Vector Database – wyspecjalizowane bazki zaprojektowane do store’owania i manipulacji wysoko-przestrzennymi wektorami danych. 80% danych jest nieustrukturayryzowane i nie pasuje do kontkestu relacyjnej bazy danych. Wektorowe bazy danych pomagają rozwiazywac problem z efektywnym store’owaniem danych i indeksowaniem by wykonywać query szybko.

Unstructured data są przetwarzane na początku w procesie vector embeddingu. Przykładem takiej danej mogą być zdjęcia, co jest 2D RGB obiektem, video, audio. Takie dane trafiają do embedding modelu, w wyniku czego otrzymujemy arraykę z liczbami, co jest vector embed. Dla przykłądu procesem vector emedingu będzie zakdowanie słowa „HELLO WORLD” „HELLO MOSHI” takie dwa słowa są wrzucane do embeddingującego algorytmu, który w wyniku przetwarzania daje arrayke np. [ 1 1 0 ] albo [1 0 1] gdzie [1 – Hello 1 – Moshin 1 – World] jak wartość 0 to znaczy ze słowa brak. Obraz zawierający tekst, zrzut ekranu, diagram, Czcionka

Opis wygenerowany automatycznie

Inny przykład z owockami/ warzywami/ kwiatami. Kategorie mniej wiecej grupia się w tym samym obszarze. Clustrowanie i Classyfikacja to co cos co się wykonuje w procesie czyli określenie labelek, klas danych grup jak np. ten przykład z owocami warzywami.

Obraz zawierający tekst, diagram, linia, Czcionka

Opis wygenerowany automatycznie

Można wyliczać odleglosci miedzy takimi grupami stosując rozne algorytmy co pozwala okreslic odleglosc miedzy punktami.

Obraz zawierający tekst, diagram, linia, Czcionka

Opis wygenerowany automatycznie

W procesie indexingu te same kategorie są ze sobą grupowane.

Przykłady vector database: Pinecone, Milvus, ScaNN, Weaviate, Vespa, FAISS, Chroma, Redis, Annoy, PgVector.

USE case’y:  
LLM – polegaja na dataset na kotrym były trenowane. Mogą nie posiada aktualnych wiaodmosci, albo specyficznej wiedzy. Vector database może pomoc rozszerzyć możliwości, albo uaktualnić infomracje.  
Semantic search - wyszukując koncpetu jak picture of waving nie chodzi o słowa kluczowe, ale o abstrakcje czynności, dla tego interpolacja wynikow jest bardziej oczekiwana do czego nadaja się wektorowe bazki. Porownujac ze sobą wkeotry, czyli wykonując vector embedding i obliczając podobniestwo miedzy wektorami można wyciagnac interesujące nas rezultaty.  
Personalizaowane rekomendacje – DLRM by Facebook/ Neural Collaborative Filtering (NCF) / Variational Auto-encoder Collaborative Filtering (VAE-CF). W takim przypadku tez sie nada bo bazujac na rekomendacji szuka odpowiedniej zwrotki.  
Wykrywanie anomalii – gdy cos znajduje się w zasięgu malicious zachowan.

Początek formularza

**What problem does a vector database address in large language models (LLMs)?**

**ANSWER: Memory issues**

Dół formularza

# Chroma

Bazda danych wektoroa działająca in memeory, embedding database, open source’owa. Sluzy do budowania LLM appek poprzez dostarczanie wiedzy, faktow, umiejetnosci do LLMa.

# Pinecone

W free tierze można jeden index utworzyć

Tworzac go definujemy coś takiego jak pods. Podsy to preconfigurowalne jednostki sprzętowe do działania Pinecone. Każdy index działa na takich podsach jednym lub więcej. Generalnie im więcej podsow tym większe stroage capaicty, lower latency, higher througput. Sa jednak typy podow które nie można zmienić po utowrzeniu indexu, co prawda można stworzyć kolekcje z nowym podtypem na podstawie już istneijacej kolekcji.

Pod type:  
Starter plan- można stowrzyc jeden pod z zasobami wspierającymi 100\_000 wektorow z 1536 wymiarowym embeddingiem i metadanymi. Capacity jest propercjonalne dla innych wymiarow.  
S1 pod – storage optmized pod który dostarczę duże capacity i lower costs, przy czym ma highger query latencies niż pod P1. Dobry dla dużych indeksow z średnim dopuszczalnym czasem opoznienim. Each S1 pod ma możliwość pojemność około 5M vectorow w 768 wymiarach.  
P1 - performance-opimized, dostarcza niskiej opóźnienia dla query, ale przechowuje mniej wektorow niż S1 pod. Dobyr dla appaek wymagających latency mniejszego niż 100ms. Każdy P1 ma możliwość pojemności około 1M wektorow z 768 wymiarach  
P2 pod -posiada lepszy throughput z niska latency. Dla wektorow o wymiarach mniejszych niż 128, p2 oblsuguej 200 QPS na replikę i zwraca zapytania w mniej niż 10 ms. Ma lepszą przepustowość niż s1 i p1. Każdy P2 ma dość zasboow by pomiesic około 1M vectorow z 768 wymiarami.

