Historia Sztucznej Inteligencji

Od systemów eksperckich do ery modeli generatywnych (GenAI) — raport badawczy dla inżynierów oprogramowania wkraczających na rynek w latach 2025–2026.

Dlaczego to ważne?

AI jako wymóg, nie opcja

Nowa rola inżyniera

Czyste "pisanie kodu" staje się zautomatyzowane. Współczesny programista to **architekt systemów** i weryfikator logiki biznesowej — tzw. **AI-Assisted Developer**.

Skala adopcji (2025)

- GitHub Copilot: **20 mln użytkowników**, wzrost 400% r/r
- **~90% firm Fortune 100** korzysta z asystentów AI
- **46%** nowego kodu generowane przez AI
- W Javie wskaźnik ten wynosi **61%**, w Pythonie **55%**

Wpływ AI na produktywność

52%

Potwierdzony wzrost

programistów potwierdza pozytywny wpływ AI na wydajność (Stack Overflow 2025)

55%

Szybsze zadania

programiści z GenAI kończą zadania szybciej w kontrolowanych badaniach

75%

Krótszy code review

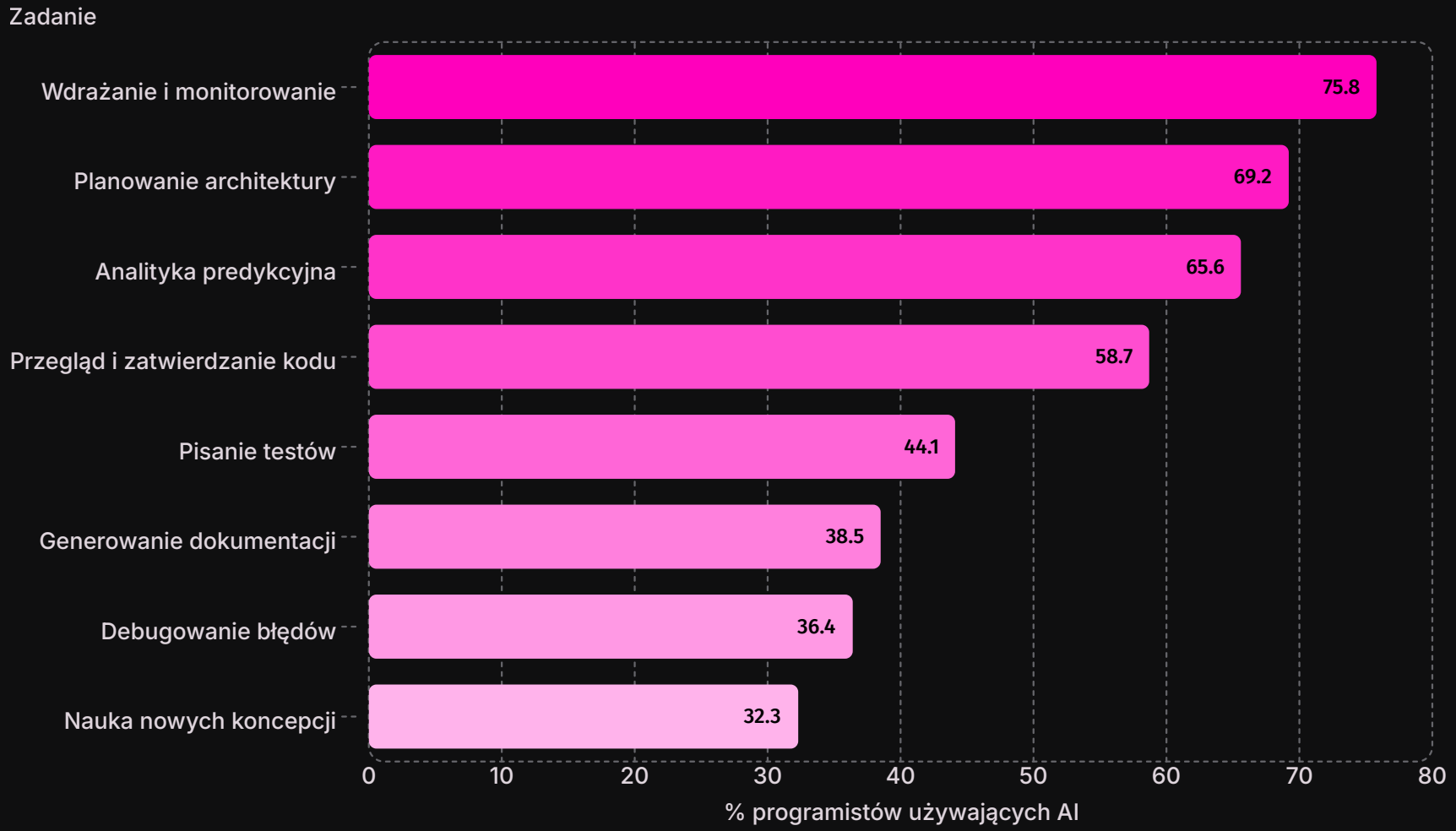
czas przeglądu kodu spada z 9,6 do 2,4 dnia

