Sztuczna inteligencja

BEZ TAJEMNIC

M.ZDROWY

2026

Sztuczna inteligencja bez tajemnic

Sztuczna inteligencja bez tajemnic

Poradnik dla każdego

M.ZdrowY

2026

Sztuczna inteligencja bez tajemnic

Poradnik dla każdego

Spis treści

Rozdział 1. Skąd się wzięła sztuczna inteligencja?

Od pomysłu na papierze do asystenta w Twoim telefonie – historia, którą warto znać.

POZIOM 1

Dla laika

Nie musisz znać się na informatyce. Wszystko tłumaczymy od zera, na przykładach z życia codziennego.

Marzenie stare jak nauka

Wyobraź sobie rok 1950. Nie ma jeszcze kolorowej telewizji, większość ludzi nigdy nie widziała komputera, a lot na Księżyc to czysta fantastyka. I właśnie wtedy pewien brytyjski matematyk o nazwisku Alan Turing zadał pytanie, które zmieniło historię techniki: czy maszyna może myśleć?

Turing nie pytał, czy maszyna może liczyć szybciej niż człowiek – to już wiedział, że może. Pytał o coś głębszego: czy kiedyś uda się zbudować maszynę, która będzie rozmawiać z człowiekiem tak przekonująco, że ten nie będzie w stanie odróżnić jej od drugiego człowieka. Swoją myśl opisał w słynnym artykule i zaproponował test, który dziś nosi jego imię – Test Turinga.

Przez kolejne dekady tysiące naukowców próbowało sprostać temu wyzwaniu. Jedni budowali programy grające w szachy. Inni pisali systemy, które miały "rozumieć" proste zdania. Wszyscy po kilku latach entuzjazmu i milionach dolarów wydanych na badania odkrywali to samo: to jest znacznie trudniejsze, niż nam się wydawało.

Wzloty i upadki – długa droga do dziś

Historia sztucznej inteligencji to historia kilku wielkich nadziei i kilku bolesnych rozczarowań. Naukowcy nadali tym okresom mroczne nazwy – „zimy sztucznej inteligencji" – bo entuzjazm i finansowanie dosłownie zamarzały, gdy kolejne wielkie obietnice nie spełniały się w praktyce.

Pierwsza zima przyszła w latach 70. Badacze obiecywali, że za kilka lat komputery będą tłumaczyć języki i rozwiązywać problemy naukowe. Programy radziły sobie jedynie z bardzo wąskimi, z góry zdefiniowanymi zadaniami. Fundusze zostały obcięte, wielu naukowców straciło pracę, a temat AI na lata stał się passé w środowisku akademickim.

Kolejny okres zapału nastąpił w latach 80., gdy popularność zyskały tak zwane systemy eksperckie – programy, którym "wgrano" wiedzę tysięcy specjalistów w postaci reguł. Działały całkiem nieźle w wąskich dziedzinach, jak diagnozowanie chorób czy konfigurowanie sprzętu komputerowego. Ale miały poważną wadę: każdą regułę musiał ręcznie wpisać człowiek. Kiedy świat się zmieniał, programy nie nadążały. Nadeszła druga zima.

Dopiero na początku XXI wieku coś się przełamało. Nie dzięki jednemu przełomowemu odkryciu, ale dzięki połączeniu trzech czynników, które dojrzewały jednocześnie: ogromne ilości danych (internet), tania i dostępna moc obliczeniowa (karty graficzne) oraz nowe algorytmy. Ten splot okoliczności doprowadził do tego, co dziś nazywamy głębokim uczeniem.

2012 – rok, w którym zaczął się nowy świat

Gdybyś miał wskazać jeden moment, od którego warto zacząć rozumieć dzisiejszą AI, byłby to rok 2012. Na prestiżowym konkursie rozpoznawania obrazów, w którym komputery musiały samodzielnie rozpoznawać zdjęcia, pewna drużyna z Uniwersytetu w Toronto pokazała program, który pobił wszystkich konkurentów z miażdżącą przewagą.

Program nazywał się AlexNet i był siecią neuronową – systemem zainspirowanym działaniem ludzkiego mózgu, który sam się uczył rozpoznawać obrazy, zamiast być programowany regułami. Różnica w wynikach była tak duża, że świat akademicki momentalnie przestawił się na ten kierunek. Niemal z dnia na dzień tysiące laboratoriów zaczęły porzucać stare podejścia i budować sieci neuronowe.

W ciągu zaledwie kilku lat komputery nauczyły się rozpoznawać twarze, diagnozować choroby na zdjęciach rentgenowskich i rozumieć mowę lepiej niż człowiek. To, co przez dekady było science fiction, stawało się rzeczywistością.

ChatGPT i wielki przełom 2022 roku

Przez większość tej historii AI była czymś, o czym słyszałeś, ale czego nie dotykałeś. Była w laboratoriach, w serwerowniach korporacji, w samochodach autonomicznych. Przeciętny człowiek nie miał z nią bezpośredniego kontaktu.

30 listopada 2022 roku firma OpenAI udostępniła publicznie chatbota o nazwie ChatGPT. Można go było po prostu otworzyć w przeglądarce i porozmawiać. Bez rejestracji w korporacyjnym programie, bez wiedzy technicznej, bez kodowania. Wystarczyło wpisać pytanie po polsku – albo po angielsku, chińsku, arabsku – i dostać odpowiedź.

Reakcja świata była bezprecedensowa. W ciągu pięciu dni ChatGPT zdobył milion użytkowników. Dla porównania: Facebook potrzebował na to dziesięciu miesięcy, Spotify – pięciu lat. Po dwóch miesiącach użytkowników było już sto milionów. Żaden produkt w historii technologii nie rósł tak szybko.

Nagle wszyscy zaczęli rozmawiać o sztucznej inteligencji – nie tylko informatycy i futurolodzy, ale dziennikarze, nauczyciele, prawnicy, lekarze, a nawet politycy. Pojawiły się pytania, które jeszcze rok wcześniej brzmiałyby absurdalnie: czy AI zastąpi moją pracę? Czy AI może pisać lepiej ode mnie? Czy moje dzieci w ogóle będą musiały się czegoś uczyć?

Co AI naprawdę potrafi – a co to nie jest

Zanim pójdziemy dalej, warto rozwiać kilka mitów, które wyrosły na planie filmów science fiction.

MITY, KTÓRE WARTO OBALIĆ

MIT 1: "AI to robot jak z filmów"

Dzisiejsza AI to przede wszystkim oprogramowanie – programy komputerowe. Humanoidalne roboty z filmów to osobna dziedzina inżynierii, która na razie jest daleko w tyle za inteligencją.

MIT 2: "AI myśli i czuje jak człowiek"

Modele językowe nie "myślą" i nie "czują". Wykonują bardzo zaawansowane obliczenia statystyczne na tekście. Często wychodzi coś, co wygląda jak myślenie, ale pod spodem nie ma żadnej świadomości ani intencji.

MIT 3: "AI wie wszystko i zawsze ma rację"

AI może i będzie się mylić – czasem z dużą pewnością siebie. Zjawisko to nazywamy halucynacją i dokładnie je opisujemy w rozdziale 3.

MIT 4: "AI za chwilę przejmie kontrolę nad światem"

Dzisiejsza AI jest "wąska" – świetna w konkretnych zadaniach, bezradna poza nimi. Ogólna sztuczna inteligencja na poziomie ludzkim to temat badań i debat, a nie bliska rzeczywistość.

POZIOM 2

Głębiej

Dla tych, którzy chcą zrozumieć "pod maską". Wciąż bez wzorów matematycznych – ale z konkretnymi nazwami, datami i mechanizmami.

Kamienie milowe – szczegółowa oś czasu

Poniższa tabela pokazuje najważniejsze przełomy, które doprowadziły do dzisiejszego stanu AI. Każdy z nich jest momentem, gdy coś, co wcześniej było niemożliwe, nagle stało się możliwe.

Rok	Wydarzenie	Znaczenie
1950	**Test Turinga (Alan Turing)**	Pierwsze formalne pytanie: czy maszyna może myśleć? Turing proponuje test konwersacyjny jako miarę inteligencji.
1956	**Konferencja w Dartmouth**	Narodziny terminu "Artificial Intelligence". Naukowcy optymistycznie szacują, że rozwiążą problem AI w ciągu jednego lata badań.
1958	**Perceptron (Frank Rosenblatt)**	Pierwszy model matematyczny neuronu. Prosta sieć ucząca się rozpoznawać wzorce. Entuzjazm szybko gaśnie, gdy okazuje się, że nie skaluje się na trudniejsze problemy.
1974–80	**Pierwsza zima AI**	Brak postępów, cięcie funduszy. AI znika z agendy badań głównego nurtu na kilka lat.
1986	**Propagacja wsteczna**	Rumelhart, Hinton i Williams opisują algorytm, który umożliwia trenowanie głębszych sieci neuronowych. Przełom teoretyczny, ale sprzęt jeszcze nie nadąża.
1987–93	**Druga zima AI**	Systemy eksperckie okazują się zbyt kosztowne w utrzymaniu. Kolejna fala rozczarowania.
1997	**Deep Blue pokonuje Kasparowa**	Komputer IBM wygrywa mecz szachowy z mistrzem świata. Przełom medialny, ale Deep Blue to wąski system – nie rozumie nic poza szachami.

Rok	Wydarzenie	Znaczenie
2006	**Głębokie uczenie (Hinton)**	Geoffrey Hinton pokazuje, że głębsze sieci neuronowe można efektywnie trenować. Zaczyna się era deep learning.
2012	**AlexNet**	Sieć neuronowa druzgocąco wygrywa ImageNet. Świat akademicki masowo przestawia się na deep learning. Prawdziwy początek dzisiejszej AI.
2016	**AlphaGo pokonuje Lee Sedola**	DeepMind wygrywa z mistrzem świata w Go – grze uważanej za zbyt złożoną dla komputerów. Coś fundamentalnego się zmienia.
2017	**"Attention Is All You Need"**	Google Brain publikuje artykuł opisujący architekturę Transformer. To fundament wszystkich dzisiejszych modeli językowych – GPT, Claude, Gemini.
2020	**GPT-3 (OpenAI)**	Model z 175 miliardami parametrów potrafi pisać teksty nie do odróżnienia od ludzkich. Pierwsza demonstracja skali jako klucza do jakości.
2022	**ChatGPT**	Udostępnienie publiczne. 100 milionów użytkowników w 2 miesiące. AI wchodzi do mainstreamowej kultury.
2023–26	**Wyścig modeli**	GPT-5.5, Claude Opus 4.7, Gemini 3 Pro, Llama 5, DeepSeek V4-Pro, Qwen 3.7-Max. Modele stają się multimodalne, okna kontekstu rosną do milionów tokenów, modele lokalne stają się praktycznie użyteczne.

Dlaczego 2022 był punktem zwrotnym – trzy zbieżne fale

ChatGPT nie pojawił się znikąd. Był wynikiem jednoczesnego dojrzewania trzech niezależnych trendów, które przez lata rozwijały się równolegle, aż w końcu zbiegły się w jednym miejscu i czasie.

Fala pierwsza: dane

Sieć neuronowa uczy się ze wzorców w danych – im więcej danych, tym lepiej. Przez lata problemem był brak odpowiednio dużych zbiorów danych. Internet zmienił wszystko. Do końca 2022 roku ludzie opublikowali w sieci setki miliardów dokumentów, artykułów, książek,

rozmów i kodów. Firmy AI nauczyły się zeskrobywać te dane i używać ich jako surowca do trenowania modeli. Common Crawl – jeden z głównych zbiorów danych treningowych – zawiera petabajty tekstu z całej historii internetu.

Fala druga: moc obliczeniowa

Trening modelu językowego to miliardy operacji matematycznych. Przez dekady jedynym sposobem na ich wykonanie były drogie superkomputery. Zmianę przyniosły karty graficzne (GPU), które pierwotnie projektowano do renderowania gier wideo. Okazało się, że GPU są doskonałe do masowych obliczeń równoległych – dokładnie takich, jakich wymaga deep learning. NVIDIA, początkowo producent kart do gier, stała się głównym dostawcą infrastruktury AI. Jednocześnie chmura obliczeniowa (AWS, Google Cloud, Azure) umożliwiła wynajmowanie tysięcy GPU bez konieczności ich kupowania.

Fala trzecia: architektura Transformer

Sama ilość danych i moc obliczeniowa nie wystarczą – potrzebny jest jeszcze właściwy algorytm. Przełom nastąpił w 2017 roku, gdy zespół z Google Brain opublikował artykuł zatytułowany „Attention Is All You Need". Opisana w nim architektura Transformer rozwiązała fundamentalny problem wcześniejszych sieci: jak efektywnie przetwarzać długie sekwencje tekstu i rozumieć zależności między odległymi słowami. Mechanizm uwagi (attention) pozwala modelowi skupiać się na właściwych fragmentach tekstu podczas generowania każdego kolejnego słowa – jak czytanie ze zrozumieniem zamiast mechanicznego skanowania.

Wszystkie wielkie modele językowe, które znasz – GPT, Claude, Gemini, Llama, DeepSeek, Qwen, Mistral – są zbudowane na architekturze Transformer lub jej pochodnych.

Gracze, którzy kształtują dzisiejszą AI

Wyścig o dominację w AI to jeden z największych wyścigów technologicznych w historii. Poniżej najważniejsi gracze i ich rola:

- **OpenAI** – firma założona w 2015 roku jako non-profit przez Sama Altmana, Elona Muska i innych. Twórcy serii GPT i ChatGPT. Dziś przekształcona w firmę komercyjną, blisko powiązaną z Microsoftem (łącznie ponad 13 miliardów dolarów w latach 2019–

2024, w tym 10 miliardów w styczniu 2023, wycena przekraczająca 300 miliardów w 2026).

- **Google DeepMind** – połączenie Google Brain i londyńskiego DeepMind. Twórcy Gemini, modeli AlphaFold (rewolucja w biologii), AlphaGo. Jedni z najważniejszych badaczy fundamentalnych zagadnień AI.
- **Anthropic** – założone w 2021 roku przez byłych pracowników OpenAI, między innymi Daniela i Dario Amodei. Twórcy modeli Claude. Skupiają się mocno na bezpieczeństwie i interpretowalności AI.
- **Meta AI** – dział AI Mety (Facebook). Twórcy serii modeli Llama – udostępnianych jako otwarte oprogramowanie, co umożliwiło rozkwit ekosystemu lokalnych modeli AI.
- **Microsoft** – nie tworzy własnych dużych modeli podstawowych, ale jest głównym inwestorem OpenAI i integratorem AI w produktach (Copilot, Azure AI). Twórca BitNet i pionier formatów ONNX/Windows ML.
- **Mistral AI** – europejska firma (Paryż, 2023), która zaskoczyła świat wydajnymi, otwartymi modelami dużo mniejszymi niż konkurencja. Symbol europejskiej odpowiedzi na dominację Stanów.
- **DeepSeek** – chińska firma (Hangzhou), która w 2026 roku wstrząsnęła rynkiem modelem V4-Pro o wydajności porównywalnej z GPT-5.5 przy ułamku kosztów trenowania. Symbol rosnącej potęgi Chin w AI.
- **Alibaba (Qwen)** – chiński gigant e-commerce, którego modele z serii Qwen (3.6, 3.7-Max) regularnie plasują się w czołówce światowych rankingów. Qwen 3.7-Max to jeden z nielicznych modeli API-only, który dorównuje najlepszym zachodnim odpowiednikom.

Krótka uwaga o nazewnictwie

Wchodząc w świat AI, szybko natkniesz się na garść skrótów, które brzmią podobnie, ale oznaczają różne rzeczy. Oto najważniejsze z nich, które będą pojawiać się przez cały poradnik:

AI (Artificial Intelligence) – sztuczna inteligencja. Termin parasolowy na wszystkie technologie naśladujące ludzkie zdolności poznawcze.

ML (Machine Learning) – uczenie maszynowe. Poddziedzina AI, w której systemy uczą się z danych zamiast być programowane regułami.

DL (Deep Learning) – głębokie uczenie. Poddziedzina ML oparta na głębokich sieciach neuronowych. To tutaj następuje większość dzisiejszych przełomów.

LLM (Large Language Model) – duży model językowy. Typ sieci neuronowej wytrenowanej na ogromnych zbiorach tekstu. ChatGPT, Claude, Gemini – to wszystko LLM. Poświęcamy im cały rozdział 3.

GPT (Generative Pre-trained Transformer) – konkretna rodzina modeli OpenAI. Słowo Transformer odnosi się do architektury z 2017 roku. GPT nie jest synonimem AI – to jedna z wielu rodzin modeli.

→ W następnym rozdziale: **Czym dokładnie jest sztuczna inteligencja i jak naprawdę działa uczenie maszynowe?**

Rozdział 2. Czym jest sztuczna inteligencja?

AI, ML, DL, sieci neuronowe, parametry – rozplątujemy terminologię, którą wszyscy mieszają.

POZIOM 1

Dla laika

Nie musisz znać się na informatyce. Wszystko tłumaczymy od zera, na przykładach z życia codziennego.

Trzy pojęcia, które ludzie mylą

Gdy czytasz o sztucznej inteligencji, prawdopodobnie spotykasz trzy pojęcia: AI, uczenie maszynowe i głębokie uczenie. Większość ludzi używa ich zamiennie, jakby znaczyły to samo. Ale to tak, jakbyś powiedział, że samochód, silnik i tłok to to samo. Są ze sobą powiązane – ale każde oznacza coś innego i pełni inną funkcję.

Najlepszą analogią są rosyjskie matrioszki. Wyobraź sobie trzy drewniane lalki, z których każda mieści się w większej:

- Największa lalka to sztuczna inteligencja (AI) – cała dziedzina nauki o tym, jak sprawić, by maszyny zachowywały się inteligentnie.
- Średnia lalka to uczenie maszynowe (ML) – część AI, w której maszyny uczą się z danych, zamiast być programowane regułami.
- Najmniejsza lalka to głębokie uczenie (DL) – zaawansowana część ML, korzystająca z sieci neuronowych o wielu warstwach.

Wszystkie trzy pojęcia mieszczą się jedno w drugim. Nie każde AI to ML (są systemy AI, które nie uczą się, tylko działają według sztywnych reguł). Nie każdy ML to DL (są prostsze techniki uczenia się). Ale każdy DL to jednocześnie ML i AI.

Co AI już potrafi – ze swojego telefonu

Sztuczna inteligencja nie jest już futurystyczną wizją z laboratoriów. Używasz jej codziennie, często nawet nie świadomie. Oto kilka przykładów z Twojego telefonu:

- Netflix i Spotify – AI analizuje, co oglądasz i słuchasz, i podpowiada Ci kolejne treści. To uczenie maszynowe w czystej postaci: system widzi miliony użytkowników i uczy się wzorców Twoich zachowań.
- Mapy Google – AI przewiduje, jaki będzie ruch na drodze za godzinę, i proponuje optymalną trasę. Bierze pod uwagę dane historyczne, porę dnia, wypadki i tysiące innych czynników.
- Aparat w telefonie – gdy robisz zdjęcie w nocy, AI „domyśla się", jakie kolory powinny być w ciemnych obszarach, i dokłada je cyfrowo.
- Asystenci głosowi (Siri, Google Assistant) – rozpoznają Twoją mowę, zamieniają ją na tekst, rozumieją intencję i wykonują polecenie.
- ChatGPT, Claude, Gemini – modele językowe, które rozumieją pytania i generują odpowiedzi. To najbardziej zaawansowana forma AI dostępna publicznie.

Czego AI nie umie i prawdopodobnie długo nie będzie umieć

O AI krąży wiele mitów. Warto wiedzieć, czego dzisiejsza technologia na pewno nie potrafi – i dlaczego nie powinieneś się obawiać, że za chwilę zastąpi Cię we wszystkim.

[UWAGA] CO JEST POZA ZASIĘGIEM AI (NA RAZIE)

- **Świadomość i emocje**

AI nie „czuje" ani „odczuwa". Nie ma własnych pragnień, lęków ani celów. Gdy chatbot pisze „cieszę się, że mogłem pomóc", to wzorzec językowy, a nie prawdziwa radość.

✓ **Zdrowy rozsądek**

AI może napisać esej o fizyce kwantowej, ale nie wie, że mokry parasol zostawia kałużę na podłodze. Nie ma doświadczenia życiowego – ma tylko teksty, które przeczytało.

✓ **Prawdziwe rozumowanie**

AI nie rozumie w ludzkim sensie. Przetwarza wzorce statystyczne. Może zdać egzamin prawniczy, ale nie rozumie sprawiedliwości.