# Operacje na Vectorowej DB

Create -> wstawanienie nowego wektora, np. embedding new img do wartości wetkora, i wstawaienie wetkora do db.

Read -> poszukiwania podobnych obrazów (google lens)

Update – updating tekstowy document wektory z wieksza iloscia contentu, na podstawie tego co wyciagnał z kontekstu reada.

Deletee – wyszukanie w kotenkscie oepracji read podobnych np. image vectorow do zadanego i usuniecie ich.

# Index, collection, vector partitioning, metrics

Index – to nazwa kolekcji vectorow. Vector może mieć kilka wymairow w swojej przestrzeni. Ja torouzmiem tak ze jak wektor jest 3 wymiarowy to posiada 3 wartosciktore go charakteryzują. Collection – to statyczna kopia indkesu, nie queryowalna, uzyteczna do backupu indeksów

Vector partitioning – partycje bd utrzymywane w innych przestrzeniach dyskowych, daje scailability, speed, load balancing, fault tolerance. Tworzac partycje można poprzez namespace, separate index, metadata.

Obraz zawierający tekst, diagram, zrzut ekranu, design

Opis wygenerowany automatycznie

Obraz zawierający tekst, diagram, zrzut ekranu, design

Opis wygenerowany automatycznieObraz zawierający tekst, diagram, zrzut ekranu, linia

Opis wygenerowany automatycznie

Namespace – index może mieć 1 lub więcej nemsapce, każdy namespace musi mieć unikatowa nazwe w indexie, każdy wektor może być przypisany tylko do jednego namespacea, domyslna nazwa w pineconie dla indeksu to „”.

Metrics – sposób kalkulacji miedzy vectorami.   
Euklidiansa odleglosc -szuka najmniejszego dystnasu, do query vectora, wartości to cokolwiek miedzy 0 a Inf  
Cosine – 1 - kąt między wektorami, jego wartości to od -1 do 1. Mierzy odleglosc kątową. Czyli gdy wetkory takie same to cos 0 = 1   
Dot Product – not normalized, wielkość brana pod uwagę. Z tego co rozumiem wieksza wartość dot produktu tym odleglosc blizsza.

Obraz zawierający linia, tekst, diagram, Czcionka

Opis wygenerowany automatycznie

# Semantic Search

Sematnic search to poszukiwanie slow kluczowych na podstawie synonimiow/ powiązanych terminow – zwraca zwykle bardziej dokładne i odpowiedajace informacje np. query „movie about space exploration”, tradycyjny search poszuka „space” i „explorantion”. Schematic poszuka raczej „space exploration”

Działa to na zasadzie obudowanie/ embedingu wszystkich wejść w wektory, konwersje search’a w vecotry. Query leci do DB i znajduje najbliższych sasiadow którzy interpretowani są jako resulty.

Benefity – bardziej precyzyjny wynik, lepsze zrozumienie kontkestu, wzbogacenie personalizowanych rekomendacji, oszczędność czasowa i wydajny zwrot informacji, natural language processing.

Implementacja:   
1. Zebranie danych  
2. NLP/ NLU (tokenizacja/ NER/ POS)  
3. Semantic Model (embeding transfoemrs) -> Word Embeddings (word2vec/ glove/ elmo/ fasttext) || transfomers (BERT/ Roberta/ GPT/ Trasnformer-XL/ XLNet)  
4. Indexing Vector Database

# Named entity Regonition

Process identyfikacji I kategoryzacji entities w text. Identyfikujący prawdziwe obiekty/ osoby/ organizacje/ lokacje/ daty. Wyciaga meaningful informacje z tekstu rozpoznając i klasyfikujące te entities np. mmay tekst ze ktoś w jakim roku cos tam zrobil w ramach czegoś – to wyciągnie informacje kto, kiedy, gdzie,z czym itd. Probuje wyciagnac informacje z tekstu.

Implementacja:

Obraz zawierający tekst, Czcionka, zrzut ekranu, linia

Opis wygenerowany automatycznie

Text preprocessing – oczyszczenie tekstu, spacje, znaki specjalne, cos czego nie potrzebuje.  
Part of speech (POS) Tagging - przechodzi przez speeach engine który wyciąga informacje o elementach.   
Named enitty Recognition – chyba poszukuje dopasowuej do odpowiedniej odmiany słowa np. rzeczownika.  
Entity Classificiton / Post processing – podział danych na kategorie.