817K

Stanowisk w Polsce

wysoce podatnych na interakcję z systemami generatywnymi (raport NASK)

Wykorzystanie AI według kategorii zadań (2025)



Dane z raportu Stack Overflow 2025. Najwyższy wskaźnik adopcji notują zadania operacyjne i architektoniczne — AI staje się standardem w całym cyklu wytwarzania oprogramowania.

ROZDZIAŁ 1

Prehistoria i narodziny AI

1

1950 — Alan Turing

Esej *Computing Machinery and Intelligence*. Propozycja "Gry w Naśladownictwo" — Testu Turinga jako benchmarku oceny maszyny.

2

1955–56 — Logic Theorist

Newell i Simon tworzą program dowodzący 38 twierdzeń z *Principia Mathematica*. Jeden dowód — krótszy niż autorski.

3

1956 — Dartmouth

McCarthy, Minsky, Shannon. Oficjalne narodziny dyscypliny "Sztuczna Inteligencja". Wielki optymizm — pełna inteligencja w kilka tygodni?

4

1964–67 — ELIZA

Weizenbaum (MIT) tworzy pierwszy chatbot. Dopasowanie wzorców regex wywołuje "Efekt ELIZY" — użytkownicy zwierzają się maszynie.

ELIZA i Test Turinga — dwie lekcje pokory

Test Turinga (1950)

Sędzia rozmawia z maszyną i człowiekiem przez terminal. Jeśli nie odróżni — maszyna jest "inteligentna". Współczesne LLM zdają ten test z łatwością, **nie posiadając świadomości** — naśladownictwo jest algorytmicznie prostsze niż ogólna racjonalność.

ELIZA (1966)

Architektura oparta w **100% na dopasowywaniu wzorców**. Wzorzec `I feel X` → odpowiedź `Why do you feel X?`. Brak rozumienia słów. Mimo to użytkownicy zwierzali się maszynie z intymnych szczegółów życia — **Efekt ELIZY**.

Pierwsza Zima AI

Eksplozja kombinatoryczna zatrzymała wszystko

Gdy algorytmy wyszły poza zamknięte środowiska logiczne, napotkały na **eksplozję kombinatoryczną** — liczba stanów decyzyjnych rośnie szybciej niż wykładniczo, a sprzęt lat 70. dysponował śladową pamięcią RAM. Raport Lighthilla (1973) brutalnie podsumował niemożliwość skalowania poza laboratorium. Sponsorzy odcięli finansowanie — nastała **Pierwsza Zima AI**.

ROZDZIAŁ 2

Era systemów eksperckich

Baza Wiedzy

Tysiące faktów i reguł `JEŚLI A i B → TO C`, ręcznie kodowanych przez programistów na podstawie wywiadów z ekspertami.

Silnik Wnioskowania

Moduł procesujący dynamicznie dobierający łańcuchy reguł do danych wejściowych użytkownika, niezależny od bazy wiedzy — można aktualizować reguły bez psucia logiki silnika.

Lata 80. przyniosły odrzucenie mrzonek o "myślącej maszynie" na rzecz wyspecjalizowanych narzędzi komercyjnych. Systemy eksperckie miały jedno zadanie: emulować eksperta w **bardzo wąskiej dziedzinie**.

DENDRAL i MYCIN — triumfy i bariery

DENDRAL (Stanford, lata 60.)

Analiza wyników spektrometrii mas. Dedukował strukturę cząsteczek organicznych z ogromną dokładnością — wyniki wielokrotnie przewyższające początkujących badaczy w tej wąskiej niszy.

MYCIN (Stanford, lata 70.)

Diagnostyka bakteryjnych infekcji krwi i dobór dawek antybiotyków. W ślepych testach trafniejszy niż specjaliści. Reguła: *"JEŚLI barwienie Grama ujemne ORAZ pałeczka ORAZ gorączka → zakażenie jelitowe (pewność 0.8)"*.

Dlaczego MYCIN nie trafił do szpitali?

Nie kod był problemem — lecz **kwestie prawne i etyczne**. Kto odpowiada za śmierć pacjenta z winy algorytmu? Lekarze nie ufali komputerom na tyle, by powierzyć im ludzkie życie.

