

Spis treści

Przedmowa

Podziękowania

O tej książce

O autorze

1. Czym są duże modele językowe?

- 1.1. Czym jest model LLM?
- 1.2. Zastosowania modeli LLM
- 1.3. Etapy tworzenia modeli LLM i korzystania z nich
- 1.4. Wprowadzenie do architektury transformerów
- 1.5. Wykorzystanie dużych zbiorów danych
- 1.6. Szczegóły architektury modeli GPT
- 1.7. Tworzenie dużego modelu językowego
- Podsumowanie

2. Praca z danymi tekstowymi

- 2.1. Czym są osadzenia słów?
- 2.2. Tokenizacja tekstu
- 2.3. Konwersja tokenów na identyfikatory
- 2.4. Dodawanie specjalnych tokenów kontekstowych
- 2.5. Kodowanie par bajtów
- 2.6. Próbkowanie danych z oknem przesunym
- 2.7. Tworzenie osadzeń tokenów
- 2.8. Kodowanie pozycji słów
- Podsumowanie

3. Kodowanie mechanizmów uwagi

- 3.1. Problem z modelowaniem długich sekwencji
- 3.2. Przechwytywanie zależności między danymi za pomocą mechanizmów uwagi
- 3.3. Zwracanie uwagi na różne części danych wejściowych przez mechanizm samouwagi
 - 3.3.1. Prosty mechanizm samouwagi bez trenowalnych wag
 - 3.3.2. Obliczanie wag uwagi dla wszystkich tokenów wejściowych
- 3.4. Implementacja mechanizmu samouwagi z trenowalnymi wagami
 - 3.4.1. Obliczanie wag uwagi krok po kroku
 - 3.4.2. Implementacja kompaktowej klasy samouwagi w Pythonie
- 3.5. Ukrywanie przyszłych słów dzięki zastosowaniu uwagi przyczynowej
 - 3.5.1. Wykorzystanie maski uwagi przyczynowej
 - 3.5.2. Maskowanie dodatkowych wag uwagi z użyciem dropoutu
 - 3.5.3. Implementacja zwężonej klasy przyczynowej uwagi
- 3.6. Rozszerzenie uwagi jednogłowicowej na wielogłowicową
 - 3.6.1. Utworzenie stosu wielu jednogłowicowych warstw uwagi

- 3.6.2. Implementacja uwagi wielogłowicowej z podziałem wag
- Podsumowanie

4. Implementacja od podstaw modelu GPT do generowania tekstu

- 4.1. Kodowanie architektury LLM
- 4.2. Normalizacja warstwowa aktywacji
- 4.3. Implementacja sieci ze sprzężeniem w przód z aktywacjami GELU
- 4.4. Dodawanie połączeń skrótowych
- 4.5. Łączenie warstw uwagi i warstw liniowych w bloku transformera
- 4.6. Kodowanie modelu GPT
- 4.7. Generowanie tekstu
- Podsumowanie

5. Wstępne szkolenie na nieoznakowanych danych

- 5.1. Ocena generatywnych modeli tekstowych
 - 5.1.1. Używanie modelu GPT do generowania tekstu
 - 5.1.2. Obliczanie strat związanych z generowaniem tekstu
 - 5.1.3. Obliczanie strat w zestawie szkoleniowym i walidacyjnym
- 5.2. Szkolenie modelu LLM
- 5.3. Strategie dekodowania w celu zarządzania losowością
 - 5.3.1. Skalowanie temperaturą
 - 5.3.2. Próbkowanie top-k
 - 5.3.3. Modyfikacja funkcji generowania tekstu
- 5.4. Wczytywanie i zapisywanie wag modeli z użyciem frameworka PyTorch
- 5.5. Ładowanie wstępnie przeszkolonych wag z modelu OpenAI
- Podsumowanie

6. Dostrajanie modelu LLM do zadań klasyfikacji

- 6.1. Różne kategorie dostrajania
- 6.2. Przygotowanie zbioru danych
- 6.3. Tworzenie mechanizmów ładujących dane
- 6.4. Inicjalizacja modelu z użyciem wag wstępnie przeszkolonego modelu
- 6.5. Dodawanie nagłówka klasyfikacji
- 6.6. Obliczanie straty i dokładności klasyfikacji
- 6.7. Dostrajanie modelu na danych nadzorowanych
- 6.8. Wykorzystanie modelu LLM jako klasyfikatora spamu
- Podsumowanie

7. Dostrajanie modelu LLM do zadań wykonywania instrukcji

- 7.1. Wprowadzenie do dostrajania do wykonywania instrukcji
- 7.2. Przygotowanie zbioru danych do nadzorowanego dostrajania pod kątem wykonywania instrukcji
- 7.3. Organizowanie danych w partie szkoleniowe
- 7.4. Tworzenie mechanizmów ładujących dane dla zbioru danych instrukcji
- 7.5. Ładowanie wstępnie przeszkolonego modelu LLM
- 7.6. Dostrajanie modeli LLM do zadań wykonywania instrukcji

- 7.7. Wyodrębnianie i zapisywanie odpowiedzi
- 7.8. Ocena dostrojonego modelu LLM
- 7.9. Wnioski
 - 7.9.1. Co dalej?
 - 7.9.2. Bądź na bieżąco w szybko zmieniającej się dziedzinie
 - 7.9.3. Na koniec
- Podsumowanie

Dodatek A Wprowadzenie w tematykę frameworka PyTorch

Dodatek B Bibliografia i lektura uzupełniająca

Dodatek C Rozwiązania ćwiczeń

Dodatek D Usprawnianie pętli szkoleniowej

Dodatek E Skuteczne dostrajanie parametrów za pomocą LoRA