✓ **Generalizacja poza zakresem**

AI jest świetna w tym, do czego została wytrenowana. Gdy dasz jej zadanie spoza jej domeny, zwykle zawodzi spektakularnie.

✓ **Przyczynowość**

AI widzi korelacje, ale nie rozumie przyczynowości. Widzi, że po spadku cen parasoli wzrasta sprzedaż, ale nie łączy tego z deszczem.

Skala inteligencji – gdzie jesteśmy?

Gdy naukowcy mówią o AI, rozróżniają dwa zupełnie różne poziomy inteligencji. Wąska AI (Narrow AI) to wszystko, co mamy dziś: systemy, które świetnie radzą sobie z jednym zadaniem. Ogólna AI (AGI) to hipotetyczna maszyna, która dorówna człowiekowi we wszystkim – naukowcy dyskutują, czy to w ogóle możliwe.

Dla porównania: dzisiejsze AI potrafi napisać lepszą notatkę służbową niż większość ludzi, ale nie potrafi nauczyć się wiązać sznurowadeł bez specjalnego treningu. To klasyczna wąska AI. Jesteśmy mniej więcej w punkcie, w którym byliśmy z lotnictwem w latach 20. XX wieku – samoloty istniały i działały, ale nikt jeszcze nie latał na Księżyc.

POZIOM 2

Głębiej

Dla tych, którzy chcą zrozumieć „pod maską". Wciąż bez wzorów matematycznych – ale z konkretnymi nazwami, datami i mechanizmami.

Jak działa sieć neuronowa – fundament dzisiejszej AI

Sieć neuronowa to matematyczny model zainspirowany (bardzo luźno) tym, jak działają neurony w mózgu. Nie jest to dosłowna kopia mózgu – raczej inspiracja, tak jak skrzydła samolotu są inspirowane ptakami, ale działają inaczej.

Neurony, wagi i funkcje aktywacji

Wyobraź sobie, że sieć neuronowa składa się z warstw małych „kalkulatorów" (neuronów). Każdy neuron otrzymuje na wejście liczby, mnoży je przez wagi (parametry określające, jak ważne jest dane wejście), dodaje stałą wartość zwaną biasem, a następnie przepuszcza wynik przez funkcję aktywacji, która decyduje, czy neuron się „uruchamia".

Przykład z życia: wyobraź sobie, że neuron to ekspert, który odpowiada na pytanie „czy to jest zdjęcie kota?". Na wejściu dostaje informacje: „ma futro" (waga: 0.9), „ma wąsy" (waga: 0.8), „ma cztery nogi" (waga 0.3), „jest z metalu" (waga: -0.9). Wagi ujemne oznaczają, że cecha przeczy kocie. Neuron waży wszystkie sygnały i podejmuje decyzję.

Propagacja wsteczna – jak sieć się uczy

Gdy sieć popełnia błąd (np. mówi, że pies to kot), uruchamia się mechanizm zwany propagacją wsteczną. To jak nauczyciel, który czerwonym długopisem zaznacza błędy i wypisuje prawidłowe odpowiedzi. Algorytm oblicza, które wagi w której warstwie najbardziej przyczyniły się do błędu, i poprawia je o małą wartość.

Ten proces powtarza się miliony razy. Z każdą iteracją sieć staje się lepsza. To nic innego jak optymalizacja – poszukiwanie najlepszego zestawu wag, który minimalizuje błąd. W tym sensie uczenie sieci neuronowej przypomina próbę znalezienia najniższego punktu w górzystym terenie – algorytm (gradient descent) robi małe kroki w dół, aż znajdzie dolinę.

Wąska vs ogólna AI – gdzie jesteśmy na skali

To jedna z najważniejszych klasyfikacji w świecie AI:

- **Wąska AI (ANI)** – Artificial Narrow Intelligence. Specjalizuje się w jednym zadaniu. Każdy dzisiejszy system AI to wąska AI. ChatGPT jest wąskie – świetne w tekście, bezużyteczne w jeździe samochodem. Szachiści Deep Blue i AlphaGo to wąska AI – nie umieją nic poza swoją grą.
- **Ogólna AI (AGI)** – Artificial General Intelligence. Hipotetyczna maszyna, która dorównuje człowiekowi w każdej dziedzinie: od matematyki, przez malowanie, po rozmowę. To wciąż science fiction – nie ma consensusu, czy powstanie za 5, 50, czy 500 lat.
- **Super AI (ASI)** – Artificial Superintelligence. Maszyna przewyższająca człowieka we wszystkim. Czysta spekulacja, temat debat filozoficznych i ostrzeżeń ze strony ekspertów takich jak Nick Bostrom.

Obecnie znajdujemy się zdecydowanie na etapie wąskiej AI. Nawet najpotężniejsze modele są „inteligentne" tylko w swojej dziedzinie. GPT-5.5 nie potrafi samodzielnie zrobić kawy, nawet gdybyś kazał mu przeczytać sto książek kucharskich.

Rodzaje modeli – każdy do czegoś innego

Nie wszystkie modele AI są takie same. Różnią się tym, co potrafią robić – a to zależy od tego, na czym były trenowane i do czego zostały zaprojektowane.

Modele językowe (LLM)

Działają na tekście. Rozumieją pytania, generują odpowiedzi, tłumaczą, streszczają, piszą kody. Przykłady: GPT-5.5 (OpenAI), Claude Opus 4.7 (Anthropic), Gemini 3 Pro (Google), Llama 5 (Meta), DeepSeek V4-Pro (DeepSeek). Poświęcamy im cały rozdział 3. To właśnie one zrobiły największą rewolucję w ostatnich latach.

Modele obrazowe

Analizują i generują obrazy. Dzielą się na modele rozpoznawania (np. rozpoznawanie chorób na zdjęciach RTG) i modele generatywne (tworzenie obrazów z opisu tekstowego, jak Midjourney, DALL-E 4, FLUX.2, Stable Diffusion Ultra). Te drugie zyskały ogromną popularność od 2022 roku.

Modele dźwiękowe

Rozpoznają mowę (jak Whisper od OpenAI), generują mowę (TTS, text-to-speech) i analizują dźwięki. Dzięki nim możesz mówić do ChatGPT, a on odpowiada głosem.

Modele multimodalne

To najnowsza i najbardziej ekscytująca kategoria. Jeden model obsługuje tekst, obrazy, dźwięki – wszystko naraz. GPT-5.5, Claude Opus 4.7, Gemini 3 Pro są multimodalne: możesz pokazaĄ im zdjęcie i zapytać o nie, a one odpowiedzą. To krok w stronę tego, jak działa człowiek – łączymy różne zmysły.

Czym są parametry i dlaczego ich liczba ma znaczenie

Gdy czytasz, że model ma 70 miliardów parametrów (70B), co to właściwie znaczy? Parametry to wszystkie wagi i biasy w sieci neuronowej. Każdy parametr to po prostu liczba zmiennoprzecinkowa, która została wyuczona w procesie treningu. Im więcej parametrów, tym więcej model może zapamiętać wzorców.

Analogia: parametry są jak półki w bibliotece. Mały model (7B parametrów) to biblioteczka w domowym gabinecie – ma najważniejsze książki. Model średni (70B) to biblioteka uniwersytecka. Wielki model (405B) to Biblioteka Kongresu Stanów Zjednoczonych. Więcej półek (parametrów) = więcej miejsca na wiedzę.

Ale większy model to nie tylko więcej wiedzy – to także większe koszty. GPT-3 miał 175 miliardów parametrów, a jego trening kosztował około 4,6 miliona dolarów. GPT-4 (szacunkowo 1,8 biliona parametrów) kosztował około 100-200 milionów dolarów. Dla porównania, GPT-5.5 (2026) ma szacunkowo ponad 5 bilionów parametrów (architektura MoE), a jego trening kosztował około 500 milionów dolarów. Dlatego właśnie liczba parametrów ma znaczenie – więcej = drożej, ale potencjalnie lepiej.

Tabela: przegląd ważniejszych modeli według typu

Typ modelu	Przykłady	Zastosowanie
Językowe (LLM)	GPT-5.5, Claude Opus 4.7, Gemini 3 Pro, Llama 5, DeepSeek V4-Pro, Qwen 3.7-Max	Pisanie, tłumaczenie, kod, analiza tekstu
Obrazowe	Midjourney, DALL-E 4, Stable Diffusion Ultra, FLUX.2	Generowanie obrazów, edycja zdjęć, design
Dźwiękowe	Whisper (OpenAI), ElevenLabs, Bark	Transkrypcja, synteza mowy, rozpoznawanie głosu
Multimodalne	GPT-5.5, Gemini 3 Pro, Claude Opus 4.7	Tekst + obraz + dźwięk w jednym

---

→ W następnym rozdziale: **Modele językowe (LLM) – serce nowoczesnej AI. Jak działają, jak są trenowane i dlaczego halucynują?**

Rozdział 3. Modele językowe (LLM) – serce nowoczesnej AI

Jak działają, jak są trenowane, dlaczego halucynują – i który wybrać dla siebie.

POZIOM 1

Dla laika

Nie musisz znać się na informatyce. Wszystko tłumaczymy od zera, na przykładach z życia codziennego.

Co to jest model językowy?

Model językowy to program komputerowy, który „rozumie" i generuje tekst. Nie jest to jednak zrozumienie w ludzkim sensie – model językowy to maszyna do przewidywania. Jego podstawowe zadanie brzmi: „które słowo (a dokładniej: fragment słowa) najprawdopodobniej pojawi się jako następne?".

Wyobraź sobie, że ktoś pisze zdanie: „Królowa Elżbieta II zmarła w zamku …". Nawet człowiek, który nie zna odpowiedzi, może się domyślić, że następnym słowem jest „Balmoral". Model językowy robi to samo, ale na ogromną skalę. „Czytał' miliardy tekstów i zapamiętał, które słowa najczęściej pojawiają się w jakim kontekście.

Gdy wpisujesz pytanie do ChatGPT, model nie szuka odpowiedzi w bazie danych ani w internecie (chyba że ma włączoną opcję przeglądania sieci). On po prostu generuje tekst słowo po słowie, za każdym razem pytając siebie: „co najprawdopodobniej powinno być następne?". To wyjaśnia, dlaczego modele językowe potrafią być tak płynne – i dlaczego czasem mówią rzeczy, które nie są prawdą.

Skąd model wie to, co wie?

Model językowy nie ma dostępu do internetu ani do żadnej bazy wiedzy w tradycyjnym sensie. Jego wiedza pochodzi wyłącznie z procesu zwanego treningiem. Podczas treningu model jest karmiony ogromnymi ilościami tekstu – całymi bibliotekami cyfrowymi, milionami książek, artykułów, stron internetowych, kodów źródłowych i rozmów.

Model nie zapamiętuje tych tekstów słowo w słowo (to było by niemożliwe – nawet największy model ma ograniczoną pojemność). Zamiast tego uczy się wzorców i zależności: jak język jest zbudowany, jakie są reguły gramatyki, jakie fakty są ze sobą powiązane. To trochę jak nauka języka przez zanurzenie – nie uczysz się reguł z podręcznika, tylko wyciągasz je z milionów usłyszanych zdań.

Ponieważ trening jest zamknięty w czasie, wiedza modela ma datę ważności. GPT-5.5 (model z 2026 roku) nie wie o wydarzeniach z 2027 roku, chyba że ma włączone przeszukiwanie internetu. To ważne ograniczenie, o którym warto pamiętać, korzystając z AI.

Kontekst – dlaczego AI pamięta tylko do pewnego momentu

Gdy rozmawiasz z ChatGPT, on pamięta, o czym była mowa wcześniej w tej samej rozmowie. Ale tylko do pewnego momentu. To dlatego, że każdy model ma ograniczony „kontekst" – maksymalną długość tekstu, jaką może jednorazowo przetworzyć.

Wyobraź sobie stół, na którym możesz położyć tylko określoną liczbę kartek. Gdy dokładasz nową, najstarsza spada. W wczesnych modelach (GPT-3) kontekst wynosił 4 096 tokenów – mniej więcej 3 000 słów. Współczesne modele potrafią obsłużyć setki tysięcy, a nawet miliony tokenów. Gemini 3 Pro ma kontekst 2 milionów tokenów, a Llama 4 Scout oferuje 10 milionów tokenów – to jak przeczytanie całego „Potopu" Sienkiewicza i odpowiedzenie na pytania o niego.

Ale większy kontekst to większy koszt obliczeniowy. Im więcej tekstu model musi uwzględnić, tym wolniej odpowiada i tym więcej energii zużywa. Dlatego producenci modeli oferują różne wersje z różnymi rozmiarami okna kontekstu – w zależności od tego, do czego ma służyć model.

Halucynacje – dlaczego AI pewnie mówi nieprawdę

To jedno z najważniejszych zjawisk, jakie musisz zrozumieć, korzystając z modeli językowych. Halucynacja to sytuacja, w której model generuje informację, która brzmi przekonująco, ale jest całkowicie nieprawdziwa. Co więcej, model mówi to z taką samą pewnością, jak wtedy, gdy mówi prawdę.

Dlaczego tak się dzieje? Ponieważ model nie jest zaprojektowany do bycia prawdomównym. Jest zaprojektowany do generowania przekonującego tekstu. Gdy nie zna odpowiedzi, nie mówi „nie wiem" – on „zgaduje" na podstawie wzorców i statystyk, a potem formułuje to w piękne, przekonujące zdania.

Przykład: zapytaj model, kto wygrał wyścig kolarski w 1875 roku. Model nie ma tej informacji w danych treningowych, ale jeśli jest dobrze wytrenowany, nie powie „nie wiem". Zamiast tego wygeneruje nazwisko, datę i opis wyścigu, które będą brzmieć autorytatywnie – a będą całkowicie zmyślone.

Jak się przed tym bronić? Zawsze traktuj informacje z AI jako podejrzane i weryfikuj je w źródłach. To szczególnie ważne przy danych, które mają wpływ na decyzje – zdrowotne, prawne, finansowe. Nowoczesne techniki, takie jak RAG (o których w części praktycznej), znacznie redukują halucynacje, ale nie eliminują ich całkowicie.

Popularne modele językowe – przegląd

Rynek modeli językowych rozwija się błyskawicznie. Poniżej najważniejsze z nich:

- **GPT-5.5 (OpenAI)** – najlepszy model (SOTA) do kodowania i matematyki. Multimodalny (tekst, obrazy, dźwięk). Dostępny przez ChatGPT Plus/Pro i API. Jest punktem odniesienia, z którym porównuje się wszystkie inne.
- **Claude Opus 4.7 (Anthropic)** – najlepszy model do długich dokumentów i precyzyjnego rozumowania. Kontekst do 1M tokenów. Doskonały w analizie kontraktów, kodu i prac badawczych.
- **Gemini 3 Pro / 3.5 Flash (Google)** – multimodalne, kontekst do 2M tokenów [FAKT?]. Głęboko zintegrowane z ekosystemem Google. Flash jest szybszy i tańszy, Pro oferuje najwyższą jakość.

- **Llama 5 / Llama 4 (Meta)** – otwarte modele. Llama 4 Maverick (koder) i Scout (10M kontekstu). Llama 5 to najnowsza generacja, dostępna w wariantach 8B, 70B i 405B+.
- **Mistral AI (Francja)** – europejska konkurencja. Oferuje Codestral 2 (model do kodowania), Mistral Large 4 oraz Le Chat. Otwarte, wydajne – dostępne przez API i lokalnie.
- **DeepSeek V4-Pro / V4-Flash (DeepSeek)** – chińskie modele open-weight na licencji MIT. V4-Pro to SOTA wśród otwartych modeli, V4-Flash to szybsza, lżejsza wersja. Architektura MoE, dostępne również DeepSeek V3.2 i R2.

POZIOM 2

Głębiej

Dla tych, którzy chcą zrozumieć „pod maską". Wciąż bez wzorów – ale z konkretami.

Tokenizacja i osadzanie – jak model widzi tekst

Zanim model językowy w ogóle zacznie przetwarzać tekst, musi go zamienić na liczby. Modele nie czytają liter ani słów – czytają ciągi liczb. Ten proces nazywa się tokenizacja.

Tokenizacja

Token to podstawowa jednostka tekstu, jaką rozumie model. Nie musi to być całe słowo – często jest to fragment. Na przykład słowo „nieszczęśliwy" może zostać podzielone na tokeny: „nie" + „szczęś" + „liwy". Model ma słownik takich tokenów (dla GPT-4o to około 100 000 różnych tokenów) i każdemu przypisuje unikalny numer identyfikacyjny.

Różne modele używają różnych metod tokenizacji. GPT-4 używa BPE (Byte-Pair Encoding), która dzieli tekst na najczęstsze podciągi znaków. To wyjaśnia, dlaczego modele językowe lepiej radzą sobie z językami, które są dobrze reprezentowane w danych treningowych (angielski) niż z tymi słabiej reprezentowanymi (polski potrzebuje więcej tokenów na tę samą treść).

Osadzanie (embedding)

Po tokenizacji każdy token zostaje zamieniony na wektor – ciąg liczb (np. 4096 liczb dla GPT-4o). Ten wektor koduje znaczenie tokena. Ważne jest, że tokeny o podobnym znaczeniu mają podobne wektory. Wektor dla „króla" i „królowej" są bliżej siebie niż wektor dla „króla" i „pomidora".

Co ciekawe, można na tych wektorach wykonywać operacje arytmetyczne. Słynny przykład: wektor(„król") – wektor(„mężczyzna") + wektor(„kobieta") = wektor(„królowa"). Model nie wie świadomie, co robi – ale matematyka wektorów uchwyciła tę relację. To właśnie siła osadzań.

Mechanizm uwagi (attention) – klucz do kontekstu

Gdy model czyta zdanie „Jan wziął książkę, którą Maria mu pożyczyła, i położył ją na stole", musi wiedzieć, że „ją" odnosi się do książki, a nie do Marii. Dla człowieka to oczywiste, dla maszyny było to przez dekady jednym z najtrudniejszych wyzwań.

Mechanizm uwagi (attention) to pomysł, który to zmienił. Opisany w 2017 roku w artykule „Attention Is All You Need", pozwala modelowi dynamicznie oceniać, które słowa w zdaniu są ważne dla zrozumienia każdego innego słowa.

Wyobraź sobie, że czytasz książkę i markerem zaznaczasz zależności między słowami. „Jan" → „wziął", „książkę" → „którą", „Maria" → „pożyczyła". Teraz wyobraź sobie, że robisz to jednocześnie dla wszystkich słów i ważysz siłę każdego połączenia. To właśnie robi mechanizm uwagi.

Dzięki tej architekturze modele Transformer (wszystkie współczesne LLM) potrafią „rozumieć" kontekst na ogromnych dystansach – setek tysięcy słów. Wcześniejsze modele (RNN, LSTM) traciły kontekst po kilkudziesięciu słowach. To właśnie mechanizm uwagi jest tajną bronią współczesnych modeli językowych.

Proces treningu – dane, czas, prąd

Trenowanie dużego modelu językowego to jedna z największych operacji obliczeniowych współczesnego świata. Oto skala:

Dane

GPT-3 (175B parametrów) był trenowany na około 570 GB tekstu. To mniej więcej tyle, ile zajmują wszystkie książki w Bibliotece Kongresu USA pomnożone przez trzy. GPT-4 (szacunkowo 1.8T parametrów) był trenowany na jeszcze większym zbiorze – szacunki mówią o 10-20 bilionach tokenów.

Czas

Trening GPT-3 zajął kilka miesięcy na tysiącach kart graficznych NVIDIA. GPT-4 trenował się prawdopodobnie przez około 3-4 miesiące na 25 000+ GPU NVIDIA A100. Dla porównania:

gdybyś próbował wytrenować GPT-4 na jednym nowoczesnym laptopie, zajęłoby Ci to około 50 000 lat.

Prąd

Trening GPT-3 zużył około 1 300 MWh energii elektrycznej. To mniej więcej tyle, ile 130 amerykańskich domów zużywa w ciągu roku. GPT-4 zużył szacunkowo 50 000 MWh. Emisja CO2 była ogromna, choć dokładne liczby nie zostały ujawnione. Dlatego rośnie presja na tworzenie bardziej efektywnych energetycznie modeli.

Po treningu następuje fine-tuning (dostrojenie) na mniejszych, starannie wyselekcjonowanych zbiorach danych. To jak korekta kursu po przepłynięciu oceanu. A potem – RLHF.