Druga Zima AI (lata 90.)

Systemy regułowe uderzyły w mur: **niezdolność do nauki**, problem kruchości (brittleness), ogromne koszty utrzymania bazy wiedzy. Ręczne kodowanie złożoności świata okazało się architektonicznym absurdem.

ROZDZIAŁ 3

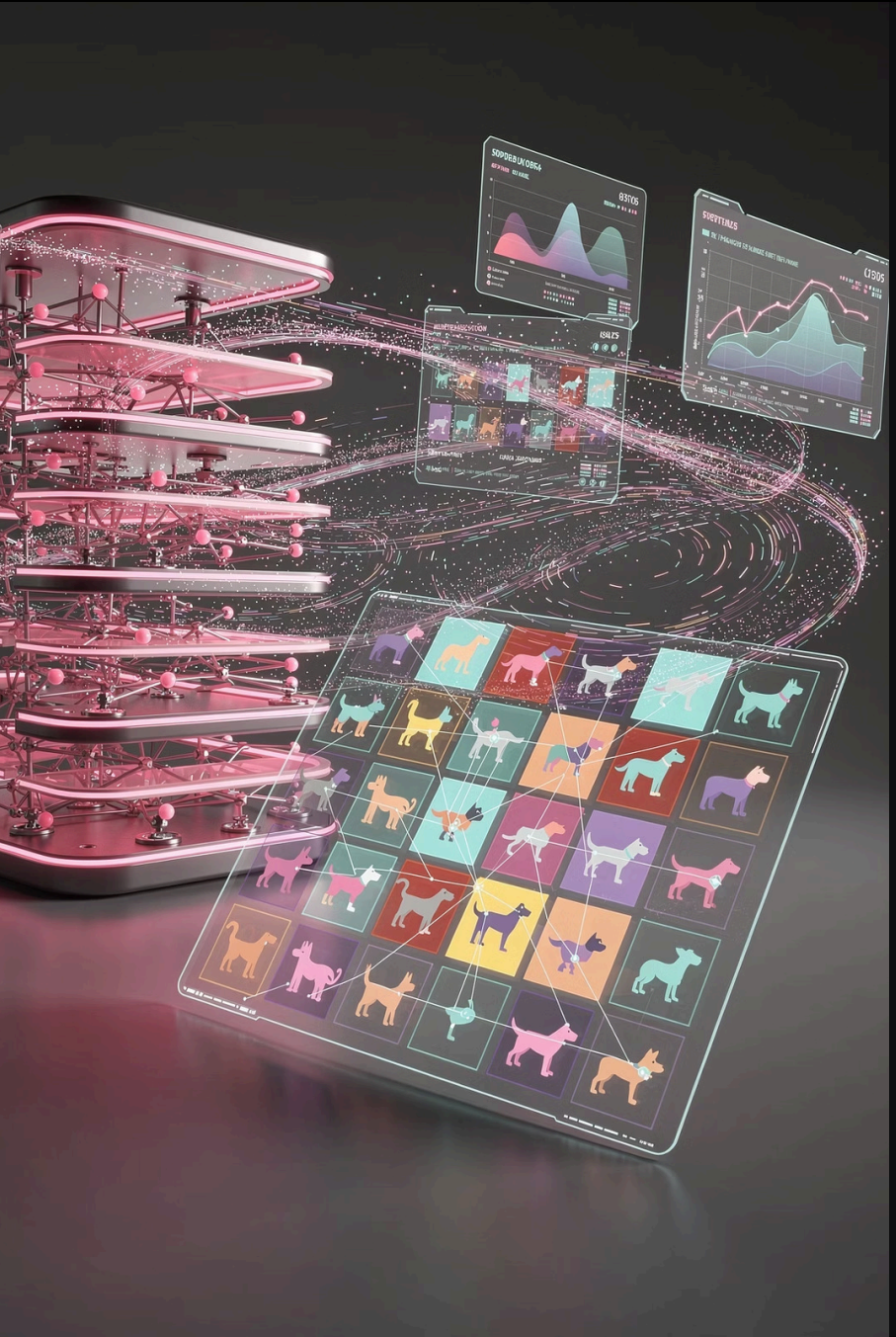
Rewolucja Big Data i uczenie maszynowe

Odwrócona logika programowania

W tradycyjnym paradygmacie: **reguły + dane → wynik**. W uczeniu maszynowym: **dane + poprawne odpowiedzi → reguły odkryte przez model**. Przejście od z góry ustalonych poleceń do adaptacyjnych macierzy prawdopodobieństwa.

