5) są pobierane jako kontekst.
FAZA 2	**Augmentation – Wzbogacenie promptu** Pobrane fragmenty dokumentów są dołączane do oryginalnego pytania, tworząc wzbogacony prompt. Model generatywny otrzymuje teraz pytanie + relewantny kontekst.
FAZA 3	**Generation – Generowanie odpowiedzi** Model językowy (np. GPT, T5, LLaMA) generuje końcową odpowiedź na podstawie wzbogaconego promptu. Odpowiedź jest zakorzeniona w rzeczywistych dokumentach, co redukuje halucynacje.

5.3. Komponenty techniczne RAG

Embeddingi i wektory semantyczne

Kluczowym elementem fazy retrieval jest konwersja tekstu na reprezentacje wektorowe (embeddingi). Modele embeddingów (np. sentence-transformers, text-embedding-ada-002 od OpenAI) przekształcają zdania i akapity w wektory liczb rzeczywistych w przestrzeni o wysokim wymiarze (np. 768 lub 1536 wymiarów). Podobne semantycznie teksty mają w tej przestrzeni bliskie sobie wektory.

Bazy wektorowe (Vector Databases)

Do efektywnego przeszukiwania milionów wektorów używane są specjalne bazy danych, takie jak FAISS (Facebook), Pinecone, Weaviate, Chroma czy Qdrant. Pozwalają one na szybkie wyszukiwanie najbliższych sąsiadów (ang. Approximate Nearest Neighbor search, ANN), znajdując najbardziej podobne dokumenty do zapytania w czasie subsekundowym.

Chunking – podział dokumentów

Przed dodaniem dokumentów do bazy wektorowej należy je podzielić na mniejsze fragmenty (ang. chunks). Rozmiar chunka jest ważnym hiperparametrem: zbyt małe chunki tracą kontekst, zbyt duże przekraczają limit tokenów modelu. Typowe rozmiary to 256–1024 tokeny z nakładającymi się fragmentami (ang. overlapping chunks) dla zachowania ciągłości.

5.4. Zalety i wyzwania RAG

RAG łączy mocne strony obu podejść:

- **Aktualność wiedzy:** baza dokumentów może być aktualizowana bez ponownego trenowania całego modelu.
- **Możliwość weryfikacji:** odpowiedzi można śledzić do konkretnych źródeł (ang. grounding), co redukuje halucynacje.
- **Elastyczność:** ten sam model językowy może być używany z różnymi bazami wiedzy dla różnych zastosowań.

Wyzwania RAG:

- **Jakość retrieval:** jeśli system pobierze nieodpowiednie dokumenty, model wygeneruje błędną odpowiedź ("garbage in, garbage out").
- **Latencja:** dodatkowy krok wyszukiwania zwiększa czas odpowiedzi.
- **Złożoność architektury:** integracja embeddingów, bazy wektorowej i modelu generatywnego wymaga więcej komponentów.

5.5. RAG w praktyce – popularne frameworki

- LangChain – najpopularniejszy framework do budowania aplikacji opartych na LLM, z wbudowanym wsparciem dla RAG.
- LlamaIndex (dawniej GPT Index) – specjalizowany framework do indeksowania danych i budowania RAG pipeline'ów.
- Haystack (deepset) – open-source'owy framework NLP z kompleksowym wsparciem dla systemów QA i RAG.

6. Porównanie Podejść

Poniższa tabela zbiera najważniejsze różnice między trzema omawianymi podejściami do QA. Zrozumienie tych różnic jest kluczowe dla świadomego wyboru odpowiedniej architektury dla konkretnego zastosowania.

Cecha	Extractive QA	Generative QA	RAG
Źródło odpowiedzi	Fragment kontekstu	Własna wiedza modelu	Pobrane dokumenty
Wymaga kontekstu	Tak (obowiązkowy)	Nie	Nie (pobiera sam)
Ryzyko halucynacji	Niskie	Wysokie	Niskie-średnie
Aktualizacja wiedzy	Poprzez kontekst	Wymaga re-treningu	Poprzez bazę dokumentów
Przykładowe modele	BERT, RoBERTa	GPT-4, T5, FLAN-T5	BERT + GPT / LlamaIndex
Typowe zastosowanie	Asystenci z bazą FAQ	Chatboty ogólne	Asystenci korporacyjne

6.1. Kiedy wybrać które podejście?

Wybór architektury QA powinien być podyktowany specyfiką problemu:

Extractive QA – wybierz gdy:
Dysponujesz dobrze zdefiniowanym korpusem dokumentów (FAQ, dokumentacja, artykuły).
Wymagana jest wysoka wiarygodność – odpowiedź musi być dosłownym fragmentem zaufanego źródła.
Ważne jest wskazanie proweniencji odpowiedzi (skąd pochodzi fragment).
Zasoby obliczeniowe są ograniczone – modele extractive są zazwyczaj lżejsze.

