

Spis treści

O autorach	11
O korektorze merytorycznym	12
Przedmowa	13
Wprowadzenie	17
ROZDZIAŁ 1	
Architektura i koncepcja projektu LLM Twin	23
Koncepcja kryjąca się za aplikacją LLM Twin	24
Czym jest projekt LLM Twin?	24
Dlaczego sensowne jest tworzenie projektu LLM Twin?	25
Dlaczego w omawianym celu nie można użyć ChatGPT (lub podobnego chatbota)?	27
Planowanie produktu o minimalnej niezbędnej funkcjonalności dla projektu LLM Twin	28
Czym jest produkt o minimalnej niezbędnej funkcjonalności?	29
Zdefiniowanie produktu o minimalnej niezbędnej funkcjonalności w projekcie LLM Twin	29
Budowanie systemu uczenia maszynowego z wykorzystaniem potoków cech, trenowania i wnioskowania	31
Aspekty, jakie należy uwzględnić podczas budowania systemów uczenia maszynowego	31
Problem z poprzednimi rozwiązaniami	32
Rozwiązanie — potoki uczenia maszynowego dla systemów uczenia maszynowego	35
Zalety architektury FTI	37
Opracowanie architektury systemu dla projektu LLM Twin	39
Szczegóły techniczne dotyczące infrastruktury projektu LLM Twin	39
Jak opracować architekturę LLM Twin za pomocą projektu opartego na potoku FTI?	40
Kilka uwag końcowych na temat projektu FTI i architektury LLM Twin ...	46

Podsumowanie	47
Źródła	47
ROZDZIAŁ 2	
Narzędzia i ich instalacja	49
Ekosystem Pythona i przygotowanie projektu	50
Poetry — menedżer zależności i środowisk wirtualnych	51
Poe the Poet — narzędzie do wykonywania zadań	53
Narzędzia MLOps i LLMOps	54
Hugging Face — rejestr modelu	54
ZenML — oprogramowanie koordynujące, artefakty i metadane	56
Comet — oprogramowanie do śledzenia eksperymentu	66
Opik — monitorowanie promptu	68
Bazy danych do przechowywania danych niestrukturyzowanych i wektorowych	68
MongoDB — baza danych typu NoSQL	69
Qdrant — wektorowa baza danych	69
Przygotowanie do użycia chmury AWS	70
Utworzenie konta AWS i klucza dostępu oraz przygotowanie narzędzia powłoki do pracy z usługą AWS	70
SageMaker — obliczenia związane z trenowaniem i wnioskowaniem	72
Dlaczego SageMaker?	72
Podsumowanie	74
Źródła	74
ROZDZIAŁ 3	
Inżynieria danych	76
Opracowanie potoku pobierania danych do projektu LLM Twin	77
Implementacja potoku pobierania danych do projektu LLM Twin	81
Potok ZenML i kroki	81
Dyspozytor — jak zainicjalizować odpowiedni crawler?	85
Crawlery	87
Dokumenty hurtowni danych typu NoSQL	96

Umieszczanie nieprzetworzonych danych bezpośrednio w hurtowni danych	104
Rozwiązywanie problemów	108
Podsumowanie	109
Źródła	109
ROZDZIAŁ 4	
Potok wykorzystujący technikę RAG	111
Wyjaśnienie techniki RAG	112
Dlaczego warto używać techniki RAG?	112
Zwykły framework systemu RAG	114
Czym są osadzenia?	118
Więcej informacji na temat wektorowych baz danych	125
Ogólne omówienie zaawansowanej techniki RAG	127
Przed pobraniem danych	129
Pobieranie danych	132
Po pobraniu danych	134
Prezentacja architektury techniki RAG wykorzystanej w projekcie LLM Twin ...	136
Problem, który chcemy rozwiązać	136
Magazyn danych cech	137
Skąd pochodzą nieprzetworzone dane?	138
Opracowanie architektury potoku cech techniki RAG	138
Implementacja potoku techniki RAG w projekcie LLM Twin	148
Klasa Settings	148
Potok ZenML i kroki	149
Encje dziedziny Pydantic	157
Warstwa dyspozytora	164
Procedury obsługi	167
Podsumowanie	175
Źródła	176
ROZDZIAŁ 5	
Nadzorowane dostrajanie modelu	177
Tworzenie wysokiej jakości zbioru danych instrukcji	178