Dwa katalizatory

- **Big Data:** internet, social media, IoT — niewyobrażalne ilości nieustrukturyzowanych danych jako paliwo treningowe
- **GPU:** tysiące prostych rdzeni działających równolegle — trening sieci neuronowych przyspieszył **tysiąckrotnie**



AlexNet 2012 — przebudzenie głębokiego uczenia

Konkurs ImageNet

30 września 2012. Klasyfikacja setek tysięcy zdjęć na 1000 kategorii. AlexNet (Hinton, Toronto) zmiażdżył konkurencję — błąd niższy o **9,8 p.p.** od drugiego miejsca.

Inżynieryjny majstersztyk

Trening rozdzielony na **dwie karty NVIDIA GTX 580**. Funkcja aktywacji **ReLU** rozwiązała problem "zanikającego gradientu" — umożliwiając trenowanie głębokich, wielowarstwowych sieci.

Hierarchia warstw

Pierwsza warstwa: linie brzegowe. Kolejna: tekstury i kształty. Ostatnia: predykcja końcowa (np. "Siberian Husky"). Gwałtowny boom na wielowarstwowe sieci neuronowe.

AlphaGo vs Lee Sedol — Rubikon przekroczony (2016)

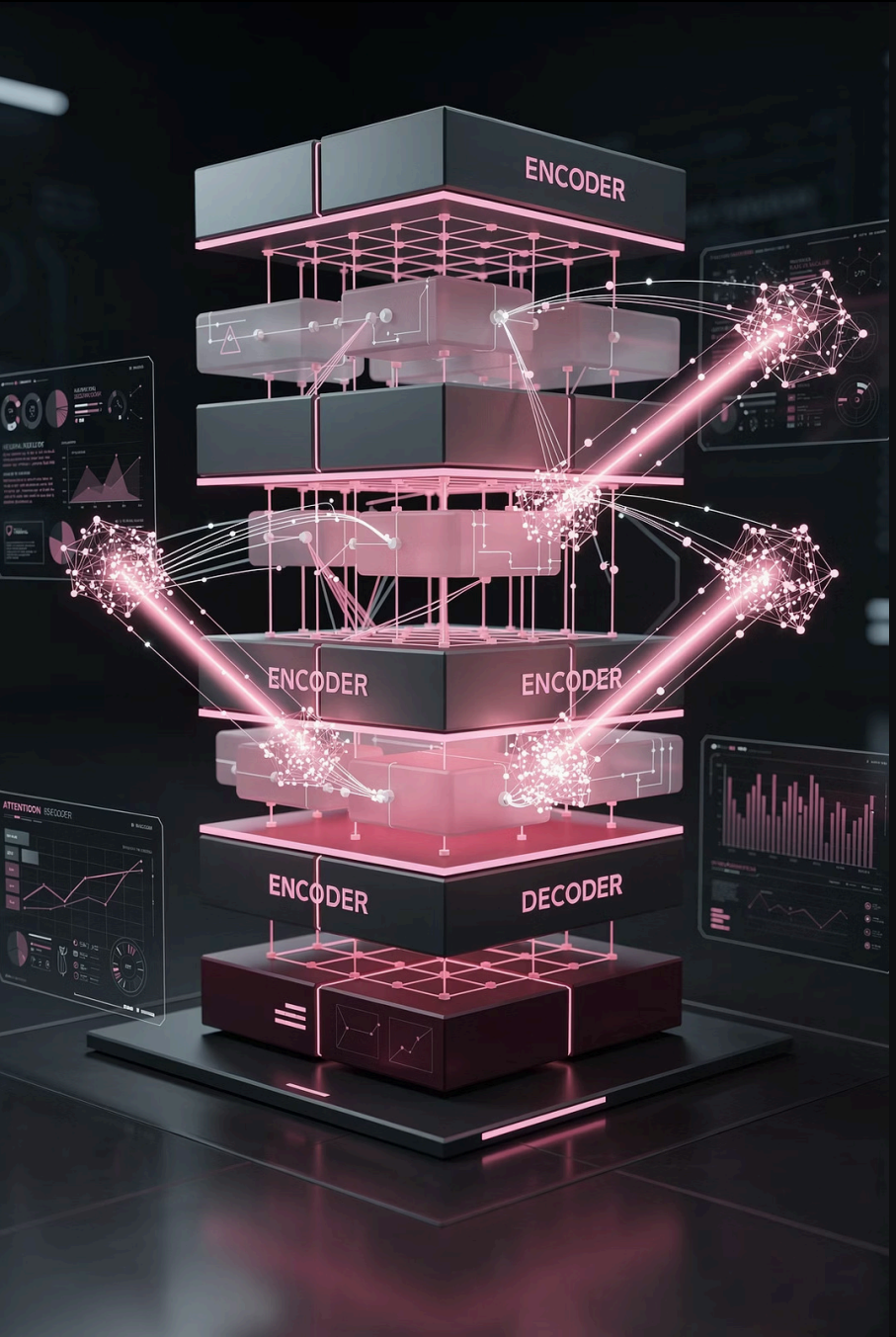
Dlaczego Go było "Świętym Graalem"?