Generative QA – wybierz gdy:
Pytania mają otwarty charakter i wymagają syntezy wiedzy z wielu źródeł.
Odpowiedzi powinny być płynne, rozbudowane i pisane naturalnym językiem.
Akceptowalne jest pewne ryzyko błędów faktograficznych (np. aplikacje edukacyjne z weryfikacją).
Brak stałej bazy dokumentów – model ma odpowiadać na podstawie wiedzy ogólnej.

RAG – wybierz gdy:

Wiedza musi być aktualna i możliwa do aktualizacji bez re-trenowania modelu.

Pytania dotyczą specyficznej, wewnętrznej wiedzy organizacji (dokumenty firmowe, raporty).

Wymagana jest zarówno płynność odpowiedzi, jak i grounding w konkretnych źródłach.

Chcesz minimalizować halucynacje modelu generatywnego.

7. Ewaluacja Systemów QA

Ocena jakości systemów QA jest nietrywialnym zadaniem. Odpowiedź "Paryż" i "miasto Paryż we Francji" są równie poprawne na pytanie o stolicę Francji, ale jeden ze standardowych modułów ewaluacyjnych może je ocenić inaczej. Dlatego w badaniach nad QA stosuje się kilka uzupełniających się metryk.

7.1. Metryki dla Extractive QA

- **Exact Match (EM):** miara binarna – 1 jeśli wygenerowana odpowiedź jest dokładnie identyczna z odpowiedzią referencyjną (po normalizacji: usunięciu interpunkcji, małe litery), 0 w przeciwnym razie.
- **F1 Score:** miara częściowego pokrycia – obliczana na poziomie tokenów między odpowiedzią modelu a odpowiedzią referencyjną. F1 jest średnią harmoniczną precyzji i pokrycia.

7.2. Metryki dla Generative QA

- **BLEU (Bilingual Evaluation Understudy):** miara oparta na n-gramach, pierwotnie opracowana dla tłumaczenia maszynowego. Porównuje n-gramy wygenerowanej odpowiedzi z odpowiedziami referencyjnymi.
- **ROUGE (Recall-Oriented Understudy for Gisting Evaluation):** rodzina metryk popularnych przy streszczaniu tekstu i QA. ROUGE-L mierzy najdłuższą wspólną podsequencję.
- **BERTScore:** zamiast porównywania n-gramów, używa embeddingów BERT do mierzenia semantycznego podobieństwa między odpowiedzią modelu a odpowiedzią referencyjną – lepsza korelacja z oceną ludzką.

7.3. Popularne benchmarki QA

Benchmark	Typ QA	Opis
SQuAD 1.1 / 2.0	Extractive	100k+ pytań z artykułów Wikipedii; v2.0 dodaje pytania bez odpowiedzi
TriviaQA	Open-Domain	Pytania quizowe z dokumentami dowodowymi z Wikipedii i internetu
Natural Questions	Extractive / Gen.	Pytania z prawdziwych zapytań Google z odpowiedziami z Wikipedii
HotpotQA	Extractive	Wieloetapowe wnioskowanie wymagające syntezy wielu dokumentów
MS MARCO	Generative	Pytania z Bing Search z odpowiedziami generowanymi przez ludzi
MMLU	Generative	57 dziedzin wiedzy akademickiej – test rozumowania wielodziedzinowego

8. Słownik Pojęć

Poniżej zebrano kluczowe pojęcia, których znajomość ułatwi zrozumienie materiału laboratoryjnego oraz literatury naukowej z zakresu QA i NLP.

Attention Mechanism

> Mechanizm uwagi – komponent modeli Transformer pozwalający modelowi skupić się na różnych częściach wejściowego tekstu przy generowaniu każdego tokenu wyjściowego. Podstawa sukcesu BERT i GPT.

Chunking

> Podział dokumentów na mniejsze fragmenty przed indeksowaniem w bazie wektorowej. Kluczowy krok w budowie systemu RAG.

Context Window

> Okno kontekstu – maksymalna liczba tokenów, którą model może przetworzyć jednorazowo. Ogranicza długość tekstu wejściowego.

Embedding

> Reprezentacja wektorowa tekstu (słowa, zdania, akapitu) w przestrzeni wielowymiarowej. Podobne semantycznie teksty mają bliskie embeddingi.

Fine-tuning

> Dostrajanie wstępnie nauczonego modelu na konkretnym zadaniu lub zbiorze danych. Pozwala uzyskać wysoką jakość przy małej ilości danych treningowych.