Co oznaczają liczby 7B, 70B, 405B?

Oznaczają liczbę parametrów modelu (w miliardach). 7B = 7 miliardów, 70B = 70 miliardów, 405B = 405 miliardów. Im więcej parametrów, tym większą wiedzę model może przechować, ale też tym więcej pamięci i mocy obliczeniowej potrzebuje.

Parametry	Wymagania	Przykłady modeli
7B-13B	8-16 GB RAM/VRAM	Llama 5 8B, Gemma 4 26B A4B, Qwen 3.6 7B, Mistral Large 4, DeepSeek V4-Flash
30B-70B	32-64 GB VRAM	Llama 5 70B, DeepSeek V4-Pro, Qwen 3.7-Max, Mistral Large 4 70B
170B-405B	80-512 GB VRAM	Llama 5 405B, DeepSeek V4-Pro (1.6T MoE), GPT-5.5 (szac. 5T MoE)

MoE (Mixture of Experts) – duże modele udające małe

Tradycyjny model językowy (tzw. gęsty, dense) używa wszystkich swoich parametrów do każdego wygenerowanego tokena. Model MoE (mieszanina ekspertów) działa inaczej: ma wiele „ekspertów" (oddzielnych podsieci) i router, który decyduje, którego eksperta użyć dla danego tokena.

Wyobraź sobie szpital z wieloma specjalistami. Gdy pacjent ma problem z sercem, idzie do kardiologa, a nie do ortopedy. W modelu gęstym każdy pacjent byłby badany przez wszystkich lekarzy naraz. W modelu MoE tylko właściwy ekspert jest aktywny.

DeepSeek V4-Pro ma 1 600 miliardów (1.6 biliona) parametrów ogółem, ale na każdy token aktywuje tylko 49 miliardów. Reszta parametrów „śpi". Dzięki temu model jest tak potężny jak modele z setkami miliardów parametrów, ale tak szybki (i tani) jak model kilkudziesięciomiliardowy. To dlatego większość nowoczesnych modeli (DeepSeek V4, Qwen 3.7-Max, Mixtral, GLM-5.1) przechodzi na architekturę MoE.

RLHF – jak człowiek „wychowuje" model

Po treningu na surowych danych model jest inteligentny, ale niekoniecznie pomocny czy bezpieczny. Wie, jak odpowiadać na pytania, ale może też generować treści szkodliwe, nieprawdziwe lub obraźliwe. Tu wchodzi RLHF – Reinforcement Learning from Human Feedback (uczenie przez wzmacnianie z informacją zwrotną od człowieka).

Proces RLHF składa się z trzech etapów:

- **Etap 1: Zbieranie danych** – ludzie oceniają odpowiedzi modelu, rankingując je od najlepszej do najgorszej. Dla tego samego pytania generuje się kilka odpowiedzi i człowiek wybiera najlepszą.
- **Etap 2: Model nagrody** – na podstawie tych ocen trenuje się osobny model (model nagrody), który uczy się przewidywać, która odpowiedź bardziej spodoba się człowiekowi.
- **Etap 3: Dostrajanie wzmocnieniowe** – główny model jest dostrajany tak, by maksymalizować wynik z modelu nagrody. Innymi słowy: uczy się, co ludzie lubią.

Dzięki RLHF ChatGPT stał się tym, czym jest – pomocnym asystentem, a nie tylko statystycznym generatorem tekstu. Claude od Anthropic używa rozszerzonej wersji (Constitutional AI), gdzie część „wychowania" odbywa się automatycznie na podstawie zestawu zasad.

Bonus: PLLuM – polski model językowy

Polska nie pozostaje w tyle. W lutym 2025 roku Ministerstwo Cyfryzacji zaprezentowało PLLuM (Polish Large Language Model) – rodzimą rodzinę modeli językowych, stworzoną przez konsorcjum sześciu instytucji pod przewodnictwem Politechniki Wrocławskiej i NASK.

PLLuM to nie jeden model, a cała rodzina 18 wersji, od 8 do 70 miliardów parametrów. Co wyróżnia PLLuM na tle konkurencji?

- Dane organiczne – model był trenowany na polskich tekstach zebranych ręcznie, nie generowanych syntetycznie przez inne modele. Obejmuje 100 miliardów słów, w tym teksty literackie, dokumenty urzędowe i zasoby naukowe.
- Pełna kontrola – jako narodowy model, PLLuM eliminuje zależność od zagranicznych dostawców. Jest rozwijany pod marką HIVE AI w NASK.
- Zastosowania urzędowe – model trafi do administracji publicznej. Wirtualny asystent w aplikacji mObywatel to pierwsze wdrożenie. Kolejne to inteligentny asystent dla urzędników.
- Architektura MoE – część modeli PLLuM używa miksowania ekspertów, co zapewnia wydajność.

PLLuM jest dostępny na otwartej licencji na Hugging Face (CYFRAGOVPL) oraz jako interfejs online na pllum.clarin-pl.eu. To ważny krok w stronę cyfrowej suwerenności Polski.

→ W następnym rozdziale: **Chmurowe modele językowe – ChatGPT, Claude, Gemini w chmurze. Kiedy płacić, a kiedy zostać na darmowym planie?**

Rozdział 4. Chmurowe modele językowe

ChatGPT, Claude, Gemini, Copilot – na czym polegają, ile kosztują i co się dzieje z Twoimi danymi?

POZIOM 1

Dla laika

Zrozumiesz, czym są chmurowe usługi AI, jak je porównywać i czy warto płacić.

Czym jest chmura – analogia z mieszkaniem

Gdy mówimy o „chmurze" w kontekście AI, nie chodzi o pogodę. Chmura to po prostu ogromne centra danych pełne serwerów, które należą do firm takich jak Microsoft, Google czy Amazon. Gdy korzystasz z ChatGPT, Twoje pytanie nie jest przetwarzane na Twoim komputerze – wędruje przez internet do jednego z tych serwerów, tam model AI generuje odpowiedź, a wynik wraca do Ciebie.

Najlepszą analogią jest różnica między wynajmem mieszkania a kupnem domu.

WYNAJEM MIESZKANIA (CHMURA)

Płacisz miesięczny czynsz (subskrypcję). Nie przejmujesz się remontem, ogrzewaniem ani awariami – to robi właściciel (dostawca chmury). Możesz wyprowadzić się w każdej chwili. Ale nie masz wpływu na to, co sąsiad robi ściany obok, i nie wiesz dokładnie, kto jeszcze ma klucze.

KUPNO DOMU (LOKALNIE)

Płacisz raz (zakup sprzętu). Masz pełną kontrolę, nikt nie zagląda Ci przez okno. Ale sam musisz zająć się konserwacją, a awarie kosztują.

Większość ludzi zaczyna od wynajmu (chmury): to proste, nie wymaga inwestycji, działa od razu. Później, gdy potrzeby rosną lub pojawia się kwestia prywatności, część osób decyduje się na

własny „dom" (modele lokalne). Ale większość użytkowników pozostaje w chmurze – to po prostu wygodniejsze.

ChatGPT, Claude, Gemini, Copilot – czym się różnią?

Na pierwszy rzut oka wszystkie wyglądają podobnie: okno czatu, wpisujesz pytanie, dostajesz odpowiedź. Ale pod spodem każdy używa innego modelu i ma inne mocne strony.

- **ChatGPT (OpenAI)** – najpopularniejszy, najbardziej wszechstronny. Działa na modelach GPT-5.5 i GPT-4o (dla tanszych zapytan). Świetny do pisania, kodowania, burzy mózgów i zadań ogólnych. Ma tryb głosowy, generowanie obrazów (DALL-E) i analizę plików.
- **Claude (Anthropic)** – specjalista od długich dokumentów i precyzyjnego rozumowania. Ma okno kontekstu 1 miliona tokenów (opcjonalnie 200K dla szybszych odpowiedzi). Często wybierany do analizy kontraktów, kodu i prac badawczych. Bardzo bezpieczny – rzadko generuje szkodliwe treści.
- **Gemini (Google)** – zintegrowany z ekosystemem Google. Ma największe okno kontekstu (2 miliony tokenów). Potrafi analizować filmy, obrazy i dźwięk.
  Działa w Gmailu, Dyskach Google i Dokumentach. Darmowa wersja jest bardzo hojna.
- **Copilot (Microsoft)** – wbudowany w Windows, Office i przeglądarkę Edge.
  Działa na modelach OpenAI, ale zintegrowany z Wordem, Excelem i PowerPointem.
  Idealny, jeśli pracujesz w pakiecie Microsoft.
- **Grok (xAI)** – stosunkowo nowy gracz od Elona Muska. Wyróżnia się osobowością i dostępem do bieżących informacji z platformy X (dawniej Twitter). Dobry do analizy trendów w czasie rzeczywistym.
- **Mistral (francuski start-up)** – europejska alternatywa z siedzibą w Paryżu.
  Oferuje modele przez API i czat internetowy (Le Chat). Dla osób, które chcą uniknąć amerykańskich dostawców.

Darmowe czy płatne – co realnie zyskujesz?

Każdy z głównych dostawców oferuje darmowy plan. W 2026 roku wygląda to mniej więcej tak:

Usługa	Darmowy plan	Płatny plan	Co daje płatny?
ChatGPT	GPT-4o mini, ograniczona liczba zapytań	ChatGPT Plus: $20/mies. ($240/rok)	Pełny GPT-5.5, szybsze odpowiedzi, DALL-E 4, analiza plików, tryb głosowy
Claude	Claude Sonnet 4.6, ~20-30 zapytań dziennie	Claude Pro: $20/mies.	Dostęp do Claude Opus 4.7, 5x więcej użycia, dłuższy kontekst do 1M
Gemini	Gemini 3.5 Flash, hojny limit	Gemini Advanced: $20/mies. (w tym 2TB Google Drive)	Gemini 3 Pro, 2M kontekst, integracja z Workspace
Copilot	GPT-4o mini, ale ograniczony	Copilot Pro: $30/mies.	Integracja z Office, priorytetowy dostęp

Dla kogo płatny plan? Sprawdź, czy wykupując subskrypcję, naprawdę potrzebujesz dostępu do najlepszego modelu. Jeśli używasz ChatGPT kilka razy w tygodniu do prostych pytań, darmowy plan w zupełności wystarczy. Jeśli pracujesz z AI codziennie, piszesz kod, analizujesz dokumenty – płatny plan zwróci się wielokrotnie w zaoszczędzonym czasie.

Co dzieje się z Twoimi danymi?

To najważniejsze pytanie, jakie powinieneś sobie zadać, korzystając z chmurowych usług AI. Gdy wpisujesz pytanie do ChatGPT, Twoje dane wędrują na serwery OpenAI w Stanach Zjednoczonych. To samo dotyczy Claude i Gemini.

NA CO UWAŻAĆ W CHMURZE

1. Dane mogą być użyte do treningu

Większość darmowych planów domyślnie używa Twoich rozmów do dalszego trenowania modeli. W płatnych planach możesz to wyłączyć w ustawieniach.

2. RODO a dane za granicą

Twoje dane opuszczają UE i trafiają do USA. Zgodnie z RODO, przekazywanie danych poza EEA wymaga odpowiednich zabezpieczeń prawnych. Wszyscy główni dostawcy (OpenAI, Google, Anthropic) deklarują zgodność z RODO, ale za zgodność płacisz w wersji Enterprise.

3. Nie podawaj danych wrażliwych

Nigdy nie wpisuj do chatu AI: haseł, numerów PESEL, danych medycznych, tajemnic handlowych ani poufnych informacji firmowych – chyba że masz gwarancję, że nie zostaną użyte do treningu.

4. Opcja opt-out

W ustawieniach każdej z usług możesz wyłączyć wykorzystywanie rozmów do trenowania. Warto to zrobić od razu po rejestracji.

POZIOM 2

Głębiej

API, ceny za tokeny, RODO i porównanie modeli – jak podejmować świadome decyzje.

API – jak firmy podłączają AI do swoich produktów

API (Application Programming Interface) to techniczna uszczelka między Twoim programem a modelem AI w chmurze. To dzięki API powstają wszystkie zewnętrzne integracje: asystent AI na stronie firmowej, chatbot w aplikacji bankowej, automatyczne tłumaczenie w sklepie internetowym.

Jak to działa? Twój program wysyła żądanie (request) na serwer dostawcy AI, zawierające tekst zapytania. Serwer przetwarza go na modelu, generuje odpowiedź i odsyła ją (response). Całość trwa zwykle od kilkuset milisekund do kilku sekund. Programista nie musi wiedzieć, jak działa model – po prostu wywołuje funkcję i dostaje wynik.

Każde wywołanie API jest płatne. Cena zależy od liczby tokenów wysłanych (input) i odebranych (output). Token to podstawowa jednostka rozliczeniowa – mniej więcej 3/4 słowa w języku angielskim. Dla języka polskiego stosunek jest bliższy 1:1, bo polskie słowa są dłuższe i bardziej złożone fleksyjnie.

Okno kontekstu, cena za token, limity – jak czytać specyfikację

Wybierając model przez API, patrzysz na trzy liczby:

Okno kontekstu

Maksymalna długość tekstu, jaką model może jednorazowo przetworzyć. Im większe, tym dłuższy dokument możesz wkleić. GPT-5.5 ma 1M tokenów, Claude Opus 4.7 ma 1M, Gemini 3.1 Pro ma 2M, a Llama 4 Scout aż 10M tokenów. 256K to ok. 400 stron tekstu, a 2M to cały „Potop" Sienkiewicza.

Cena za milion tokenów

Ceny podaje się za 1 milion tokenów wejściowych (input) i wyjściowych (output). W 2026 roku kształtują się mniej więcej tak:

Aktualne na: maj 2026. Ceny zmieniają się kwartalnie — przed wyborem sprawdź platformę.

Model	Input $/1M	Output $/1M	Kontekst
GPT-5.5	$5.00	$30.00	1M
Claude Opus 4.7	$5.00	$25.00	1M
Gemini 3.1 Pro	$2.00	$12.00	2M
Gemini 3.5 Flash	$0.075	$0.30	1M
Mistral Large 3	$0.50	$1.50	1M
DeepSeek V4-Pro	$1.74	$3.48	1M

Uwaga: ceny dynamicznie spadają (oznaczone jako [SZYBKO ZMIENNE]). Między 2023 a 2026 rokiem koszt przetworzenia miliona tokenów spadł około 12-krotnie. To efekt konkurencji, lepszej architektury (MoE) i tańszego sprzętu. Aktualne ceny zawsze sprawdzaj na stronie dostawcy.

RODO a dane wysyłane do chmur – prawo i ryzyko

Dla polskiego czytelnika kwestia danych osobowych jest istotna. RODO (odpowiednik europejskiego GDPR) nakazuje, by dane osobowe były przetwarzane w sposób bezpieczny i zgodny z prawem. Przekazanie danych do USA wymaga odpowiednich zabezpieczeń, np. standardowych klauzul umownych (SCC) lub decyzji o odpowiednim poziomie ochrony (tzw. Tarcza Prywatności).

RODO A AI – CO MUSISZ WIEDZIEĆ

Korzystając z ChatGPT, Claude czy Gemini do celów służbowych (np. analiza dokumentów z danymi klientów), jesteś administratorem danych i pełnisz odpowiedzialność za ich bezpieczeństwo. Oznacza to, że:

- Musisz sprawdzić, czy dostawca AI oferuje umowę DPA (Data Processing Agreement) – wersje Enterprise tak, darmowe zwykle nie.
- Musisz wyłączyć zapisywanie rozmów do treningu modelu (opt-out).
- Nie możesz przetwarzać danych wrażliwych (zdrowie, wyroki sądowe, dane biometryczne) bez wytycznych.
- Rozwiązaniem dla firm jest hosting modelu w UE (np. Azure OpenAI w regionie Europy, Mistral AI z siedzibą we Francji) lub modele lokalne.

Tabela decyzyjna: chmura czy lokalnie?

Kiedy wybrać chmurę, a kiedy postawić na własny sprzęt? Oto praktyczne kryteria:

Kryterium	Chmura	Lokalnie
Jakość odpowiedzi	Najwyższa (GPT-5.5, Claude Opus 4.7)	Niższa (najlepsze modele 70-90% jakości chmury)
Prywatność danych	Dane na serwerach w USA; ryzyko RODO	Dane nie opuszczają Twojego komputera
Koszt początkowy	Zero (darmowy plan) lub $20/mies.	Wymaga zakupu sprzętu (2 000-20 000 zł)
Łatwość uruchomienia	Gotowe od razu, otwierasz przeglądarkę	Wymaga konfiguracji (narzędzia, model, sprzęt)
Koszt przy dużym użyciu	Rośnie liniowo; $100-1000+/mies.	Stały, tylko prąd (kilkadziesiąt zł/mies.)
Offline	Wymaga internetu	Działa bez internetu

Rekomendacja: zacznij od chmury (darmowy plan), a gdy potrzebujesz prywatności lub masz dużą skalę – rozważ modele lokalne. Większość osób i firm stosuje podejście hybrydowe: codzienne zadania w chmurze, wrażliwe dane lokalnie.

→ W następnym rozdziale: **Lokalne modele językowe – co to jest, dlaczego warto i jakie są ograniczenia?**

Rozdział 4½. Jak rozmawiać z AI?

System prompt, few-shot, chain of thought, iteracja – cztery techniki, które diametralnie poprawią jakość odpowiedzi. Z przykładami przed i po.

Modele językowe to nie magiczne kule – jakość odpowiedzi zależy głównie od tego, jak sformułujesz pytanie. Umiejętność skutecznego formułowania poleceń to prompt engineering. Nie wymaga znajomości programowania, a diametralnie poprawia wyniki. W tym rozdziale poznasz cztery podstawowe techniki i od razu zobaczysz różnicę między złym a dobrym promptem.

1. System prompt – ustal kontekst na starcie

Zanim zaczniesz rozmowę, powiedz modelowi, kim ma być i jak ma odpowiadać. To instrukcja dla nowego pracownika – im precyzyjniejsza, tym lepszy efekt.

PRZED: bez system promptu

„Napisz maila do klienta."

Efekt: model zgaduje – może napisać maila formalnego, luźnego, agresywnego. Za każdym razem inaczej.

PO: z system promptem

„Jesteś asystentem w małej firmie IT. Odpowiadasz ciepło, ale profesjonalnie. Zawsze podpisuj: Z poważaniem, Zespół TechSupport."

Następnie: „Napisz maila do klienta, który zgłasza błąd w systemie." Efekt: każdy mail brzmi tak samo, jak był pisany przez tę samą osobę.

Dobre system prompty są konkretne: określają rolę, ton, format odpowiedzi i ewentualne ograniczenia. Przykłady:

- Jesteś ekspertem od finansów osobistych. Odpowiadaj językiem zrozumiałym dla laika. Zawsze podawaj konkretne przykłady.
- Tłumaczysz teksty z angielskiego na polski. Zachowaj naturalny styl, unikaj kalk językowych.

- Jesteś nauczycielem matematyki na poziomie szkoły średniej. Wyjaśniaj krok po kroku, nie pomijaj żadnego etapu.

W ChatGPT system prompt ustawiasz w Personalizacji (Custom instructions). W Claude – w projekcie. W lokalnych modelach dopisujesz go na początku każdej rozmowy.

2. Few-shot – pokaż przykłady

Zamiast nakazywać, jak ma działać, pokaż 2–3 przykłady. To najskuteczniejszy sposób na uzyskanie oczekiwanego formatu.

PRZED: bez przykładów

„Przetłumacz ten tekst na język formalny."

Efekt: model może dać wynik zbyt sztywny lub niekonsekwentny.

PO: z przykładami (few-shot)

Potocznie: „Hej, spotkajmy się jutro o 15" → Formalnie: „Szanowni Państwo, zapraszam na spotkanie jutro o godzinie 15:00."

Potocznie: „Daj znać, czy możesz" → Formalnie: „Proszę o potwierdzenie dostępności."

Potocznie: „OK, zrobione" → Formalnie:

Efekt: model kontynuuje wzorzec i sam zamienia ostatnie zdanie na formalne.

Few-shot sprawdza się szczególnie przy transformacji formatu (potoczny → formalny, lista → akapit, punkty → tabela), gdy potrzebujesz konkretnego szablonu odpowiedzi.

3. Chain of Thought – każ modelowi myśleć na głos

W zadaniach wymagających logiki, matematyki lub analizy, poproś o rozpisanie krokowego rozumowania. To radykalnie podnosi poprawność.

PRZED: bez CoT

„Ile to 15% z 3400 zł?"