Liczba kombinacji ruchów przewyższa liczbę atomów we Wszechświecie. Metoda brute-force (jak Deep Blue w szachach) była bezużyteczna. DeepMind połączył **Monte Carlo + głębokie sieci neuronowe + uczenie przez wzmacnianie**. Wynik: **4–1** dla AlphaGo.

Dwa ikoniczne ruchy

Ruch 37 (gra 2): AlphaGo zagrało nielogicznie — komentatorzy uznali to za błąd. Po kilkudziesięciu ruchach okazało się, że maszyna zaprojektowała długofalową kontrolę terytorium. Pierwsza "kreatywna intuicja" maszyny.

Ruch 78 (gra 4): Lee Sedol wykonał desperacki, statystycznie rzadki ruch — "Boskie Dotknięcie". AlphaGo kompletnie się pogubiło i przegrało. Dowód elastyczności nieustrukturyzowanego umysłu ludzkiego.



ROZDZIAŁ 4

Rewolucja Transformer — "Attention Is All You Need" (2017)

Do 2017 roku dominowały rekurencyjne sieci RNN/LSTM — czytały tekst **sekwencyjnie**, tracąc kontekst z początku długich dokumentów. Artykuł Google Brain całkowicie to zmienił.

Query (Q)

Co dany token próbuje wyszukać względem innych w kontekście.

Key (K)

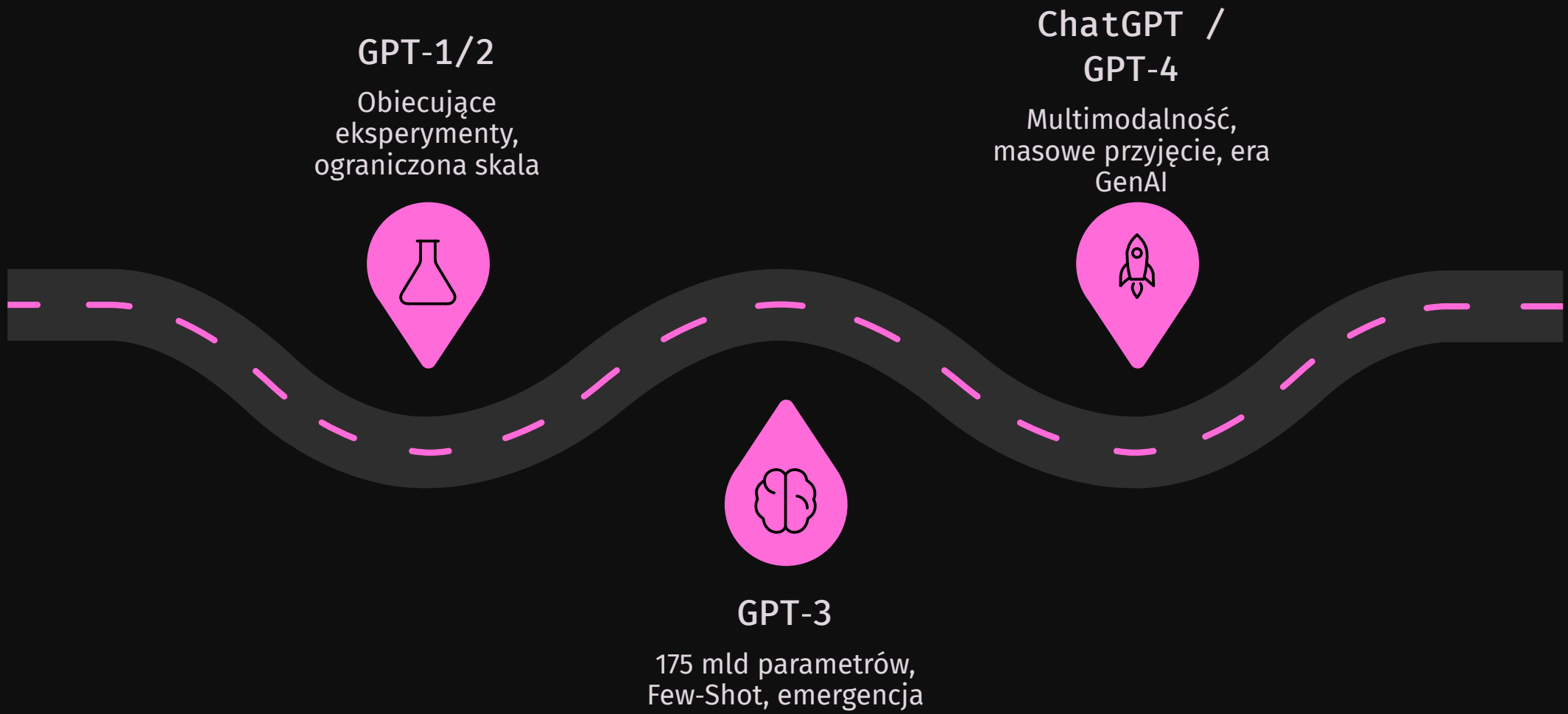
Cechy informujące resztę systemu: "oto co dokładnie zawiera ten element".