Hallucination

> Halucynacja – generowanie przez model informacji, które brzmią wiarygodnie, ale są faktycznie nieprawdziwe. Poważny problem modeli generatywnych.

Instruction Tuning

> Technika fine-tuningu, w której model uczy się na zbiorze różnorodnych zadań opisanych instrukcjami tekstowymi. Stosowana w FLAN-T5.

Knowledge Cutoff

> Data graniczna wiedzy modelu – model nie posiada informacji o zdarzeniach po tej dacie. Jeden z powodów stosowania RAG.

Perplexity

> Miara "zaskoczenia" modelu językowego – jak bardzo model jest zaskoczony testowym tekstem. Niższy perplexity = lepszy model językowy.

Pre-training

> Wstępne trenowanie modelu na ogromnych zbiorach danych (np. całej Wikipedii) przed dostrajaniem na konkretne zadanie.

Span

> Fragment (ang. span) – ciągły fragment tekstu wskazany jako odpowiedź w Extractive QA. Definiowany przez pozycję startową i końcową.

Token

> Podstawowa jednostka tekstu dla modeli językowych. Może być słowem, częścią słowa lub znakiem interpunkcyjnym. Modele operują na tokenach, nie na znakach.

Tokenization

> Tokenizacja – proces podziału tekstu na tokeny. Różne modele używają różnych strategii tokenizacji (np. WordPiece w BERT, BPE w GPT).

Transfer Learning

> Transfer uczenia – technika wykorzystania wiedzy zdobytej przy rozwiązywaniu jednego problemu do rozwiązania innego, pokrewnego problemu.

Transformer

> Architektura sieci neuronowej oparta na mechanizmie uwagi, zaproponowana w pracy "Attention is All You Need" (Vaswani i in., 2017). Podstawa większości nowoczesnych modeli NLP.

Vector Database

> Baza wektorowa – wyspecjalizowana baza danych do przechowywania i efektywnego przeszukiwania embeddingów. Kluczowy komponent systemów RAG.

9. Literatura i Zasoby Dodatkowe

Poniższe zasoby stanowią punkt wyjścia do pogłębiania wiedzy z zakresu Question Answering i NLP:

Artykuły naukowe (fundamentalne)

- **Devlin i in. (2018):** "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding" – artykuł wprowadzający BERT, podstawę Extractive QA.
- **Rajpurkar i in. (2016):** "SQuAD: 100,000+ Questions for Machine Comprehension of Text" – opis zbioru danych SQuAD 1.1.
- **Raffel i in. (2019):** "Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer" – artykuł o modelu T5.
- **Lewis i in. (2020):** "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks" – oryginalna praca o architekturze RAG.
- **Vaswani i in. (2017):** "Attention is All You Need" – przełomowy artykuł wprowadzający architekturę Transformer.

Zasoby online

- **Hugging Face Hub (huggingface.co):** repozytorium modeli, zbiorów danych i tutoriali. Punkt startowy dla praktycznej pracy z modelami QA.
- **Papers With Code (paperswithcode.com):** zestawienie artykułów z wynikami na benchmarkach i linkami do kodu.
- **Stanford NLP (nlp.stanford.edu):** materiały Grupy NLP Stanforda, w tym kurs CS224N.
- **Coursera / DeepLearning.AI – NLP Specialization:** kursy online prowadzone przez Andrew Ng i jego zespół.

Dokumentacja bibliotek

- **Hugging Face Transformers:** dokumentacja biblioteki Python do pracy z BERT, GPT, T5 i innymi modelami.
- **LangChain Docs:** framework do budowania aplikacji LLM, w tym systemów RAG.
- **LlamaIndex Docs:** dokumentacja narzędzia do indeksowania danych i tworzenia RAG pipeline'ów.

Podsumowanie – kluczowe wnioski

Question Answering to fundamentalne zadanie NLP, oceniające zdolność maszyny do rozumienia języka naturalnego.

Extractive QA wskazuje odpowiedź jako fragment kontekstu – wysoka precyzja, ale ograniczona elastyczność. Kluczowe modele: BERT, RoBERTa. Kluczowy benchmark: SQuAD.

Generative QA tworzy odpowiedź od podstaw – elastyczność i naturalność, ale ryzyko halucynacji. Kluczowe modele: GPT, T5, FLAN-T5.
RAG łączy oba podejścia: najpierw pobiera relewantne dokumenty, następnie generuje odpowiedź na ich podstawie – redukuje halucynacje i umożliwia aktualną wiedzę.
Wybór podejścia zależy od wymagań: precyzja vs. elastyczność, aktualizacja wiedzy, zasoby obliczeniowe, wymagania co do wiarygodności odpowiedzi.