Efekt: model może podać 340 zamiast 510 – nie widzisz, gdzie popełnił błąd.

PO: z CoT

„Ile to 15% z 3400 zł? Rozpisz krok po kroku."

Odpowiedź modelu: „10% z 3400 = 340 zł. 5% z 3400 = 170 zł. Razem 15% = 340 + 170 = 510 zł."

Efekt: widzisz tok rozumowania i możesz łatwo zweryfikować poprawność.

Chain of Thought działa jeszcze lepiej, gdy dodasz instrukcję: „Najpierw wypisz, co wiesz, potem zaplanuj rozwiązanie, następnie wykonaj obliczenia, na końcu podaj odpowiedź."

4. Iteracja – rozmowa, nie pojedyncze pytanie

Model zapamiętuje kontekst rozmowy. Wykorzystaj to: zamiast próbować zmieścić wszystko w jednym prompcie, prowadź dialog.

PRZED: wszystko w jednym prompcie

„Napisz artykuł o AI dla początkujących, w języku formalnym, z przykładami, podzielony na 3 sekcje, z wprowadzeniem i podsumowaniem, każda sekcja 300 słów."

Efekt: model przeciążony instrukcjami – zapomina o części wymagań.

PO: iteracja krok po kroku

1. „Napisz wstęp do artykułu o AI dla początkujących."
2. „Teraz dodaj sekcję o sieciach neuronowych, prostym językiem."
3. „Po tej sekcji dodaj przykład działania sieci."
4. „Na koniec napisz podsumowanie w 3 zdaniach."

Efekt: model skupia się na jednym zadaniu naraz, jakość każdej części jest wyższa.

Iteracja to klucz do dobrych rezultatów. Rzadko zdarza się, że pierwsza odpowiedź jest idealna. Traktuj model jak stażystę, którego prowadzisz do celu.

5. Częste błędy początkujących

- **Zbyt długi prompt** – model gubi szczegóły. Lepiej podzielić na kroki (iteracja).
- **Brak kontekstu** – pytanie bez tła wymaga od modelu zgadywania. Zawsze podawaj kontekst.

- **Jeden format na wszystko** – inaczej formułujesz pytanie do analizy, inaczej do kreatywnego pisania. Dostosuj technikę do zadania.
- **Nieiterowanie** – pierwsza odpowiedź rzadko jest idealna. Poprawiaj, doprecyzowuj, pytaj o szczegóły.

6. Ćwiczenie praktyczne

Weź jeden prompt, którego używasz na co dzień (np. „Napisz maila" lub „Wyjaśnij X") i przepisz go kolejno:

- Dodaj system prompt – kim ma być model.
- Dodaj 1–2 przykłady oczekiwanego wyjścia.
- Poproś o rozpisanie krokowego rozumowania (jeśli zadanie tego wymaga).
- Iteruj: najpierw główna część, potem poprawki.

Po każdym kroku porównaj odpowiedzi. Różnica będzie zaskakująca.

Podsumowanie technik

Technika	Kiedy stosować	Efekt
System prompt	Przed każdą rozmową o nowym temacie	Model wie, czego od niego oczekujesz
Few-shot (przykłady)	Gdy potrzebujesz konkretnego formatu	Odpowiedź od razu we właściwej formie
Chain of Thought	Zadania logiczne, matematyka, analiza	Znacznie wyższa poprawność
Iteracja	Zawsze, gdy pierwszy wynik nie jest idealny	Stopniowe ulepszanie rezultatu

Poświęć 15 minut na świadome testowanie tych technik – to najlepsza inwestycja czasu w lepsze korzystanie z AI.

Rozdział 5. Lokalne modele językowe

Wyobraź sobie, że ChatGPT działa na Twoim komputerze – bez internetu, bez subskrypcji i bez wysyłania danych do USA. Brzmi jak science fiction? W 2026 roku to rzeczywistość, która jest dostępna dla każdego.

POZIOM 1

Dla laika

Dowiesz się, czym są lokalne modele językowe, dlaczego warto je uruchamiać na własnym komputerze i jak zrobić pierwszy krok.

Co to są lokalne modele językowe?

Modele językowe, takie jak GPT-5.5 czy Claude Opus 4.7, kojarzysz głównie jako usługi działające w chmurze – otwierasz przeglądarkę, wpisujesz pytanie, dostajesz odpowiedź. Lokalne modele językowe działają dokładnie tak samo, ale uruchamiasz je na własnym komputerze, a nie na serwerach w USA czy Europie.

Różnica jest fundamentalna: zamiast wysyłać pytanie przez internet i czekać na odpowiedź z serwera, całe przetwarzanie odbywa się na Twojej maszynie. Model działa w izolacji, nie wymaga dostępu do sieci, a Twoje dane nigdy nie opuszczają dysku twardego.

Dopiero ok. 2023 roku stało się to praktyczne dzięki technice zwanej kwantyzacją (więcej o niej w części zaawansowanej) oraz optymalizacji kodu w projekcie llama.cpp. W 2026 roku jakość lokalnych modeli dogania chmurowe odpowiedniki w wielu codziennych zastosowaniach.

Dlaczego warto uruchamiać AI lokalnie?

Korzyści płynące z lokalnego AI są tak przekonujące, że według badań z 2026 roku ponad 42% programistów regularnie używa modeli na własnym sprzęcie. Oto najważniejsze powody:

Prywatność i bezpieczeństwo danych

Twoje dane nigdy nie opuszczają komputera. To istotna zaleta dla firm, które przetwarzają dane wrażliwe: dane medyczne (zgodność z HIPAA), dane osobowe klientów (RODO/GDPR), tajemnice handlowe czy dokumenty prawne. W chmurze zawsze istnieje ryzyko, że dane zostaną wykorzystane do trenowania modelu lub dostaną się w niepowołane ręce. Lokalnie to ryzyko znika.

Brak kosztów subskrypcyjnych

Kupujesz sprzęt raz i używasz go za darmo. Chmurowe API płaci się za każdy token – przy intensywnym użyciu miesięczny rachunek może sięgnąć setek dolarów. Lokalnie płacisz tylko za prąd (kilkadziesiąt złotych miesięcznie) i amortyzację sprzętu. W dłuższej perspektywie to ogromna oszczędność.

Działa offline i bez limitów

Nie potrzebujesz internetu. Możesz używać AI w samolocie, w podróży, w miejscach bez zasięgu. Nie ma też ograniczeń liczby zapytań – możesz rozmawiać z modelem cały dzień bez obaw, że wyczerpiesz limit i będziesz musiał czekać do następnego dnia.

Brak cenzury i ograniczeń tematycznych

Modele chmurowe mają filtry bezpieczeństwa, które blokują niektóre tematy. Lokalnie to Ty decydujesz, jakie treści chcesz generować. Oczywiście z odpowiedzialnością i zgodnie z prawem.

Niskie opóźnienie (latencja)

Odpowiedź pojawia się natychmiast – nie ma narzutu sieciowego. Przy modelach mieszczących się w pamięci GPU czas do pierwszej odpowiedzi to milisekundy, nie sekundy. To kluczowe w zastosowaniach czasu rzeczywistego.

Jakie narzędzia są potrzebne?

Na szczęście nie musisz być programistą, żeby uruchomić lokalny model. W 2026 roku istnieje kilka prostych narzędzi:

- **Ollama** – najprostszy sposób. Działa jak „brew install" dla AI: instalujesz, wpisujesz komendę i model działa. Ma wbudowany serwer API zgodny z OpenAI. Wersja na Windows, macOS i Linux.
- **LM Studio** – aplikacja z graficznym interfejsem. Ściągasz, wybierasz model z listy, klikasz „Uruchom" i gotowe. Idealne dla osób, które nie lubią terminala.
- **GPT4All** – jeszcze prostsze. Instalujesz jak każdą inną aplikację, wybierasz model i rozmawiasz. Dla totalnych początkujących.
- **Jan** – otwartoźródłowa alternatywa z czystym interfejsem. Działa w 100% offline, wspiera RAG (dokumenty).

Przewodnik krok po kroku: Jak uruchomić pierwszy lokalny model w 5 minut

Najszybszą drogą do własnego AI lokalnie jest Ollama. Wykonaj te cztery proste kroki:

Krok 1: Pobierz i zainstaluj Ollamę. Wejdź na stronę ollama.com i pobierz instalator dla Windows (lub macOS/Linux). Uruchom plik – instalacja trwa kilkanaście sekund.

Krok 2: Otwórz terminal. W Windows: naciśnij Win+R, wpisz cmd i Enter. Możesz też użyć PowerShell.

Krok 3: Wpisz komendę: ollama run llama5. System ściągnie model (ok. 4-5 GB przy pierwszym uruchomieniu) i automatycznie go uruchomi.

Krok 4: Rozmawiaj z modelem. Gdy zobaczysz znak zapytania ?, możesz już pisać pytania. Aby zakończyć, wpisz /bye.

GOTOWE! Właśnie uruchomiłeś swój pierwszy lokalny model językowy.

Możesz teraz rozmawiać z AI bez internetu, za darmo i z pełną prywatnością. żeby wypróbować inne modele, wystarczy zmienić nazwę: ollama run llama5, ollama run gemma4, ollama run qwen3.6.

POZIOM 2

Głębiej

Jak działają silniki inferencyjne, porównanie narzędzi, kwantyzacja, benchmarki i wybór sprzętu.

Jak to działa? Architektura lokalnego AI

Większość narzędzi do lokalnego uruchamiania modeli (Ollama, LM Studio, GPT4All, Jan) opiera się na tym samym silniku – projekcie llama.cpp. To napisana w C++ biblioteka, która optymalizuje uruchamianie modeli językowych na zwykłym sprzęcie.

llama.cpp obsługuje akcelerację GPU przez CUDA (NVIDIA), Metal (Apple), ROCm (AMD) i Vulkan. Dzięki niemu modele, które wymagają 140 GB w formacie FP16, po kompresji (kwantyzacji) mieszczą się w 8-24 GB.

RELACJA MIĘDZY NARZĘDZIAMI

Wyobraź sobie stos: na dole jest llama.cpp – silnik samochodu. Ollama to karoseria z wygodnym sterowaniem. LM Studio to ten sam silnik, ale w luksusowej kabinie z dotykowym ekranem. Różnią się wyglądem i wygodą, ale pod spodem to ta sama technologia.

Porównanie narzędzi do lokalnej inferencji

Wybór narzędzia zależy od Twoich potrzeb. Oto zestawienie najważniejszych opcji w 2026 roku:

Narzędzie	Interfejs	Otwarty kod	API OpenAI	Dla kogo
llama.cpp	CLI + API	Tak (MIT)	Opcjonalnie	Developerzy, maks. wydajność
Ollama	CLI + REST API	Tak (MIT)	Tak (wbudowane)	Większość użytkowników
LM Studio	GUI + API	Nie	Tak (opcjonalnie)	Początkujący, GUI

Narzędzie	Interfejs	Otwarty kod	API OpenAI	Dla kogo
GPT4All	GUI + SDK	Tak (MIT)	Nie	Całkiem początkujący
Jan	GUI + API beta	Tak	Tak	Prywatność, RAG

Benchmarki wydajności na identycznym sprzęcie

Niezależne testy z 2025-2026 roku na MacBooku Pro M3 Max (64 GB RAM) z modelem Llama 5 8B (kwantyzacja Q4_K_M) pokazują wyraźną hierarchię wydajności:

Narzędzie	Średnio tok./s	Czas do 1. tokena	Pamięć RAM
llama.cpp (CLI)	**62,4 t/s**	180 ms	5,2 GB
MLX (Apple)	58,7 t/s	**95 ms**	4,8 GB
Ollama	51,3 t/s	240 ms	5,6 GB
LM Studio	48,9 t/s	310 ms	5,8 GB
GPT4All	38,4 t/s	420 ms	6,3 GB

Wniosek: różnica między najszybszym (llama.cpp) a najwolniejszym (GPT4All) to ok. 60%. Jednak dla codziennego użytku nawet 38 t/s jest w pełni wystarczające – odpowiada mniej więcej szybkości czytania przez człowieka.

Kwantyzacja – jak model 140 GB mieści się w 8 GB

To jedno z najważniejszych pojęć, które warto zrozumieć. Modele językowe są trenowane w precyzji 16-bitowej (FP16) lub 32-bitowej (FP32). Każda liczba waży 2 lub 4 bajty. Dla modelu z 70 miliardami parametrów daje to 140 GB.

Kwantyzacja polega na zmniejszeniu precyzji liczb – zamiast 16 bitów używamy 8, 6, 4 lub nawet 2 bitów. To tak, jakbyś zamiast mierzyć coś z dokładnością do 0,0001 mm, wystarczyła Ci dokładność do 1 mm. Traci się szczegóły, ale ogólny obraz zostaje.

Kwantyzacja 4-bitowa (Q4_K_M) zmniejsza rozmiar modelu o ok. 75% przy stracie jakości zaledwie 2-5%. To dlatego duże modele działają na sprzęcie konsumenckim.

Format	Bity na parametr	Model 70B	Jakość
FP16	16	~140 GB	Oryginalna
Q8_0	8	~70 GB	Prawie bez strat
Q4_K_M	4,5	~40 GB	**Dobra / zalecana**
Q3_K_S	3,5	~30 GB	Wyraźnala degradacja
Q2_K	2	~17 GB	Niska / tylko awaryjnie

Standardem w 2026 roku jest format GGUF (GPT-Generated Unified Format), który łączy wagę modelu, kod kwantyzacji i tokenizer w jednym pliku. To dzięki niemu wystarczy pobrać jeden plik, zamiast całego katalogu.

Wymagania sprzętowe – jaki komputer jest potrzebny?

To zależy od tego, jak duży model chcesz uruchomić. Oto praktyczny przewodnik:

Kategoria	Model	Min. pamięć	Zalecane GPU	Szybkość (t/s)
Budżetowy laptop	3-8B (Q4)	8 GB RAM	Brak (CPU)	5-15 t/s
Standardowy PC	7-14B (Q4)	16 GB RAM	RTX 3060 12GB	30-50 t/s
Wydajny PC	32B (Q4)	32 GB RAM	RTX 4090 24GB	15-25 t/s
MacBook Pro M4	7-70B (MLX)	16-128 GB unifik.	Wbudowane GPU	20-60 t/s

Wskazówka: jeśli nie masz karty graficznej, nadal możesz używać lokalnych modeli. Procesor poradzi sobie z modelami 7-8B, choć wolniej (5-15 tokenów na sekundę). Dla modeli 32B+ potrzebujesz albo bardzo dużej pamięci RAM (32+ GB), albo karty graficznej z odpowiednią ilością VRAM.

API i integracje – jak podłączyć lokalny model do innych narzędzi

Ollama udostępnia lokalny serwer API na porcie 11434, w pełni zgodny z API OpenAI. Dzięki temu możesz zastąpić chmurowy model lokalnym, zmieniając tylko jeden adres URL w kodzie.

Przykład w Pythonie:

```
from openai import OpenAI
client = OpenAI(
 base_url='http://localhost:11434/v1',
 api_key='ollama'
)
response = client.chat.completions.create(
 model='llama3.2',
 messages=[{'role':'user','content':'Hello!'}]
)
```

Dzięki temu lokalne modele działają z narzędziami takimi jak:

- Continue.dev – asystent kodowania w Visual Studio Code i JetBrains
- Open WebUI – interfejs czatu wzorowany na ChatGPT, działający w przeglądarce
- n8n, LangChain, LlamaIndex – platformy do automatyzacji i agentów AI
- SillyTavern – role-play i rozrywka z AI
- Aplikacje własne – możesz napisać aplikację w Pythonie, JS lub dowolnym języku

Polskie modele lokalne

Dla polskojęzycznych użytkowników szczególnie istotne jest, że istnieją modele trenowane na polskich danych. Dwa główne projekty to:

Bielik (SpeakLeash)

Bielik to polski model językowy, którego kolejne wersje (11B, 7B) są dostępne do użytku lokalnego. Działają przez Ollamę (ollama run bielik) i LM Studio. Rozumie język polski lepiej niż modele angielskojęzyczne, szczególnie w kontekście polskich realiów, prawa i kultury.

PLLuM (NASK)

PLLuM (Polish Large Language Model from scratch) to model rozwijany przez NASK – państwowy instytut badawczy. Został zaprezentowany w lutym 2025 roku. Jest trenowany od podstaw na polskich danych, co oznacza, że rozumie język polski na poziomie znacznie wyższym niż modele międzynarodowe. Wersje PLLuM są dostępne jako otwartoźródłowe i działają lokalnie.

Dla osób, które potrzebują AI w języku polskim do pracy z dokumentami, księgowością, prawem czy administracją, polskie modele są zdecydowanie lepszym wyborem niż angielskojęzyczne odpowiedniki.

Bezpieczeństwo lokalnych modeli

Uruchamianie modeli lokalnie ma też swoje wyzwania. Oto najważniejsze:

Model może być „zatruty"

Pobierając model z nieznanego źródła, ryzykujesz, że ktoś umieścił w nim złośliwy kod. Format GGUF teoretycznie nie pozwala na wykonanie kodu (w przeciwieństwie do Pickle), ale zawsze pobieraj modele z zaufanych źródeł: oficjalnego rejestru Ollama, Hugging Face od zaufanych autorów.

Brak filtrów bezpieczeństwa

Modele chmurowe mają wbudowane filtry blokujące szkodliwe treści. Lokalnie tych filtrów nie ma. To może być zaleta (wolność) lub wada (łatwiej wygenerować coś niebezpiecznego). Odpowiedzialność spoczywa na Tobie.

Konieczność aktualizacji

Modele lokalne nie aktualizują się automatycznie. Jeśli chcesz korzystać z nowszej wiedzy, musisz ściągnąć nowszą wersję modelu lub uzupełnić go o RAG (indexowanie własnych dokumentów).

Kiedy lokalnie, a kiedy chmura? Tabela decyzyjna

Optymalna strategia w 2026 roku to podejście hybrydowe. Oto kiedy wybrać które środowisko:

Zastosowanie	Lokalnie	Chmura
Prywatność danych	Dane na Twoim komputerze	Dane wysyłane do USA
Najwyższa jakość	70-95% jakości chmury	(+) Najlepsze modele (GPT-5.5, Claude Opus 4.7, DeepSeek V4-Pro)
Koszt przy dużym użyciu	(+) Tylko prąd (~30-80 zł/mies.)	(-) Rosnie z użyciem
Działanie offline	(+) Pełna niezależność	(-) Wymaga internetu
Łatwość startu	Instalacja narzędzia + ściągnięcie modelu	(+) Otwierasz przeglądarkę
Język polski	(+) Bielik, PLLuM	Też dobry (GPT, Claude)

Przyszłość lokalnych modeli

Rynek lokalnych modeli rozwija się błyskawicznie. W 2024 roku modele lokalne były ciekawostką dla entuzjastów. W 2026 roku są realną alternatywą dla chmury, używaną przez ponad 40% programistów. Trendy na najbliższe lata:

- Modele specyficzne dla branż (medycyna, prawo, finanse) działające lokalnie
- Coraz lepsze kwantyzacje – IQ4_NL, Q3_K_XL minimalizują straty jakości
- Większe okna kontekstu (128K+ standardem nawet lokalnie)
- Wbudowane AI w systemach operacyjnych (Apple Intelligence, Windows AI Copilot+)
- Modele multimodalne lokalnie – widzenie, mowa, generowanie obrazów

Eksperci przewidują, że do 2028 roku lokalne modele osiągną jakość dorównującą chmurowym w większości standardowych zastosowań, a granica między „chmurą" a „lokalnie" będzie płynna.

→ W następnym rozdziale: **Formaty i kwantyzacja modeli – GGUF, GPTQ, AWQ, MLX i jak wybrać odpowiedni dla siebie?**

Rozdział 6. Formaty i kwantyzacja modeli

Plik GGUF, kwantyzacja Q4_K_M, format AWQ – co to wszystko znaczy i dlaczego ma ogromne znaczenie dla działania AI na Twoim komputerze?

POZIOM 1

Dla laika

Zrozumiesz, czym są formaty modeli, co to jest kwantyzacja i jak różne poziomy kompresji wpływają na jakość i szybkość działania AI.

Dlaczego istnieje tyle różnych formatów?

Gdy w poprzednim rozdziale ściągałeś model przez Ollamę, prawdopodobnie nie zastanawiałeś się nad formatem pliku. Po prostu wpisałeś komendę i model działał. Ale pod spodem kryje się cała nauka o tym, jak przechowywać i kompresować modele, by działały na różnym sprzęcie.