Value (V)

Prawdziwy wektor wagi pojęcia przekazywanego w procesie końcowym.

Dzięki QKV zdanie "Zamek zaciął się w dżinsach" jest w 100% obojętne na podwójne znaczenie słowa "zamek" — system powiązał wysoką wagą słowo "dżinsy". Artykuł osiągnął do 2025 roku **~173 tysiące cytowań**.

Od GPT-3 do ChatGPT — narodziny LLM

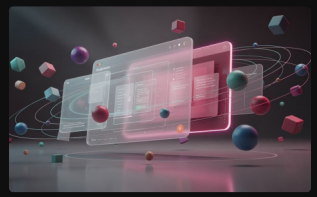


OpenAI udowodniło w latach 2018–2020, że inteligencja modelu rośnie **wykładniczo z liczbą parametrów**. GPT-3 (175 mld parametrów) ujawnił zjawisko **inteligencji emergentnej** — uczenia małopróbkowego (Few-Shot Learning): maszyna dedukowała abstrakcyjne zjawiska z kilku przykładów bez kosztownego fine-tuningu. GPT-4 dodał multimodalność, a ChatGPT wywołał erupcję mainstreamu GenAI w 2022 roku.

Krajobraz modeli AI (2025–2026)

Claude (Anthropic)

Wersje 3.5 i 4.5 Sonnet. Dominuje w benchmarkach logiki programistycznej, szczególnie frontend z nowoczesnymi frameworkami. Precyzyjnie trzyma kierunek zadania analitycznego.



Gemini (Google)

Potężne okno kontekstowe — Gemini 1.5 Pro pochłania tysiące stron lub rozległe drzewa kodu naraz, odpowiadając o spójność architektury bez gubienia zmiennych głęboko w podkatalogach.

Mistral / LLaMA (Meta)

Rewolucja wolnych licencji i **Mixture-of-Experts (MoE)** — aktywuje tylko dedykowany wycinek parametrów (np. 41 mld z 675 mld). Drastycznie obniża koszty API przy zachowaniu wysokiej celności.

Modele dyfuzyjne – generowanie obrazów

Jak działają Diffusion Models?

Forward diffusion: obraz niszczony piksel po pikselu do "losowej zupy szumu" (Gaussian noise).

Reverse diffusion (Denoising): sieć U-Net uczy się idealnie przewidywać usunięcie szumu, warunkowana tekstowym wektorem (Cross-Attention).

Efekt: wyczarowanie hiperrealistycznego portretu z szumu przez samo żądanie tekstowe — np. *"astronauta na koralach"*. Komercyjnie: DALL-E, Midjourney.

ROZDZIAŁ 5

Narzędzia AI-Assisted Developera (2025–2026)

GitHub Copilot (Microsoft)

Standard korporacyjny zintegrowany z repozytoriami. Wbudowany w Visual Studio — przewiduje obszerne bloki kodu boilerplate, rozwiązuje powtarzalne problemy bez ręcznej interwencji. **20 mln użytkowników**.