Ogólny framework rozwiązania	178
Gromadzenie danych	181
Filtrowanie oparte na regułach	182
Eliminacja duplikatów	183
Dekontaminacja danych	185
Ocena jakości danych	186
Eksploracja danych	188
Generowanie danych	190
Uzupełnienie danych	192
Tworzenie własnego zbioru danych instrukcji	194
Nadzorowane dostrajanie modelu i związanych z nim technik	202
Kiedy należy dostrajać model?	203
Formaty zbiorów danych instrukcji	204
Szablony czatu	205
Techniki Parameter-Efficient Fine-Tuning (PEFT)	206
Trenowanie parametrów	212
Dostrajanie w praktyce	215
Podsumowanie	222
Źródła	222
ROZDZIAŁ 6	
Dostrajanie modelu z uwzględnieniem preferencji użytkowników	224
Poznanie zbiorów danych preferencji	225
Dane preferencji	225
Generowanie danych i ich ocena	228
Samodzielne tworzenie zbioru danych preferencji	233
Uwzględnienie preferencji	239
Uczenie przez wzmocnienie na podstawie opinii użytkowników	239
Bezpośrednia optymalizacja preferencji	241
Implementacja bezpośredniej optymalizacji preferencji	244
Podsumowanie	250
Źródła	251

ROZDZIAŁ 7

Ocena dużych modeli językowych	252
Ocena modelu	252
Porównanie oceny uczenia maszynowego i oceny dużego modelu językowego	253
Ocena dużego modelu językowego ogólnego przeznaczenia	254
Ocena dużego modelu językowego związanego z dziedziną	256
Ocena dużego modelu językowego związanego z zadaniem	258
Ocena systemu RAG	261
Ragas	262
ARES	264
Ocena modelu TwinLlama-3.1-8B	265
Generowanie odpowiedzi	266
Ocena odpowiedzi	268
Analiza wyników	271
Podsumowanie	275
Źródła	276

ROZDZIAŁ 8

Optymalizacja wnioskowania	277
Strategie optymalizacji modelu	278
Bufor KV	279
Przetwarzanie ciągłymi partiami	281
Dekodowanie spekulatywne	282
Zoptymalizowane mechanizmy uwagi	284
Równoległość modelu	285
Równoległość danych	286
Równoległość potoku	287
Równoległość tensora	288
Łączenie różnych technik	289
Kwantyzacja modelu	290
Wprowadzenie do kwantyzacji	291

Kwantyzacja za pomocą GGUF i llama.cpp	295
Kwantyzacja za pomocą GPTQ i EXL2	297
Inne techniki kwantyzacji	298
Podsumowanie	299
Źródła	300
ROZDZIAŁ 9	
Potok wnioskowania wykorzystujący technikę RAG	301
Potok wnioskowania RAG w modelu Twin	302
Zaawansowane techniki RAG w modelu Twin	304
Zaawansowane techniki optymalizacji etapu przed pobieraniem danych w systemie RAG — rozbudowa zapytania i samozapytanie	307
Zaawansowane techniki optymalizacji etapu pobierania danych w systemie RAG — filtrowane wyszukiwanie wektorowe	314
Zaawansowane techniki optymalizacji etapu po pobieraniu danych w systemie RAG — ponowne przygotowanie rankingu	315
Implementacja potoku wnioskowania RAG w modelu Twin	319
Implementacja modułu pobierania danych	319
Połączenie wszystkiego w całość w potoku wnioskowania w systemie RAG	325
Podsumowanie	330
Źródła	331