Wyobraź sobie, że kupujesz muzykę. Możesz ją dostać jako:

- plik WAV – idealna jakość, ale jeden utwór zajmuje 50 MB
- plik FLAC – bezstratna kompresja, 25 MB
- plik MP3 (320 kbps) – bardzo dobra jakość, 10 MB
- plik MP3 (128 kbps) – wystarczająca jakość, 4 MB

Z modelami AI jest podobnie. Format pliku określa, jak dane są zapisane, a kwantyzacja decyduje o stopniu kompresji. Każdy format i poziom kompresji to kompromis między jakością a rozmiarem.

Co to jest kwantyzacja?

Kwantyzacja to skomplikowana nazwa na prostą rzecz: zmniejszanie precyzji liczb, które tworzą model. Model językowy to miliardy liczb (wag). Każda z nich to ułamek, np. 0,0038472 lub -1,4567. Im dokładniej zapiszesz te liczby, tym więcej miejsca zajmują.

To trochę jak z zapisywaniem temperatury:

- „26,3742°C" jest bardzo dokładne, ale wymaga wielu znaków
- „26°C" traci szczegóły, ale w większości sytuacji w zupełności wystarcza

Kwantyzacja robi to samo z wagami modelu. Co więcej, sieci neuronowe są zaskakująco odporne na utratę precyzji. Możesz skompresować model o 75% (z 16 bitów do 4 bitów), tracąc zaledwie 1-5% jakości odpowiedzi.

KWANTYZACJA JAK JPEG

Surowy plik RAW z aparatu (16-bitowy) możesz skompresować do JPEG (8-bitowy). Zdjęcie nadal wygląda świetnie, ale zajmuje 4 razy mniej miejsca. Kwantyzacja w AI działa identycznie – kompresujesz wagę modelu, zachowując większość jakości.

Który format wybrać?

Jeśli dopiero zaczynasz, nie musisz się tym przejmować. Używając Ollamy lub LM Studio, wszystko dzieje się automatycznie. Ale gdy zechcesz pobrać model z Hugging Face, zobaczysz mnóstwo wersji: gguf, gptq, awq, mlx, safetensors. Oto najprostszy przewodnik:

- Jeśli używasz Ollamy lub LM Studio – wybieraj format GGUF (to standard, który działa wszędzie)
- Jeśli masz komputer Apple z chipem M1/M2/M3/M4 – możesz też wybrać MLX (szybszy na Macu)
- Jeśli masz wydajną kartę NVIDIA i wdrażasz model na serwerze – wybierz AWQ lub GPTQ
- Jeśli chcesz trenować lub dostrajać model – potrzebujesz formatu safetensors (Hugging Face)

ZŁOTA ZASADA NA START

Zawsze zaczynaj od GGUF z kwantyzacją Q4_K_M. To uniwersalny standard, który działa na każdym sprzęcie i daje optymalny balans jakości i rozmiaru. Dopiero gdy poznasz swój sprzęt i potrzeby, eksperymentuj z innymi formatami.

POZIOM 2

Głębiej

Szczegółowe porównanie formatów (GGUF, GPTQ, AWQ, MLX), K-quanty, I-quanty, benchmarki i praktyczne wskazówki wyboru.

Precyzja liczb – fundament zrozumienia

Każda waga w modelu to liczba. To, jak bardzo jest precyzyjna, określa typ zmiennoprzecinkowy. Oto najważniejsze:

Typ	Bajtów	Zakres wartości	Zastosowanie	Model 7B
FP32	4	Największy	Trenowanie, research	28 GB
FP16 / BF16	2	Standard treningu	Trenowanie, baseline	14 GB
INT8 (Q8_0)	1	Ograniczony (256)	Near-lossless	7 GB
INT4 (Q4_K_M)	~0,5	16 wartości	Standard lokalny	**~4,5 GB**

Gdzie leży granica? W 2026 roku praktyczne minimum to 4 bity na wagę (Q4). Poniżej 3 bitów (Q2) jakość spada dramatycznie. Trwają prace nad kwantyzacją 1,58-bitową (BitNet), która może zrewolucjonizować lokalne AI.

Format GGUF – uniwersalny standard lokalny

GGUF (GPT-Generated Unified Format) to format pliku stworzony przez projekt llama.cpp. Jest następcą starszego GGML i w 2026 roku stanowi dominujący standard dla lokalnej inferencji. Jego główne zalety:

- Pojedynczy plik – wszystkie wagi, tokenizer, szablon czatu i metadane w jednym pliku
- Działa wszędzie – na CPU, GPU NVIDIA, AMD, Intel, Apple Silicon, a nawet na Raspberry Pi
- Pełne wsparcie dla kwantyzacji – od Q2 do Q8 z wieloma wariantami

- Podzielony na bloki – model może działać częściowo na GPU, częściowo na CPU, gdy nie mieści się w VRAM
- Obsługiwany przez: Ollamę, LM Studio, GPT4All, Jan, kobold.cpp, text-generation-webui

GGUF jest przeznaczony wyłącznie do inferencji (uruchamiania) – nie można go użyć do trenowania. Do trenowania potrzebny jest oryginalny format safetensors z Hugging Face.

Format GPTQ – GPU-optimized dla NVIDIA

GPTQ (Generative Pre-Trained Transformer Quantization) to starszy format opracowany w 2022 roku. W przeciwieństwie do GGUF, GPTQ nie jest pojedynczym plikiem, ale katalogiem z wieloma plikami.

- Optymalizowany pod kątem GPU NVIDIA – używa wydajnych jąder CUDA
- Wymaga kalibracji – podczas kwantyzacji przepuszcza próbkę danych przez model, by określić, które wagi są najważniejsze
- Dostępny w wariantach 8-bit (INT8), 4-bit (INT4) i 3-bit (INT3)
- Obsługiwany przez: vLLM, AutoGPTQ, ExLlama, text-generation-webui
- NIE działa na CPU ani Apple Silicon – wymaga CUDA

GPTQ jest dziś w dużej mierze wypierany przez AWQ, który oferuje lepszą jakość przy tym samym rozmiarze. Jednak ogromna biblioteka prekwantyzowanych modeli GPTQ na Hugging Face wciąż czyni go użytecznym dla serwerów NVIDIA.

Format AWQ – nowoczesna kwantyzacja aktywacyjna

AWQ (Activation-Aware Weight Quantization) to najnowszy format GPU, opracowany w 2023 roku przez MIT. Jego innowacja polega na analizie wzorców aktywacji, a nie tylko samych wag. Nie wszystkie wagi są równie ważne – AWQ identyfikuje te najistotniejsze i zachowuje je w wyższej precyzji.

- O 0,5-1% lepsza perplexity niż GPTQ przy tej samej liczbie bitów
- Wydajne jądro Marlin – osiąga nawet 741 tok./s na vLLM
- Wymaga kalibracji na reprezentatywnym zbiorze danych
- NIE działa na CPU ani Apple Silicon – wymaga CUDA

AWQ jest najlepszym wyborem dla serwerów z kartami NVIDIA. Jeśli wdrażasz model produkcyjnie na GPU, wybierz AWQ. Do użytku lokalnego na różnym sprzęcie zostań przy GGUF.

Format MLX – natywny dla Apple Silicon

MLX to framework uczenia maszynowego od Apple, zaprojektowany specjalnie dla chipów M1/M2/M3/M4. Wykorzystuje unifikowaną pamięć (Unified Memory) Apple, co eliminuje potrzebę kopiowania danych między RAM a VRAM.

- Nawet 20-30% bardziej wydajny pamięciowo niż GGUF na tym samym Macu
- Działa tylko na macOS z Apple Silicon
- Obsługiwany przez: mlx-lm, Ollama (od wersji 0.19+), LM Studio
- Własny format oparty na safetensors, ale może odczytywać wybrane kwantyzacje GGUF

Jeśli masz MacBooka z M3/M4, warto rozważyć MLX – szczególnie do modeli, które ledwo mieszczą się w pamięci. Dla użytkowników Windows i Linux MLX nie wchodzi w grę.

Bitsandbytes i QLoRA – trenowanie na skwantyzowanych modelach

Bitsandbytes to biblioteka, która umożliwia ładowanie modeli w 4-bitach (NF4) i 8-bitach przy użyciu zaledwie kilku linijek kodu. Jej największą zaletą jest to, że jako jedyna pozwala na trenowanie (fine-tuning) modelu bez przywracania pełnej precyzji.

Dzięki technice QLoRA (Quantized Low-Rank Adaptation) możesz dostroić model 70B na pojedynczej karcie RTX 4090 z 24 GB VRAM. To była rewolucja w 2023 roku i w 2026 nadal jest to standardowa metoda dostrajania na ograniczonym sprzęcie.

- NF4 (Normal Float 4) – specjalny format kwantyzacji 4-bitowej zaprojektowany dla QLoRA
- Unikatowa cecha: można trenować na skwantyzowanym modelu bez utraty jakości
- Obsługiwany przez: Hugging Face Transformers, PEFT, Unsloth

- NIE jest przeznaczony do szybkiej inferencji – wolniejszy niż GGUF czy AWQ

[UWAGA] WAŻNE ROZRÓŻNIENIE

GGUF/GPTQ/AWQ/MLX są przeznaczone do uruchamiania (inferencji). Bitsandbytes/QLoRA są przeznaczone do dostrajania (fine-tuningu). To dwie różne kategorie narzędzi, choć obie używają kwantyzacji.

K-quanty – nowoczesna kwantyzacja GGUF

K-quanty (K-Quantization) to seria zaawansowanych metod kwantyzacji wbudowanych w format GGUF. Ich nazewnictwo podlega ścisłej konwencji: Q{liczba_bitów}_{wariant}. Kluczowe warianty:

Oznaczenie	Bity/wagę	Rozmiar modelu 8B	Utrata jakości	Kiedy użyć?
Q8_0	8,0	~8 GB	0,2% (prawie bezstratna)	Masz dużo RAM/VRAM
Q6_K	6,5	~6,5 GB	0,6%	Jakość premium
Q5_K_M	5,5	~5,5 GB	**1,2% – złoty środek**	Masz 16+ GB RAM
Q4_K_M	4,5	~4,5 GB	**1,9% – zalecany domyślny**	Większość użytkowników
Q3_K_M	3,5	~3,5 GB	5-8% – wyraźna degradacja	Tylko gdy mało RAM
Q2_K	2,5	~2,5 GB	10-15% – awaryjna	Tylko awaryjnie

Różnica między Q4_K_M a Q8_0 w codziennym użyciu (czat, proste pytania, pomoc w kodzie) jest niezauważalna. Dopiero przy skomplikowanych zadaniach (zaawansowana matematyka, długie wnioskowania, precyzyjne generowanie kodu) widać różnicę.

I-quanty – agresywna kompresja dla ograniczonego sprzętu

I-quanty (Importance-Quantization) to nowsza rodzina kwantyzacji w GGUF, zaprojektowana dla jeszcze większej kompresji. Wykorzystują macierz ważności (importance matrix), która określa, które wagi są krytyczne dla jakości, a które można skompresować bardziej agresywnie.

- IQ4_NL (Non-Linear 4-bit) – lepszy niż Q4_K_M przy tych samych 4 bitach
- IQ3_XXS (Extra-Extra-Small 3-bit) – wyśmienita jakość jak na 3 bity
- IQ2_XXS – najmniejszy możliwy rozmiar, ale wyczuwalna degradacja
- Wymagają więcej czasu na kwantyzację i mają nieco wolniejszą inferencję

PORADA PRAKTYCZNA

Kiedy warto użyć I-quant? Gdy chcesz uruchomić model, który nie mieści się w RAM nawet w Q4_K_M. Na przykład model 70B w Q4_K_M wymaga ~40 GB, ale w IQ2_XXS ~17 GB. Jakość spada, ale nadal jest użyteczna.

Porównanie formatów – tabele zbiorcze

Zestawienie wszystkich formatów

Benchmarki aktualne na: maj 2026. Nowe formaty pojawiają się co kilka miesięcy.

Format	CPU	NVIDIA	Apple	Trenowanie	Zastosowanie
GGUF	**TAK**	**TAK**	**TAK**	**NIE**	Uniwersalnie lokalnie
GPTQ	**NIE**	**TAK**	**NIE**	TAK (Adapter)	Serwery NVIDIA
AWQ	**NIE**	**TAK**	**NIE**	TAK (Adapter)	Nowoczesne GPU
MLX	**NIE**	**NIE**	**TAK**	**NIE**	Mac tylko
Bitsandbytes	**TAK**	**TAK**	**TAK**	**TAK (QLoRA)**	Fine-tuning tylko

Benchmarki jakości kwantyzacji (perplexity)

Poniższe dane pochodzą z niezależnych testów na zbiorze Wikitext-2 (maj 2026). Niższa wartość perplexity = lepsza jakość. W nawiasie procent degradacji względem oryginalnego FP16.

Benchmarki aktualne na: maj 2026. Wyniki różnią się w zależności od sprzętu i konfiguracji.

Kwantyzacja	Llama 5 8B	Qwen 3.7-Max 32B	Llama 5 70B	Uwagi
FP16	**5,21**	**4,78**	**3,42**	Referencja
Q8_0	5,22 (+0,2%)	4,79 (+0,2%)	3,42 (+0,0%)	Near-lossless
Q5_K_M	5,27 (+1,2%)	4,83 (+1,0%)	3,45 (+0,9%)	**Złoty środek**
Q4_K_M	5,31 (+1,9%)	4,87 (+1,9%)	3,47 (+1,5%)	**Zalecany domyślny**
AWQ 4-bit	5,30 (+1,7%)	4,85 (+1,5%)	3,46 (+1,2%)	Najlepszy na GPU
GPTQ 4-bit	5,34 (+2,5%)	4,89 (+2,3%)	3,49 (+2,0%)	Starsza technologia
Q3_K_M	5,46 (+4,8%)	5,02 (+5,0%)	3,55 (+3,8%)	Wyraźna degradacja
Q2_K	5,89 (+13%)	5,45 (+14%)	3,78 (+11%)	Tylko awaryjnie

Wniosek z benchmarków: różnica między Q4_K_M a AWQ 4-bit jest marginalna (ok. 0,2-0,4%). Wybór formatu powinien być podyktowany głównie kompatybilnością ze sprzętem i narzędziami, nie jakością.

Wymagania pamięciowe dla różnych kwantyzacji

Kluczowy wzór do zapamiętania: pamięć w GB = (liczba parametrów × bity na wagę) / 8. Należy dodać 1-2 GB na bufor kontekstu (KV cache) i system.

Model	FP16	Q8_0	Q5_K_M	Q4_K_M	Q3_K_M
3B	6 GB	3,2 GB	2,1 GB	**1,8 GB**	1,4 GB
7-8B	15 GB	8 GB	5,3 GB	**4,6 GB**	3,5 GB
14B	26 GB	14 GB	9,3 GB	8 GB	6,2 GB
32B	60 GB	32 GB	21 GB	18 GB	14 GB
70B	131 GB	69 GB	46 GB	40 GB	31 GB
120B	225 GB	118 GB	79 GB	69 GB	53 GB

Zalecenie: model powinien zajmować nie więcej niż 60-70% dostępnej pamięci, aby zostało miejsce na system operacyjny, bufor kontekstu i inne aplikacje.

Jak wybrać odpowiedni format i kwantyzację?

Oto algorytm decyzyjny oparty na Twoim sprzęcie i potrzebach:

Krok 1: Określ swój sprzęt

- 8 GB RAM – modele 3B w Q4_K_M lub 8B w Q3_K_M
- 16 GB RAM – modele 8B w Q5_K_M, 14B w Q4_K_M
- 24 GB VRAM (RTX 4090) – modele 32B w Q4_K_M lub 14B w Q8_0
- 32-64 GB RAM – modele 32B w Q5_K_M, 70B w Q4_K_M
- 128 GB RAM (Mac) – modele 70B w Q8_0 lub większe

Krok 2: Wybierz format

- Używasz Ollamy/LM Studio na Windows/Linux? → GGUF
- Masz Maca z Apple Silicon? → GGUF (lub MLX dla maks. wydajności)
- Wdrażasz na serwerze z NVIDIA? → AWQ (lub GPTQ jak są gotowe modele)
- Chcesz dostroić model? → safetensors + bitsandbytes (NF4)

Krok 3: Wybierz kwantyzację

- Wolne miejsce i chcesz jakości? → Q8_0 lub Q6_K
- Standardowy wybór (złoty środek)? → Q5_K_M
- Oszczędzasz pamięć? → Q4_K_M (najlepszy stosunek jakości do rozmiaru)
- Bardzo mało pamięci? → Q3_K_M lub IQ3_XXS (awaryjnie)

Jak samodzielnie skwantyzować model?

Jeśli chcesz skwantyzować własny model (np. pobrany z Hugging Face w formacie safetensors), potrzebujesz narzędzia llama.cpp i jego podkomendy llama-quantize. Proces wygląda następująco:

1. Pobierasz oryginalny model w formacie Hugging Face (safetensors)

2. Konwertujesz go do GGUF (zachowując pełną precyzję FP16) za pomocą skryptu convert.py z repozytorium llama.cpp

3. Uruchamiasz kwantyzację: ./llama-quantize model.gguf q4_k_m

4. Otrzymujesz skwantyzowany plik GGUF gotowy do użycia w Ollamie, LM Studio lub bezpośrednio w llama.cpp

W przypadku AWQ lub GPTQ proces wygląda podobnie, ale używasz narzędzi AutoAWQ lub AutoGPTQ. Kwantyzacja wymaga karty NVIDIA z przynajmniej tyle samo VRAM, ile wagi modelu w FP16 (np. dla modelu 8B potrzeba ~16 GB VRAM).

[UWAGA] CZAS KWANTYZACJI

Kwantyzacja modelu 70B na RTX 4090 zajmuje od 2 do 12 godzin, w zależności od wybranej metody i kalibracji. Dla modeli 8B to kwestia 15-60 minut. Warto skorzystać z gotowych prekwantyzowanych modeli na Hugging Face.

Przyszłość kwantyzacji

Kwantyzacja to jedna z najszybciej rozwijających się dziedzin AI. Oto trendy na najbliższe lata:

BitNet i kwantyzacja 1,58-bitowa

Najbardziej obiecujący kierunek. Zamiast przechowywać wagi jako liczby 16-bitowe, BitNet koduje je jako wartości {-1, 0, +1}. To radykalnie zmniejsza rozmiar i zużycie energii. Wczesne implementacje (BitNet b1.58) pokazują, że modele 1,58-bitowe osiągają porównywalną jakość do FP16 przy kilkunastokrotnie mniejszym rozmiarze.

FP8 – nowy standard trenowania

Karty NVIDIA Blackwell (RTX 50-series) i AMD RX 9000-series wprowadzają natywne wsparcie dla FP8, które staje się nowym standardem trenowania. FP8 oferuje jakość zbliżoną do FP16 przy połowie rozmiaru.

Kwantyzacja aktywacji i KV cache

Dotychczas kwantyzowano głównie wagi. Kolejnym krokiem jest kwantyzacja aktywacji (wartości przepływających przez sieć podczas inferencji) oraz bufora kontekstu (KV cache). Pozwoli to na obsługę znacznie dłuższych kontekstów bez zwiększania pamięci.

Automatyczny dobór kwantyzacji

Narzędzia takie jak Ollama i LM Studio coraz częściej automatycznie dobierają optymalną kwantyzację na podstawie dostępnego sprzętu. W przyszłości wybór formatu i poziomu kompresji będzie całkowicie przezroczysty dla użytkownika.

→ W następnym rozdziale: **Trzy ekosystemy: Apple, NVIDIA, Microsoft – jak każdy z gigantów podchodzi do AI na urządzeniach użytkownika?**

Rozdział 7. Trzy ekosystemy: Apple, NVIDIA, Microsoft

Każda rewolucja technologiczna potrzebuje swojego ekosystemu â€“ spójnej ścieżki od sprzętu, przez oprogramowanie, aż po narzędzia dla programistów. W świecie sztucznej inteligencji trzy firmy wytyczyły tę ścieżkę w sposób radykalnie różny. Apple zbudowało ekosystem zamknięty, stawiając na prywatność i przetwarzanie na urządzeniu. NVIDIA stworzyła otwartą platformę obliczeniową, która napędza większość modeli AI na świecie. Microsoft połączył siły z OpenAI i wbudował asystenta AI w każdy zakątek swojego imperium biurowego. Każde z tych podejść ma inne zalety, ograniczenia i grupę docelową. W tym rozdziale przyjrzymy się każdemu z nich z osobna, a następnie porównamy je w praktyczny sposób.