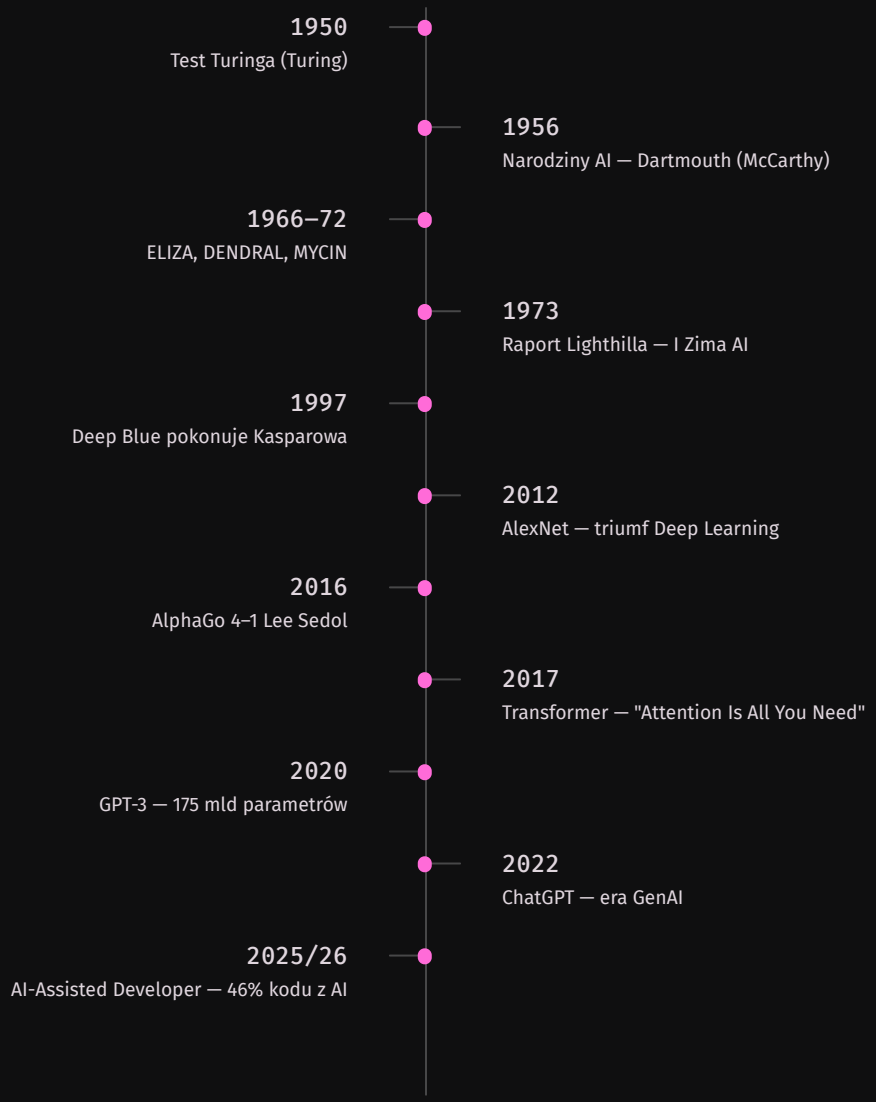
Cursor AI

Król zaawansowanych edytorów, bazujący na VS Code. Jeden rozkaz `CMD+K` tworzy spójną usługę z testami jednostkowymi. Dynamiczne zmiany symultanicznie na kilkudziesięciu plikach jednocześnie.

Windsurf (Codeium)

Faworyt deweloperów z otwartymi planami komercyjnymi. Wybitny w projektach wymagających **żelaznej pamięci kontekstowej** — informuje operatora krok po kroku z potoku działań, "nigdy nie gubi wątku" inżynieryjnego.

Oś czasu — kamienie milowe AI



ĆWICZENIA PRAKTYCZNE

Trzy zadania dla przyszłego inżyniera

1 Naiwny system regułowy w Jupyter

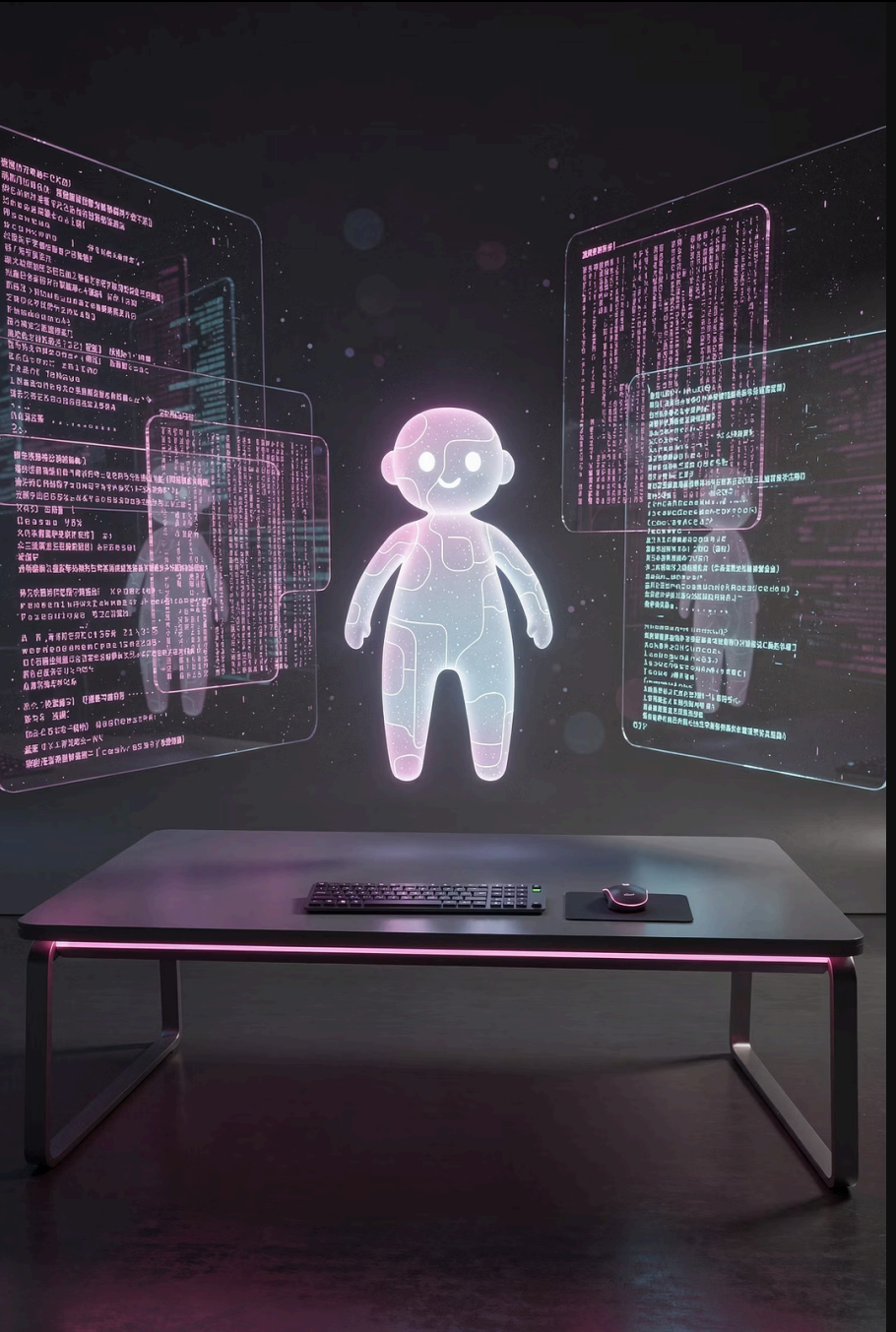
Zbuduj chatbot inspirowany ELIZĄ używając wyłącznie słowników i biblioteki `re` (regex). Wejście `"I feel scared about my job"` → dynamiczny string f-formatowania. Docblock: dlaczego regex nie ma nic wspólnego z głębokim wektorowym powiązaniem semantycznym Deep Learning.

2 Inżynieria kontekstowa — benchmarking LLM

Wklej błędny kod asyncio z zakleszczeniem (silent-bug). Zaprojektuj prompt "krok-po-kroku z objaśnieniem założeń". Porównaj dwa modele (np. OpenAI vs Anthropic) w tabeli Markdown — który halucynował, który utrzymał wektor konwersacyjny.

3 Esej techniczny (~300 słów, plik .md)

Który skok paradygmatyczny (twarde reguły → ML → Transformer → agentowe IDE) ma największy przełomowy wpływ na Twoje środowisko inżynierskie? Maksymalnie punktowane: odwołanie do "pustej bazy wiedzy" vs symultaniczności mechanizmu QKV. Wymagana krytyczna analiza zniekształceń modelu AI.



Kluczowe wnioski

Rozumiej historię

Każde ograniczenie architektoniczne dzisiejszych LLM ma korzenie w historii — od kruchości systemów regułowych po problem zanikającego gradientu.

Bądź architektem

Nie "klepaczem kodu". Definiuj problem, instruuj modele, audytuj bezpieczeństwo i integrację wygenerowanego kodu.

Zachowaj krytyczny horyzont

Model bez wagi tokenowej zaserwuje wadliwą architekturę bez wątpliwości. Twoja wartość to właśnie zdolność do wykrycia halucynacji maszyny.

Ignorowanie AI przez firmę lub programistę jest w 2025 roku równoznaczne z wydaleniem z rynku — dysonans produktywności jest niewspółmierny. Twoja przewaga to zrozumienie tej osi czasowej.