POZIOM 1

Dla laika

Poznasz trzy główne ekosystemy AI — Apple, NVIDIA i Microsoft. Zrozumiesz ich filozofie i różnice bez wchodzenia w szczegóły techniczne.

Apple â€“ AI na własnych warunkach

Apple od lat konsekwentnie realizuje strategię, którą można streścić w trzech słowach: prywatność, kontrola, integracja. W przeciwieństwie do konkurentów, którzy przenoszą obliczenia AI do chmury, Apple uparcie dąży do tego, by jak najwięcej zadań wykonywać bezpośrednio na urządzeniu użytkownika. Ta filozofia znalazła swój pełny wyraz w Apple

Intelligence â€“ systemie AI ogłoszonym w czerwcu 2024 roku, który w 2025 i 2026 roku stał się integralną częścią iOS 19, iPadOS 19 i macOS Tahoe.

Apple Intelligence w pigułce:

System AI działający na urządzeniu, z opcją odciążenia do Private Cloud Compute.

Obejmuje narzędzia do pisania (Writing Tools), generowanie obrazów (Image Playground), Genmoji, ściąganie tła, inteligentne podsumowania i priorytetyzację powiadomień.

Tłumaczenie na żywo w czasie rzeczywistym w rozmowach FaceTime, wiadomościach i rozmowach telefonicznych.

Krzem, który myśli

Sercem ekosystemu Apple są własne procesory z rodziny M â€“ począwszy od przełomowego M1 w 2020 roku, aż po najnowszy M5 ogłoszony w październiku 2025 roku. Każda generacja przynosiła istotne ulepszenia w wydajności AI, ale M5 stanowi prawdziwy skok jakościowy.

Apple M5 â€“ AI w każdym rdzeniu

M5 został wykonany w procesie 3 nm trzeciej generacji (N3P) i według doniesień medialnych zawiera 28 miliardów tranzystorów (Apple nie podało oficjalnej liczby). Jego 10-rdzeniowy procesor (4 wydajnościowe + 6 efektywnościowych) oferuje do 15% wyższą wydajność wielowątkową względem M4. Kluczową innowacją jest jednak grafika: każdy z 10 rdzeni GPU zawiera wbudowany akcelerator neuronowy (Neural Accelerator), dzięki czemu M5 osiąga ponad 4-krotnie wyższą szczytową wydajność GPU dla obciążeń AI w porównaniu z M4.

16-rdzeniowy Neural Engine w M5 jest szybszy i bardziej energooszczędny od poprzedników, a przepustowość pamięci zunifikowanej wzrosła do **153 GB/s** (30% więcej niż M4). Pozwala to na uruchamianie dużych modeli językowych lokalnie na MacBooku Pro, iPadzie Pro i Vision Pro bez konieczności łączenia się z chmurą.

Architektura zunifikowanej pamięci

Największą przewagą Apple Silicon nad tradycyjnymi komputerami PC jest architektura zunifikowanej pamięci (Unified Memory). CPU, GPU i Neural Engine współdzielą tę samą pulę RAM, co eliminuje konieczność kopiowania danych między pamięcią procesora a kartą graficzną. Dla AI oznacza to, że modele mogą korzystać z pełnej dostępnej pamięci â€“ Mac z 64 GB RAM

może załadować model 70B w kwantyzacji Q4, co na komputerze z oddzielną kartą graficzną wymagałoby karty z 48 GB VRAM.

Warianty M5 Pro i M5 Max oferują do 64 GB pamięci, a nadchodzący M5 Ultra może obsłużyć nawet 256 GB. To sprawia, że Mac Studio i Mac Pro są platformami zdolnymi do uruchamiania modeli klasy Llama 5 70B, DeepSeek V4-Pro czy Bielik 11B bezpośrednio na biurku.

Od Core ML do Core AI

Przez lata głównym frameworkiem Apple dla uczenia maszynowego był Core ML. Służył on do wdrażania wytrenowanych modeli na urządzeniach Apple â€“ od klasyfikacji obrazów po przetwarzanie języka naturalnego. W 2026 roku Apple ogłosiło jednak koniec tej ery.

Core AI â€“ nowoczesny framework, który zastąpi Core ML. Zmiana nazwy z ML (Machine Learning) na AI (Artificial Intelligence) nie jest przypadkowa â€“ Apple uznało, że pojęcie „uczenie maszynowe" jest przestarzałe i nie oddaje współczesnych możliwości modeli językowych. Core AI ma umożliwić programistom łatwiejsze integrowanie zewnętrznych modeli AI z aplikacjami na iOS 20 i macOS 21.

MLX â€“ otwarty framework Apple

MLX (ML eXplore) to otwartoźródłowy framework uczenia maszynowego zaprojektowany przez zespół badawczy Apple. W przeciwieństwie do Core ML, który służy do wdrażania gotowych modeli, MLX został stworzony z myślą o badaczach i programistach, którzy chcą trenować i uruchamiać modele od podstaw.

MLX używa architektury zunifikowanej pamięci Apple Silicon, dzięki czemu operacje na CPU i GPU nie wymagają kosztownego kopiowania danych. W 2026 roku MLX zyskał na znaczeniu dzięki wsparciu dla nowych akceleratorów neuronowych w rdzeniach GPU M5. Popularne narzędzia, takie jak LM Studio i Ollama, zaczęły korzystać z MLX jako backendu, osiągając znaczące przyspieszenia.

MLX w liczbach:

Ollama 0.19 z backendem MLX: przyspieszenie prefillu o 1.6x, generowania o 1.8x na M5.

7B Q4 na Macu Mini M4: ~150 tokenów/s z MLX vs ~100 tokenów/s z llama.cpp (Metal).

Fine-tuning modelu 7B na MacBooku Pro M3: poniżej 10 minut.

Ekonomia: Mac Mini M4 Pro 48 GB ~1 600 USD za lokalne AI gotowe do pracy.

Możliwości Apple Intelligence

Apple Intelligence w 2026 roku oferuje następujące funkcje, dostępne w iPhonie 17 (A19), iPadzie Pro (M5) i MacBooku Pro (M5):

- Narzędzia do pisania â€“ korekta, przepisywanie, zmiana tonu, podsumowywanie w każdej aplikacji.
- Image Playground â€“ generowanie obrazów w stylach: animacja, ilustracja, szkic + style ChatGPT.
- Genmoji â€“ tworzenie spersonalizowanych emoji na podstawie zdjęć znajomych.
- Clean Up â€“ inteligentne usuwanie niechcianych obiektów ze zdjęć.
- Tłumaczenie na żywo â€“ napisy w rozmowach FaceTime i tłumaczenie rozmów telefonicznych.
- Smart Reply â€“ automatyczne propozycje odpowiedzi w Mailu.
- Reduce Interruptions â€“ tryb skupienia przepuszczający tylko pilne powiadomienia.
- Private Cloud Compute â€“ odciążenie złożonych zapytań do serwerów Apple Silicon z zachowaniem prywatności.

W 2026 roku Apple borykało się jednak z opóźnieniami w rozwoju Siri. Planowana, głęboka aktualizacja asystenta głosowego â€“ oparta na umowie z Google w sprawie Gemini AI â€“ została przesunięta z iOS 19.4 na iOS 19.5, a niektóre funkcje trafią dopiero do iOS 20. Apple zapowiedziało też przebudowę Siri „od zera” z nowym interfejsem chatbotowym na WWDC 2026.

Mimo opóźnień, ekosystem Apple pozostaje najsilniejszy w segmencie konsumenckim. Użytkownik nie musi wiedzieć, że używa AI â€“ funkcje są wbudowane w system i działają automatycznie, bez rejestracji, subskrypcji czy wysyłania danych na serwery.

POZIOM 2

Głębiej

Szczegółowa analiza sprzętu NVIDIA (CUDA, TensorRT, RTX, DGX Spark), ekosystemu Microsoft (Copilot, Azure AI Foundry) i porównanie wszystkich trzech ścieżek.

NVIDIA â€“ kręgosłup AI

Jeśli AI ma kręgosłup, to z pewnością jest nim NVIDIA. Firma założona w 1993 roku jako producent kart graficznych do gier stała się w 2026 roku największą firmą półprzewodnikową na świecie, z kapitalizacją rynkową przekraczającą 5 bilionów dolarów. NVIDIA kontroluje około 75–80% rynku akceleratorów AI, generując ponad 150 miliardów dolarów rocznego przychodu z centrów danych.

NVIDIA w liczbach (2026):

75–80% udziału w rynku akceleratorów AI ($150B+ przychodu z data center).

Architektura Rubin (następca Blackwell) w pełnej produkcji od połowy 2026 roku.

Karta RTX 5090 (32 GB VRAM) dostępna dla konsumentów â€“ ok. 50 tokenów/s dla 30B modeli.

DGX Spark â€“ osobisty superkomputer AI za ~3 000 USD.

Sprzęt â€“ od konsumenta do data center

Wyższością NVIDIA jest to, że jej architektura GPU jest skalowalna od karty graficznej za 500 aż po superkomputer za 5 milionów dolarów. Programista może napisać kod na laptopie z RTX 4060, a następnie wdrożyć go na klastrze GPU w chmurze bez zmiany ani jednej linii.

Segment konsumencki â€“ seria RTX 50 (Blackwell)

Karty graficzne NVIDIA pozostają najpopularniejszym wyborem do lokalnego AI. W 2026 roku flagowym modelem jest RTX 5090 z 32 GB pamięci GDDR7. Niższe modele, jak RTX 5070 Ti (16 GB) i RTX 4060 Ti (16 GB), oferują dobry stosunek ceny do wydajności. Kluczową cechą dla AI jest pamięć VRAM â€“ to ona, a nie liczba rdzeni czy taktowanie, determinuje, jakie modele można uruchomić.

DGX Spark â€“ osobisty superkomputer AI

W 2025 roku NVIDIA wprowadziła DGX Spark â€“ kompaktowe urządzenie (wielkości Mac Mini) wyposażone w zmodyfikowany układ Grace Blackwell, zdolne do uruchamiania modeli 70B+ w kwantyzacji Q4. DGX Spark jest pierwszą próbą NVIDIA wejścia w segment desktopowego AI dla programistów i badaczy, oferując pełną kompatybilność z CUDA i TensorRT-LLM.

Centra danych â€“ Vera Rubin

W marcu 2026 roku na GTC 2026 Jensen Huang ogłosił produkcję architektury Vera Rubin â€“ następcy Blackwell. Rubin wprowadza prawdziwe 3D-stosowanie pamięci, osiągając przepustowość NVLink 7.0 powyżej 10 TB/s. Nowe układy są zaprojektowane do pracy w szafach o mocy 1000 W, chłodzonych cieczą. Rubin umożliwia skalowanie klastrów do ponad 100 000 zunifikowanych GPU, co jest niezbędne do trenowania modeli o bilionach parametrów.

CUDA â€“ największa fosą obronna w technologii

NVIDIA wprowadziła CUDA (Compute Unified Device Architecture) w 2007 roku. To platforma programowania równoległego, która umożliwia programistom wykorzystanie GPU do ogólnych obliczeń, nie tylko renderowania grafiki. W 2026 roku CUDA ma blisko 20 lat optymalizacji i stanowi największą przewagę konkurencyjną NVIDIA.

Fosą obronną CUDA jest sieć efektów: miliony programistów napisały miliardy linii kodu w CUDA. Każda główna biblioteka AI (PyTorch, TensorFlow, JAX) domyślnie używa CUDA jako backendu. Przejście na konkurencyjny sprzęt AMD (ROCm) lub Intel (oneAPI) wymaga przepisania kodu i przeprocesowania modeli, co dla firm oznacza miesiące pracy i ryzyko.

TensorRT-LLM

TensorRT-LLM to otwartoźródłowe narzędzie NVIDIA do optymalizacji inferencji dużych modeli językowych. Działa na zasadzie kompilacji modelu do postaci silnika TensorRT, który jest optymalizowany pod konkretny model GPU. Oferuje techniki takie jak: kwantyzacja FP8 i NVFP4, Inflight Batching, disaggregated serving, spekulatywne dekodowanie (EAGLE-3) i szerokie Expert Parallelism.

TensorRT-LLM jest dostępny dla wszystkich kart NVIDIA z serii RTX 20 i nowszych, a także w chmurze za pośrednictwem NVIDIA NIM i NVIDIA Dynamo. Programiści mogą go używać lokalnie przez Dockera lub pip install.

NVIDIA AI Enterprise

Dla firm NVIDIA oferuje platformę AI Enterprise, która łączy certyfikowane oprogramowanie, narzędzia MLOps i wsparcie techniczne. W 2026 roku w skład platformy wchodzą: NVIDIA NIM (mikrousługi do inferencji), NVIDIA Dynamo (framework do skalowania inferencji), Nemotron (rodzina otwartych modeli) oraz Cosmos (AI fizyczne dla robotyki).

Otwarte modele NVIDIA

NVIDIA aktywnie rozwija własne otwarte modele: Nemotron dla agentów AI, Cosmos dla AI fizycznego i robotyki, Alpamayo dla pojazdów autonomicznych oraz Ising â€“ pierwsze otwarte modele AI do kwantowego przetwarzania danych (ogłoszone w kwietniu 2026 roku). Te modele działają najlepiej na sprzęcie NVIDIA, tworząc spójny łańcuch wartości.

Lokalne AI na NVIDIA â€“ co to oznacza w praktyce

Dla przeciętnego użytkownika posiadanie karty NVIDIA oznacza dostęp do najszerszego ekosystemu narzędzi AI: Ollama, LM Studio, OpenAI Whisper, ComfyUI (generowanie obrazów), Stable Diffusion, Fooocus i setek innych. Wszystkie te narzędzia działają natywnie na CUDA, co gwarantuje najwyższą wydajność.

Praktyczne zalecenia dla różnych scenariuszy:

- 7–8B modele: RTX 3060 12 GB lub RTX 4060 Ti 16 GB â€“ ~25–35 tok/s.
- 13–14B modele: RTX 4070 Ti Super 16 GB â€“ ~20 tok/s.
- 30B modele: RTX 3090/4090 24 GB â€“ ~15–50 tok/s (w zależności od kwantyzacji).
- 70B modele: 2x RTX 3090 lub dedykowana stacja robocza.
- 120B+: AMD AI Max+ 395 (128 GB unifikowanej pamięci) lub układy data center.

Uwaga: w 2026 roku dostępne są również konkurencyjne rozwiązania AMD (karty RX 7000/9000 z ROCm), jednak udział rynkowy NVIDIA i kompatybilność oprogramowania sprawiają, że wybór AMD nadal wiąże się z kompromisami.

Microsoft â€“ Copilot wszędzie

Microsoft obrał strategię radykalnie odmienną od Apple: zamiast budować wszystko samodzielnie, postawił na partnerstwo z OpenAI (inwestycja łącznie ponad 13 miliardów dolarów) i wbudował AI we wszystkie swoje produkty. W 2026 roku Copilot jest dostępny w Windows, Microsoft 365 (Word, Excel, PowerPoint, Outlook, Teams), Edge, Bing, GitHub i Dynamics 365.

Ekosystem Microsoft w pigułce:

Copilot â€“ asystent AI obecny w Windows, Office, Teams, Edge i GitHub.

Copilot+ PC â€“ komputery z NPU o wydajności minimum 40 TOPS, 16 GB RAM i 256 GB SSD.

Azure AI Foundry â€“ platforma z 1 800+ modelami, prompt flow, RAG i MLOps.

Azure OpenAI â€“ OpenAI API hostowane w Azure z certyfikatami SOC 2, HIPAA, FedRAMP.

Rodzina Copilotów

Microsoft ujednolicił markę Copilot dla wszystkich swoich produktów AI. W 2026 roku wyróżniamy kilka głównych wariantów:

Microsoft 365 Copilot

Zintegrowany z pakietem biurowym. Tworzy dokumenty w Wordzie, generuje prezentacje w PowerPoint, analizuje dane w Excelu, podsumowuje wątki e-mail w Outlooku i notatki ze spotkań w Teams. Działa w oparciu o Microsoft Graph â€“ ma dostęp do kalendarza, kontaktów, plików i konwersacji użytkownika, dzięki czemu może udzielać spersonalizowanych odpowiedzi.

GitHub Copilot

Zadebiutował w 2021 roku jako asystent programisty. W 2026 roku oferuje autouzupełnianie kodu, generowanie całych funkcji z opisu, refaktoryzację, testowanie i debugowanie. Obsługuje wszystkie główne języki i środowiska programistyczne (VS Code, Visual Studio, JetBrains, Neovim).

Windows Copilot

Dostępny bezpośrednio z paska zadań Windows 11. Umożliwia zmianę ustawień systemowych, uruchamianie aplikacji, wyszukiwanie plików i odpowiedzi na pytania. W 2026 roku Microsoft intensywnie pracuje nad uruchamianiem Copilota lokalnie na NPU, zamiast wysyłać zapytania do chmury.

Copilot Studio

Platforma low-code do tworzenia własnych asystentów i agentów AI. Użytkownicy mogą definiować scenariusze, podłączać źródła danych (SharePoint, bazy SQL, ServiceNow) i publikować kopiloty w Teams, na stronie WWW lub w aplikacjach mobilnych.

Microsoft Agent Framework

Otwartoźródłowy SDK do tworzenia wieloagentowych systemów AI. Umożliwia definiowanie wyspecjalizowanych agentów, które współpracują ze sobą, np. jeden agent analizuje dokument, drugi waliduje reguły biznesowe, trzeci aktualizuje system CRM.

Copilot+ PC â€“ AI w każdym laptopie

W 2024 roku Microsoft wprowadził kategorię Copilot+ PC â€“ komputerów z dedykowanym układem NPU (Neural Processing Unit) o wydajności minimum 40 TOPS. W 2026 roku wymagania te są standardem dla nowych laptopów z Windows 11.

Wymagania sprzętowe Copilot+ PC

Aby komputer mógł nosić oznaczenie Copilot+, musi spełniać:

- NPU o wydajności co najmniej 40 TOPS (trillionów operacji na sekundę).
- Minimum 16 GB RAM (DDR5 lub LPDDR5).
- 256 GB pamięci masowej (SSD lub UFS).

Procesory spełniające te wymagania w 2026 roku to:

- Qualcomm Snapdragon X Elite/X2 â€“ NPU do 80 TOPS.
- Intel Core Ultra 200V (Lunar Lake) â€“ NPU 48 TOPS.
- AMD Ryzen AI 300 â€“ NPU 50 TOPS.

Funkcje AI dostępne na Copilot+ PC

- Recall â€“ przeszukiwalna oś czasu aktywności na ekranie (szyfrowana, lokalna, wyłączona domyślnie).
- Live Captions â€“ tłumaczenie napisów na żywo w czasie rzeczywistym.
- Paint Cocreator â€“ generowanie obrazów z tekstu i szkiców.

- Windows Studio Effects â€“ rozmycie tła, kontakt wzrokowy, kadrowanie automatyczne.
- Auto Super Resolution â€“ AI upscaling grafiki.

W 2026 roku pojawiły się głosy krytyczne, że pomiędzy obietnicami marketingowymi a rzeczywistą użytecznością Kopilota wciąż istnieje luka. Wiele funkcji wymaga połączenia z chmurą, a lokalne przetwarzanie na NPU jest ograniczone do wąskiej liczby zadań. Co więcej, popularne narzędzia AI (Ollama, LM Studio) w 2026 roku nadal nie korzystają z NPU — używają GPU lub CPU.

Azure â€“ chmura AI dla przedsiębiorstw

Azure to największe źródło przychodów Microsoft związanych z AI. W 2026 roku platforma Azure AI Foundry (dawniej Azure AI Studio) jest głównym miejscem, w którym firmy budują i wdrażają aplikacje AI.

Azure AI Foundry

Azure AI Foundry to zunifikowana platforma rozwoju AI, oferująca:

- Katalog 1 800+ modeli: GPT-5.5, Claude Opus 4.7, DeepSeek V4-Pro, Llama 5, Qwen 3.7-Max, Mistral, Gemma 4, GLM-5.1 i inne.
- Prompt flow â€“ wizualne tworzenie przepływów AI.
- RAG z Azure AI Search â€“ wyszukiwanie hybrydowe o 20–30% dokładniejsze od słownego.
- Fine-tuning modeli GPT-5.5, Gemma 4 i Llama 5 z zarządzaną infrastrukturą.
- Wbudowane filtry bezpieczeństwa, wykrywanie halucynacji i ochrona przed jailbreakiem.
- Certyfikaty SOC 2, HIPAA, FedRAMP i ISO 27001.
- Agent Service â€“ definiowanie agentów w YAML, dwie komendy CLI do wdrożenia.

Azure OpenAI

Azure OpenAI to usługa Microsoftu hostująca modele OpenAI (GPT-5.5, GPT-5, GPT-4.1, seria o3/o4) w infrastrukturze Azure. W przeciwieństwie do bezpośredniego API OpenAI, Azure OpenAI oferuje:

- Prywatność danych: dane nie opuszczają regionu Azure.
- Zgodność z regulacjami: RODO, HIPAA, FedRAMP.
- Modele cenowe: Standard (pay-as-you-go), Provisioned Throughput (rezerwacja), Batch API (50% zniżki).
- Integrację z Azure AD, Azure Monitor, Private Link.

Phi-4 â€“ małe modele Microsoftu

Phi-4 to rodzina małych modeli językowych opracowanych przez Microsoft, konkurencyjnych wobec Llama 5 i Gemma 4. Phi-4-mini (3.8B parametrów) jest dostępny na Azure AI Foundry w cenie ~0.04 USD za milion tokenów wejściowych. Phi-4 działa również lokalnie na urządzeniach Copilot+ PC dzięki optymalizacji pod NPU.

Porównanie ekosystemów

Każdy z trzech ekosystemów jest optymalny dla innej grupy użytkowników. Poniższe zestawienie tabelaryczne i analiza pomogą wybrać właściwą ścieżkę.

Tabela porównawcza: Apple vs NVIDIA vs Microsoft

Kryterium	Apple	NVIDIA	Microsoft
Sprzęt AI	M5 (Mac, iPad, Vision Pro)	RTX 5090, DGX Spark, Rubin	Copilot+ PC (Snapdragon X2, Core Ultra, Ryzen AI)
Główne AI	Apple Intelligence	CUDA, TensorRT-LLM	Copilot, Azure AI Foundry
Pamięć	Unifikowana do 256 GB (Ultra)	VRAM do 32 GB (RTX 5090)	Unifikowana lub VRAM (CPU+GPU+NPU)
Lokalne AI	MLX, Core AI	TensorRT-LLM, NIM	Gemma 4, Phi-4, ONNX Runtime
Chmura	Private Cloud Compute	DGX Cloud	Azure OpenAI, Azure AI Foundry
Platforma	macOS, iOS, iPadOS, visionOS	Windows, Linux (głównie)	Windows (głównie), Linux, Mac
Otwartość	MLX (open source)	CUDA, TensorRT-LLM (open)	Copilot (zamknięty)
Modele własne	Apple Foundation Models	Nemotron, Cosmos, Ising	Phi-4, Gemma 4, Azure AI
Narzędzia AI	Xcode, Create ML	PyTorch, TF, TensorRT	VS Code, Copilot, Prompt flow
Grupa docelowa	Konsumenci, twórcy	Programiści, badacze, firmy	Przedsiębiorstwa, programiści
Prywatność	Najwyższa (on-device)	Średnia (zależna od konfiguracji)	Różna (chmura + lokalne)
Koszt wejścia	Mac Mini M4 48 GB ~$1 600	RTX 4060 Ti 16 GB ~$400	Copilot+ PC ~$1 000+
Udział w rynku AI	Niszowy (AI na urządzeniach)	**75–80% (akceleratory)**	Dominujący w chmurze enterprise

Kiedy który ekosystem?

Wybierz Apple, jeśli:

- Pracujesz na komputerze Mac i zależy Ci na prywatności.
- Potrzebujesz AI, które działa bez konfiguracji, rejestracji i subskrypcji.
- Chcesz uruchamiać lokalnie modele AI bez zawracania sobie głowy sterownikami i bibliotekami.
- Twoje obciążenia AI mieszczą się w 64–256 GB pamięci (modele do 120B).
- Cenisz bezgłośną, chłodną i energooszczędną pracę.

Apple to najlepszy wybór dla początkujących i twórców, którzy chcą AI bez zbędnego skomplikowania.

Wybierz NVIDIA, jeśli:

- Chcesz maksymalnej wydajności za każdą cenę.
- Potrzebujesz szerokiego ekosystemu narzędzi AI (Ollama, ComfyUI, Whisper, TensorRT).
- Zajmujesz się trenowaniem lub fine-tuningiem modeli.
- Twoje obciążenia wymagają wielu GPU w klastrze.
- Pracujesz w PyTorch/TensorFlow i zależy Ci na kompatybilności w chmurze.

NVIDIA to standard branżowy. Jeśli budujesz aplikacje AI profesjonalnie, nie ma realnej alternatywy dla CUDA.

Wybierz Microsoft, jeśli:

- Twoja organizacja już korzysta z Microsoft 365 i Azure.
- Potrzebujesz AI wbudowanego w codzienne narzędzia biurowe.
- Chcesz tworzyć aplikacje AI w Azure z pełną zgodnością regulacyjną.
- Interesują Cię agenci AI i automatyzacja procesów biznesowych.
- Szukasz dostępu do modeli OpenAI (GPT-5.5) w bezpiecznej, zgodnej z RODO chmurze.

Microsoft to wybór dla firm, które chcą AI jako część istniejącej infrastruktury, a nie osobny projekt.

Podsumowanie

Trzy opisane w tym rozdziale ekosystemy nie konkurują bezpośrednio ze sobą â€“ raczej obsługują różne warstwy rynku AI. NVIDIA kontroluje fizyczną warstwę obliczeń (GPU + CUDA), Microsoft dominuje w warstwie aplikacji enterprise (Copilot + Azure), a Apple wygrywa w segmencie konsumenckim (AI na urządzeniu bez kompromisów w prywatności).

Wiele osób i firm korzysta z wszystkich trzech jednocześnie. Programista może pisać kod na MacBooku (Apple), używać GitHub Copilot w VS Code (Microsoft) i uruchamiać modele na serwerze z GPU NVIDIA w Azure. Każdy z tych elementów jest najlepszy w swoim zadaniu, a ich wspólne użycie daje największą wartość.

Kluczowa rada na 2026 rok:

Nie wybieraj jednego ekosystemu na siłę. Narzędzia AI są ze sobą kompatybilne coraz lepiej. Uruchamiaj modele lokalnie na Apple Silicon lub NVIDIA GPU, twórz w Copilot/VS Code, wdrażaj w Azure. Każde narzędzie do swojego zadania.

Niezależnie od wybranego ekosystemu, jedno jest pewne: w 2026 roku AI nie jest już przyszłością, ale codziennością. Umiejętność świadomego wyboru narzędzi â€“ i rozumienia ich

ograniczeń â€“ to kompetencja, która będzie coraz bardziej wartościowa, niezależnie od branży czy stanowiska.

Rozdział 8. Jaki sprzęt wybrać?

Wybór sprzętu do lokalnego AI to jedna z najważniejszych decyzji, jakie podejmiesz jako użytkownik modeli językowych. Zła decyzja oznacza albo przepłacenie za moc, której nie wykorzystasz, albo – co gorsza – zakup maszyny, na której wymarzony model w ogóle nie działa. W tym rozdziale przeprowadzę Cię przez wszystkie opcje: od budżetowego minikomputera za 400 dolarów po profesjonalną stację roboczą za 10 000+. Każdą konfigurację oceniam pod kątem trzech kryteriów: jakie modele uruchomisz, jak szybko będą działać i ile to kosztuje.

VRAM to król — wszystko inne to szczegóły

W lokalnym AI liczy się przede wszystkim jedno: czy model mieści się w pamięci. Szybkość rdzeni, częstotliwość taktowania, liczba tranzystorów — to wszystko ma znaczenie drugorzędne. Jeśli model się nie mieści, wydajność spada setki razy, czyniąc AI bezużytecznym.

Podstawowe rï¿½wnanie pamieci

Ilość pamięci potrzebna do uruchomienia modelu wyraża się prostym wzorem:

Potrzebna pamięć ≈ (liczba parametrów × bajty na parametr) + bufor kontekstu + narzut

Przykład: Model 7B w kwantyzacji Q4_K_M = 7 mld × 0.5 bajta = ~3.5 GB + bufor 2 GB = ~5.5 GB.

Model 70B w Q4_K_M = 70 × 0.5 = ~35 GB + bufor 5 GB = ~40 GB.

Model 13B w FP16 = 13 × 2 = ~26 GB + bufor = ~28 GB.

Tabela wymagań pamięci dla typowych modeli

Dane aktualne na: maj 2026. Zapotrzebowanie na pamięć spada z każdą nową metodą kwantyzacji.

Model	FP16	Q8	Q4_K_M	Min. VRAM
3B (Gemma 4 E2B, Qwen 3.6 3B)	~6 GB	~3 GB	~2 GB	~4 GB
7–8B (Llama 5 8B, Mistral Large 4 7B)	~16 GB	~8 GB	~5 GB	~8 GB
13–14B (Qwen 3.6 14B, DeepSeek V4-Flash)	~28 GB	~14 GB	~8 GB	~12 GB
27–30B (DeepSeek V4-Flash 32B, Qwen 3.7-Max)	~60 GB	~30 GB	~18 GB	~24 GB
70–72B (Llama 5 70B, Qwen 3.7-Max)	~140 GB	~70 GB	~40 GB	**~48 GB**
120B+ (Qwen, DeepSeek)	~240 GB+	~120 GB+	~70 GB+	**~96 GB**

W tradycyjnej architekturze PC pamięć VRAM karty graficznej jest osobną pulą — model musi się w niej całkowicie zmieścić, aby działać w pełni na GPU. W Apple Silicon z pamięcią zunifikowaną (UMA) CPU, GPU i Neural Engine współdzielą tę samą pamięć RAM, co pozwala uruchamiać większe modele kosztem nieco niższej szybkości.

NVIDIA — uniwersalny wybór

NVIDIA pozostaje w 2026 roku standardem branżowym. Karty GeForce RTX (serie 40 i 50) oferują dostęp do dojrzałego ekosystemu CUDA, co oznacza kompatybilność z każdym narzędziem AI — od Ollamy przez ComfyUI po PyTorch. Wybór karty NVIDIA to wybór pewności: wszystko będzie działać od razu, bez konfiguracji.

Porównanie kart NVIDIA dla AI

Ceny i dostępność aktualne na: maj 2026. Rynek GPU zmienia się co kwartał.

Karta	VRAM	Przepust.	Cena (nowa)	Modele 7B	Modele 30B+
RTX 4060 Ti	16 GB	288 GB/s	~$400	Q8: 50–70 t/s	Nie (za mało VRAM)
RTX 5070	12 GB	~700 GB/s	~$590	Q8: 70–90 t/s	Q4 przy offloadzie
RTX 4070 Ti Super	16 GB	672 GB/s	~$800	Q8: 65–85 t/s	Q4: ~10 t/s (offload)
RTX 3090 (używana)	**24 GB**	936 GB/s	~$650–750	Q8: 60–80 t/s	Q4: ~15–20 t/s
RTX 4090	**24 GB**	1 008 GB/s	~$1 800	Q8: 90–140 t/s	Q4: ~15–25 t/s
RTX 5090	**32 GB**	1 792 GB/s	~$2 000	Q8: 130–170 t/s	Q4: ~20–30 t/s

Którą kartę wybrać?

RTX 3060 12 GB / RTX 4060 Ti 16 GB — budżetowy start

To minimum, od którego warto zacząć w 2026 roku. 12 GB VRAM pozwala uruchomić modele 7–14B w kwantyzacji Q4. 16 GB (4060 Ti) daje komfort pracy z modelami 13–14B. Nie licz na modele większe niż 30B.

RTX 3090 używana — najlepszy stosunek ceny do możliwości

Używana RTX 3090 z 24 GB VRAM za ~$650–750 to w 2026 roku absolutny król opłacalności. Uruchomisz na niej większość modeli 7–30B w pełni na GPU, a 70B w kwantyzacji Q4 z offloadem na CPU. Jeśli budujesz pierwszą maszynę do AI — to jest twoja karta.

RTX 4090 — szybkość i zapas

Karta, która przez lata była szczytem możliwości konsumenckich. 24 GB VRAM i 1 008 GB/s przepustowości zapewniają 90–140 tokenów/s na modelach 7–8B. Ograniczeniem jest wciąż 24 GB — modele 70B mieszczą się tylko przy Q4 i offloadzie.

RTX 5090 — nowy król konsumenckiego AI

Wprowadzona w 2025 roku z 32 GB GDDR7 i przepustowością 1 792 GB/s. To pierwsza karta konsumencka, która pozwala uruchomić modele 30–32B w pełni w pamięci GPU, a modele 70B z mniejszym offloadem. Różnica w stosunku do 4090 jest znacząca: +72% wydajności na modelach 7B, możliwość pracy z większym kontekstem.

Apple Silicon — cichy gigant

Apple Silicon zrewolucjonizował lokalne AI dzięki architekturze zunifikowanej pamięci (UMA). W komputerach Mac pamięć RAM jest współdzielona między CPU, GPU i Neural Engine.

Oznacza to, że możesz uruchomić model 70B na MacBooku, który pobiera 60 W zamiast 450 W. Kosztem jest niższa przepustowość pamięci niż w dedykowanych GPU.

Porównanie konfiguracji Apple Silicon dla AI

Model	RAM	Przepust.	Modele	Szac. cena
Mac Mini M4 Pro	**48–64 GB**	273 GB/s	13–30B Q4: 12–18 t/s	~$1 600–2 000
MacBook Pro M4 Max	**64–128 GB**	~500 GB/s	70B Q4: 15–25 t/s	~$3 500–4 500
Mac Studio M4 Max	**64–128 GB**	~500 GB/s	70B Q4: 18–28 t/s	~$3 000–4 000
MacBook Pro M5 Max	64–128 GB	~600 GB/s	70B Q4: 20–30 t/s	~$3 500–4 500
Mac Studio M5 Ultra*	192–256 GB	~800 GB/s	120B: 10–15 t/s	~$7 000–10 000

** M5 Ultra spodziewany w drugiej połowie 2026 roku.*

Którego Maca wybrać?

Mac Mini M4 Pro 48–64 GB — najlepszy stosunek ceny do możliwości

Dla większości osób to optymalny wybór. Za ~$1 600–2 000 otrzymujesz maszynę zdolną do uruchamiania modeli 30B w Q4 z komfortową szybkością 12–18 t/s. Pobór mocy to zaledwie 65 W pod obciążeniem. Minikomputer wielkości małej książki. Cisza. Zero problemów z konfiguracją. To jest obecnie rekomendacja nr 1 dla osób zaczynających przygodę z lokalnym AI.

MacBook Pro M4 Max 128 GB — mobilna stacja AI

Jeśli potrzebujesz pracować w różnych miejscach, MacBook Pro z 128 GB RAM to jedyna opcja, która pozwala uruchomić model 70B w pełni lokalnie na laptopie. 15–25 t/s to użyteczna szybkość. Minusem jest cena (~$4 000+).

Mac Studio M5 Ultra 256 GB — bezkompromisowe AI

Spodziewany w 2026 roku M5 Ultra z 256 GB zunifikowanej pamięci jako pierwszy komputer konsumencki pozwoli uruchomić modele 120B+ w kwantyzacji Q8, a nawet 70B w FP16 bez utraty jakości. To już poziom profesjonalny, konkurujący z serwerami z wieloma GPU.

Apple Silicon vs NVIDIA — które lepsze?

Poniżej ~24 GB modelu: NVIDIA jest szybsza (większa przepustowość pamięci).

Powyżej ~24 GB: Apple Silicon wygrywa, bo może załadować cały model (NVIDIA nie ma tyle VRAM).

Punkt przecięcia: modele ~30B w Q4 (~18 GB) — obie platformy działają podobnie.

Moc: Mac Mini ~65 W vs PC z RTX 4090 ~450 W. Rocznie: ~$35 vs ~$394 energii przy 24/7.

AMD — alternatywa z potencjałem

Karty AMD z serii RX 7000 i 9000 zyskują w 2026 roku na znaczeniu dzięki dojrzeniu sterowników ROCm 7.2+. RX 7900 XTX (24 GB) oferuje tyle samo VRAM co RTX 4090 za znacznie niższą cenę (~$800–900). Wadą jest około 2 razy niższa szybkość inferencji (niższa przepustowość pamięci) i mniejszy ekosystem oprogramowania.

AMD jest dobrym wyborem, jeśli:

- Chcesz maksymalną ilość VRAM za najmniejsze pieniądze.
- Pracujesz na Linuksie (Ubuntu 22.04/24.04) — ROCm na Windows jest ograniczony.
- Jesteś gotów na więcej ręcznej konfiguracji niż z NVIDIA.
- Nie potrzebujesz najwyższej wydajności — akceptujesz ~20 t/s zamiast ~40 t/s.

AMD Ryzen AI Max+ 395 — ciekawa alternatywa

Procesor AMD Ryzen AI Max+ 395 to układ z 128 GB zunifikowanej pamięci LPDDR5x, montowany w minikomputerach takich jak Minisforum MS-S1 Max. Dzięki architekturze UMA może uruchomić modele 70B w Q4 w całości, podobnie jak Mac Studio. Cena (~$2 500) jest niższa niż Maca, ale wymaga Linuksa i więcej konfiguracji.

NPU — czy to się przydaje?

NPU (Neural Processing Unit) to dedykowany układ do akceleracji AI w nowych procesorach. Wymaganie Copilot+ PC to minimum 40 TOPS. W 2026 roku procesory z NPU to:

- Qualcomm Snapdragon X Elite/X2 — NPU do 80 TOPS.
- Intel Core Ultra 200V (Lunar Lake) — NPU 48 TOPS.
- AMD Ryzen AI 300 — NPU 50 TOPS.

Jednak NPU w 2026 roku nie służą do uruchamiania modeli AI, z których korzysta większość użytkowników. Ollama, LM Studio, ComfyUI — te narzędzia nie używają NPU. Używają GPU. NPU przyspiesza funkcje systemowe Windows: efekty kamer, tłumaczenie napisów, rozmycie tła. Do poważnego AI na razie się nie nadaje.

Nie kupuj komputera dla NPU.

W 2026 roku NPU jest dla systemu, nie dla twoich modeli. Wszystkie narzędzia AI używają GPU (NVIDIA CUDA) lub CPU (Apple MLX). NPU w laptopie wciąż czeka na swój przełomowy moment.

Gotowe konfiguracje według budżetu

Poniżej znajdziesz sprawdzone zestawy sprzętowe na każdą kieszeń, od $400 do $10 000. Każdą konfigurację dobrano tak, by maksymalizować stosunek VRAM do ceny.

Poziom 1: Startowy (~$400–600)

Opcja A: Minikomputer Beelink EQR6 32 GB

Minikomputer z AMD Ryzen 7 i 32 GB RAM. Uruchomisz modele 3–8B w Q4 na CPU. Szybkość: ~10 t/s. Idealny, by spróbować lokalnego AI za minimalne pieniądze.

Opcja B: Używany PC + RTX 3060 12 GB

Używany komputer biurowy (np. Dell Optiplex z i7) za ~$200 + RTX 3060 12 GB za ~$200. Razem ~$400. Uruchomisz modele 7–8B w Q8 pełnią na GPU. To najtańsza realna ścieżka do lokalnego AI na GPU.

Poziom 2: Średni (~$1 000–1 600)

Rekomendacja: Mac Mini M4 Pro 48 GB

Najlepszy stosunek ceny do możliwości w 2026 roku. Za ~$1 600 otrzymujesz model 30B w Q4 z szybkością 12–18 t/s. Cisza, mały pobór prądu, zero konfiguracji.

Alternatywa: PC z używaną RTX 3090 24 GB

PC (Ryzen 5, 32 GB RAM) + używana RTX 3090 = ~$1 200–1 400. Modele 30B w pełni na GPU, szybkość: 15–25 t/s. Więcej hałasu i ciepła.

Poziom 3: Zaawansowany (~$2 500–4 000)

Rekomendacja: PC z RTX 4090 lub RTX 5090

Pełny zestaw z Ryzen 7, 64 GB DDR5 i RTX 4090 (24 GB) lub RTX 5090 (32 GB). Cena: $2 500–4 000. Modele 30B: 60–90 t/s. Modele 70B: offload na CPU z szybkością 8–12 t/s. To wybór entuzjastów potrzebujących maksymalnej wydajności.

Alternatywa: AMD Ryzen AI Max+ 395 128 GB

Minikomputer (Minisforum) z 128 GB LPDDR5x za ~$2 500. Uruchamia 70B Q4 w całości. Niższa szybkość niż RTX 4090, ale większa pamięć. Wymaga Linuksa.

Poziom 4: Profesjonalny (~$4 000–10 000)

Rekomendacja: Mac Studio M4 Max 128 GB

Za ~$4 000 uruchomisz 70B Q4 z szybkością 18–28 t/s. Cisza, niezawodność, macOS. Dla osób, dla których czas to pieniądz, a nie chcą walczyć z konfiguracją.

Alternatywa: Podwójna RTX 3090/4090 (NVLink/bez)

- Dwie używane RTX 3090 (48 GB łącznie) za ~$1 300–1 500.
- Wymaga płyty głównej z dwoma slotami x16 i zasilacza 1000 W+.
- Modele 70B w pełni na GPU: 20–35 t/s.

Szczyt: Mac Studio M5 Ultra 256 GB (~$7 000–10 000)

Spodziewany w drugiej połowie 2026 roku. Uruchomi modele 120B+ w Q8, a nawet 70B w FP16 bez utraty jakości. Dla badaczy i firm, które potrzebują maksymalnej dokładności.

Koszt całkowity (TCO)

Kupno sprzętu to nie jedyny wydatek. Poniżej zestawiłem szacunkowe koszty 3-letnie dla trzech głównych opcji:

Koszt	Mac Mini M4 Pro	PC + RTX 4090	Mac Studio M4 Max
Sprzęt	~$1 600	~$3 000	~$4 000
Energia (3 lata, 8h/dzień)	~$80	~$400	~$150
Chłodzenie (jeśli potrzebne)	$0	~$100	$0
Konserwacja/części	~$0	~$100	~$0
Łącznie (3 lata)	**~$1 700**	**~$3 600**	**~$4 150**
Koszt miesięczny	~$47	~$100	~$115

Porównanie z chmurą: abonament ChatGPT Pro kosztuje $200/miesiąc. Po 3 latach to $7 200. Mac Mini M4 Pro za $1 700 zwraca się po ~9 miesiącach. PC z RTX 4090 po ~18 miesiącach. Po tym czasie masz własne AI bez comiesięcznych opłat.

Pozostałe komponenty

RAM systemowy

Minimum 32 GB dla komputerów PC. Jeśli uruchamiasz modele z offloadem na CPU (gdy model nie mieści się w VRAM), potrzebujesz tyle RAM, ile wynosi wielkość modelu. Dla modeli 70B z 40 GB modelem potrzebujesz co najmniej 64 GB RAM, najlepiej 128 GB.

Dysk NVMe SSD

Modele AI zajmują dużo miejsca. Plik 70B w Q4_K_M to ~40 GB. Jeden model Llama 5 70B w kilku kwantyzacjach może zająć 150 GB. Rekomenduję dysk 1–2 TB NVMe SSD. Szybkość odczytu ma znaczenie przy ładowaniu modeli — im szybszy dysk, tym krótszy czas uruchamiania.

CPU

W lokalnej inferencji AI CPU ma znaczenie drugorzędne, o ile masz wydajne GPU. Do offloadu na CPU ważna jest liczba rdzeni i pamięć podręczna. Każdy nowoczesny 8-rdzeniowy procesor (Ryzen 7, Core i7, Apple M-series) wystarczy.

Zasilacz

Karty grażowe AI są prądożerne: RTX 3090 ~350 W, RTX 4090 ~450 W, RTX 5090 ~575 W. Do zestawu z pojedynczą kartą potrzebujesz zasilacza 750–1 000 W. Do dwóch kart — 1 200–1 500 W. Nie oszczędzaj na zasilaczu.

Decyzja w 30 sekund

Jeśli nie masz czasu na analizę wszystkich opcji, oto skrócona ścieżka decyzyjna:

Schemat wyboru sprzętu do AI w 2026 roku:

Budżet < $800? → Używany PC + RTX 3060 12 GB lub Mac Mini M1 16 GB.

Chcesz po prostu działać? → [Mac Mini M4 Pro 48 GB] — najlepszy stosunek ceny do możliwości.

Potrzebujesz maksymalnej szybkości dla modeli <24 GB? → PC z RTX 4090 lub RTX 5090.

Docelowo modele 70B+? → Mac Studio M4 Max 128 GB lub PC z 2x RTX 3090.

Pracujesz w podróży? → MacBook Pro M4 Max 128 GB.

Jesteś badaczem i potrzebujesz 120B+ w pełnej precyzji? → Mac Studio M5 Ultra 256 GB.

Niezależnie od wyboru, pamiętaj o najważniejszej zasadzie: VRAM lub zunifikowana pamięć to jedyna rzecz, której nie da się obejść. Możesz mieć najszybszy procesor i najwięcej rdzeni — jeśli model się nie mieści w pamięci, nie będziesz z niego korzystać. Inwestuj w pamięć, a resztę dostosuj do budżetu.

Rozdział 9. Jak uruchomić AI u siebie w domu?

W poprzednim rozdziale dowiedziałeś się, jaki sprzęt wybrać. Teraz czas na działanie — zainstalujemy niezbędne narzędzia, ściągniemy pierwsze modele i uruchomimy własne AI lokalnie. Pokażę Ci dwie drogi: szybką przez graficzny interfejs LM Studio oraz elastyczną przez terminal z Ollamą. Na końcu sięgniesz głębiej — zrozumiesz, jak działają te narzędzia i jak je dostroić do swoich potrzeb.

POZIOM 1

Dla laika

Krok po kroku: zainstalujesz Ollamę i LM Studio, ściągniesz pierwszy model i zaczniesz z nim rozmawiać w kilka minut.

Droga 1: Ollama — szybki start z terminala

Ollama to w 2026 roku najpopularniejsze narzędzie do uruchamiania modeli językowych lokalnie. Działa na Windows, macOS i Linuksie, oferuje terminal, API oraz integrację z Chmurą Ollama dla większych modeli. Od wersji 0.19 (marzec 2026) używa Apple MLX na procesorach M-series, dając nawet 2x większą szybkość inferencji.

Instalacja

Windows

Pobierz instalator z oficjalnej strony ollama.com i uruchom. Po instalacji otwórz terminal (cmd lub PowerShell) i sprawdź działanie:

ollama --version

Jeśli widzisz numer wersji (np. 0.23.x), instalacja się powiodła. Ollama działa w tle jako usługa systemowa. Możesz ją kontrolować z poziomu ikonki na pasku zadań.

macOS

Pobierz plik .dmg z ollama.com i przeciągnij do folderu Aplikacje. Możesz też użyć menedżera paczek:

brew install ollama

Po uruchomieniu Ollamy zobaczysz ikonę na pasku menu. Terminal jest gotowy do pracy.

Linux

Jedna komenda w terminalu:

curl -fsSL https://ollama.com/install.sh | sh

Instalator sam wykryje dystrybucję, zainstaluje zależności i skonfiguruje usługę systemd. Po instalacji uruchom Ollamę:

ollama serve

Ściąganie i uruchamianie pierwszego modelu

Po instalacji otwórz terminal i wpisz:

ollama run llama5

Ollama automatycznie pobierze model Llama 5 8B — waży ~2 GB. Po kilku minutach zobaczysz znak zachęty i możesz zacząć rozmowę. Wpisz /help, by zobaczyć komendy wewnętrzne, lub /exit, by zakończyć.

Wskazówka: Jeśli nie masz karty NVIDIA z dużą ilością VRAM, model 8B możesz uruchomić na większości komputerów. Na Apple Silicon z MLX działa błyskawicznie — ponad 80 tokenów na sekundę.

Podstawowe komendy Ollamy

Komenda	Opis
ollama list	Wyświetla pobrane modele z rozmiarem
ollama pull <model>	Pobiera model bez uruchamiania
ollama run <model>	Pobiera (jeśli trzeba) i uruchamia model
ollama rm <model>	Usuwa model z dysku
ollama ps	Pokazuje modele załadowane w pamięci
ollama stop <model>	Zwalnia model z pamięci
ollama show <model>	Szczegóły modelu (parametry, kwantyzacja)

Modele, które warto wypróbować

Ollama w 2026 roku oferuje setki modeli. Na początek polecam te sprawdzone:

Model	Rozmiar	Parametry	Min. RAM	Zastosowanie
llama5:1b	~1.5 GB	1B	4 GB	Szybkie odpowiedzi, testy
qwen3.6:3b	~2.5 GB	3B	4 GB	Ogólny chat, polecam na start
gemma4:4b	~3 GB	4B	6 GB	Bardzo dobry stosunek jakości do rozmiaru
qwen3.6:7b	~4.5 GB	7B	8 GB	Doskonały model ogólny
mistral-large:7b	~4.5 GB	7B	8 GB	Szybki i niezawodny
gemma4:14b	~9 GB	14B	16 GB	Kodowanie i logika
gemma4:12b	~8 GB	12B	12 GB	Najlepszy model średniej wielkości
qwen3.7:35b	~22 GB	35B	24 GB	Jakość bliska GPT-5.5, wymaga VRAM

Przyszłość: model hybrydowy lokalnie + chmura

Wkrótce możliwe stanie się korzystanie z chmurowych modeli przez ten sam interfejs co lokalnych. Wyobraźmy sobie taką pracę: małe modele uruchamiasz u siebie, a gdy potrzebujesz większej mocy — sięgasz po zasoby w chmurze, bez zmiany narzędzi. To naturalny kierunek ewolucji lokalnych platform AI.

Hybryda lokalnie + chmura

Chmura: większe modele (120B+), zero obciążenia sprzętu, wymaga internetu.

Lokalnie: pełna prywatność, działa offline, zero opłat, ograniczone rozmiarem VRAM.

Przyszłość: małe modele lokalnie, duże w chmurze — wszystko z tego samego interfejsu.

Droga 2: LM Studio — dla niecierpliwych

Jeśli wolisz graficzny interfejs i nie chcesz używać terminala, LM Studio jest dla Ciebie. To darmowa aplikacja desktopowa na Windows, macOS i Linux, która łączy przeglądarkę modeli, chat i lokalny serwer API w jednym miejscu. Działa na silniku llama.cpp z akceleracją GPU przez CUDA, Metal, Vulkan i ROCm.

Instalacja w 5 minut

1. Wejdź na lmstudio.ai i pobierz wersję dla swojego systemu.
2. Uruchom instalator (Windows: .exe, macOS: .dmg, Linux: .AppImage).
3. Po instalacji otwórz aplikację. Główne okno podzielone jest na trzy części:
    - Model Browser — wyszukiwarka modeli z Hugging Face.
    - Chat — interfejs rozmowy z modelem.
    - Local Server — panel uruchamiania serwera API.

Ściąganie pierwszego modela

W zakładce Model Browser wpisz w wyszukiwarkę 'gemma4:4b' lub 'qwen3.6:7b'. Kliknij wybrany model, potem przycisk Download. LM Studio pobierze model w formacie GGUF. W zależności od rozmiaru modelu i prędkości internetu zajmie to od kilku sekund do kilku minut.

Po pobraniu wróć do zakładki Chat. Po prawej stronie ekranu zobaczysz panel Model. Wybierz z listy pobrany model. Kliknij Load. Po załadowaniu (zwykle 2–10 sekund) możesz zacząć rozmowę.

Ustawienia GPU i wydajności

W panelu Model przed załadowaniem możesz dostosować kluczowe parametry:

- GPU Offloading — suwak określający, ile warstw modelu ma działać na GPU. Dla NVIDIA z 8 GB VRAM ustaw 100% dla modeli 7B. Dla 4 GB GPU ustaw 50–70%.
- Context Length — długość kontekstu. Dla modeli 7B z 8 GB VRAM ustaw 4096–8192. Więcej = więcej pamięci.
- Flash Attention — włącz, jeśli obsługiwane. Zmniejsza zużycie pamięci i przyspiesza inferencję.

Jeśli model nie działa płynnie:

Zmniejsz długość kontekstu (np. z 8192 do 4096).

Zmniejsz GPU Offloading (niektóre warstwy na CPU).

Wybierz model o niższej kwantyzacji (Q4 zamiast Q8).

Sprawdź w menedżerze zadań, czy GPU jest używane.

LM Studio jako serwer API

LM Studio może działać jako serwer zgodny z API OpenAI. Przejdź do zakładki Local Server, wybierz model, który ma być serwowany, i kliknij Start. Twój model jest teraz dostępny na:

http://localhost:1234/v1

Możesz używać go z dowolnego narzędzia obsługującego OpenAI API — wystarczy zmienić adres URL i klucz API (może być dowolny).

LM Studio vs Ollama — co wybrać?

Kryterium	Ollama	LM Studio
Interfejs	Terminal / API	Graficzny (GUI)
Instalacja	Bardzo łatwa	Łatwa
Wydajność	Identyczna (oba używają llama.cpp)	Identyczna
Modele cloudowe	Potencjalnie (przyszłość)	Nie
RAG (dokumenty)	Przez Open WebUI	Wbudowany
MCP / agenci	Przez API	Wbudowane wsparcie
API OpenAI	http://localhost:11434/v1	http://localhost:1234/v1
Najlepszy dla	Programistów, automatyzacji	Początkujących, GUI

Prawda jest taka, że nie musisz wybierać. Możesz mieć oba narzędzia na jednym komputerze — używaj LM Studio do eksperymentów i szybkich testów, a Ollamy do automatyzacji i długotrwałych zadań.

Głębiej: jak to działa?

Zarówno Ollama, jak i LM Studio korzystają z tego samego silnika — llama.cpp. To napisana w C++ biblioteka, która potrafi uruchomić modele językowe na CPU z optymalizacjami dla GPU (CUDA, Metal, Vulkan, ROCm). Gdy instalujesz Ollamę, tak naprawdę instalujesz llama.cpp z wygodnym interfejsem.

Format GGUF

GGUF (GPT-Generated Unified Format) to format plików, w którym zapisuje się modele dla llama.cpp. Zastąpił wcześniejszy GGML i jest dziś standardem lokalnego AI. Plik .gguf zawiera wagi modelu, tokenizer i metadane w jednym pliku. Możesz go pobrać z Hugging Face i uruchomić bezpośrednio.

> **Co to jest kwantyzacja w praktyce?**
>
> Model w FP16 (16-bit) waży 2 bajty na parametr. Model w Q4_K_M waży 0.5 bajta na parametr. Różnica w jakości jest dla większości zastosowań niezauważalna, ale model zajmuje 4x mniej pamięci. Kwantyzacja Q8 to ~1 bajt na parametr — kompromis między jakością a rozmiarem.

Stos oprogramowania lokalnego AI

W 2026 roku ekosystem narzędzi AI układa się w wyraźną hierarchię. Zrozumienie jej pomoże Ci wybrać właściwe narzędzie do zadania:

Warstwa	Przykłady	Dla kogo	Zastosowanie
Aplikacje GUI	LM Studio, Open WebUI, Jan	Każdego	Chat z modelem, dokumenty
Menedżery modeli	Ollama	Programistów	CLI, API, automatyzacja
Silnik inferencyjny	llama.cpp, MLX, PyTorch	Zaawansowanych	Podstawowa wydajność
Serwery produkcyjne	vLLM, SGLang, TGI	DevOps, firmy	Duże obciążenie, wiele zapytań

Warstwa	Przykłady	Dla kogo	Zastosowanie
Frameworki agentów	LangChain, CrewAI, AutoGen	Deweloperów AI	Łańcuchy narzędzi, RAG

Dla 90% użytkowników wystarczy pierwsza lub druga warstwa. Nie musisz znać szczegółów działania llama.cpp, by efektywnie używać lokalnego AI. Ale gdy coś nie działa — znajomość tej struktury pomoże Ci znaleźć przyczynę.

Open WebUI — ChatGPT na swoim komputerze

Open WebUI to interfejs webowy dla Ollamy, który wygląda i działa jak ChatGPT, ale wszystko działa lokalnie. Oferuje zarządzanie rozmowami, historie, wyszukiwanie, wgrywanie plików i obsługę wielu modeli jednocześnie.

Instalacja przez Dockera

Najprostszy sposób (wymaga zainstalowanego Dockera):

docker run -d -p 3000:8080 -v open-webui:/app/backend/data --name open-webui ghcr.io/open-webui/open-webui:main

Po uruchomieniu otwórz przeglądarkę na http://localhost:3000. Zarejestruj się (pierwsze konto to admin) i gotowe. Open WebUI automatycznie wykryje działającą Ollamę na porcie 11434.

Jeśli nie masz Dockera, możesz uruchomić Open WebUI bezpośrednio przez Pythona:
pip install open-webui; open-webui serve

Co daje Open WebUI?

- Interfejs znany z ChatGPT — rozmowy, historie, edycja wiadomości.
- RAG — wgrywaj PDF-y, obrazy, dokumenty, a model odpowiada na ich podstawie.
- Zarządzanie modelami — przełączanie między modelami w trakcie rozmowy.
- Wyszukiwanie w internecie — łączność z Google (przez klucz API).
- Współdzielenie rozmów — linki do rozmów dla innych użytkowników.

API i integracje

Zarówno Ollama, jak i LM Studio oferują API w pełni zgodne z OpenAI. Oznacza to, że możesz zamienić adres api.openai.com na localhost, a twoje dotychczasowe skrypty będą działać bez zmian.

Test API przez curl

Ollama (domyślnie port 11434):

curl http://localhost:11434/v1/chat/completions -d '{ "model": "llama5", "messages": [{"role": "user", "content": "Cześć!"}] }'

LM Studio (domyślnie port 1234):

curl http://localhost:1234/v1/chat/completions -d '{ "model": "gemma4:4b", "messages": [{"role": "user", "content": "Cześć!"}] }'

Z Pythona

Zarówno Ollama, jak i LM Studio mają oficjalne biblioteki Python:

```
# pip install ollama
from ollama import chat
response = chat(model='llama5', messages=[{'role': 'user', 'content': 'Cześć!'}])
print(response.message.content)
```

Z JavaScript / Node.js

```
npm i ollama
import { Ollama } from 'ollama'; const ollama = new Ollama(); const r = await ollama.chat({ model: 'llama5', messages: [{ role: 'user', content: 'Cześć!' }] }); console.log(r.message.content);
```

Z dowolnego programu

Każde narzędzie obsługujące OpenAI API może korzystać z lokalnego modelu. Wystarczy ustawić:

- base_url = http://localhost:11434/v1 (Ollama) lub http://localhost:1234/v1 (LM Studio).
- api_key = dowolny string (lub 'ollama' dla Ollamy, 'lm-studio' dla LM Studio).

Przykładowo: w rozszerzeniu Continue (VS Code) lub narzędziach takich jak Cline, Open Interpreter, MCP — po prostu zmieniasz adres API na lokalny.

Zaawansowane: co dalej?

Gdy opanujesz podstawy, otwiera się cały świat możliwości. Oto kierunki, w które warto pójść:

RAG — rozmowa z własnymi danymi

RAG (Retrieval-Augmented Generation) pozwala modelowi odpowiadać na pytania na podstawie twoich dokumentów. LM Studio ma wbudowaną funkcję „dodaj plik". Open WebUI obsługuje RAG przez wgranie PDF. Do zaawansowanych zastosowań polecam LangChain z lokalnym embedding modelem (np. nomic-embed-text).

Agenci AI i MCP

Model Context Protocol (MCP) to standard łączenia modeli z narzędziami. LM Studio od 2026 roku obsługuje MCP natywnie. Możesz podłączyć czytnik plików, przeglądarkę, kalkulator — a model sam zdecyduje, kiedy ich użyć. To krok w kierunku autonomicznych agentów.

Wiele modeli jednocześnie

Możesz uruchomić kilka modeli równolegle, jeśli masz wystarczająco pamięci. Przykładowo: mały model (3B) do szybkich odpowiedzi i duży (30B) do złożonych zadań. Open WebUI pozwala przełączać modele w trakcie rozmowy.

Własne Modelfile (Ollama)

Ollama pozwala tworzyć własne konfiguracje modeli przez Modelfile. Możesz zmienić parametry domyślne (temperatura, kontekst), dodać system prompt, a nawet połączyć modele:

```
FROM llama5
SYSTEM "Jesteś pomocnym asystentem, który zawsze odpowiada po polsku."
PARAMETER temperature 0.7
PARAMETER num_ctx 8192
Potem: ollama create moj_model -f Modelfile
```

Uruchamianie bez GUI (headless)

Zarówno Ollama (ollama serve), jak i LM Studio (lmster) mogą działać jako usługi systemowe bez interfejsu graficznego. Idealne na serwer, Raspberry Pi 5 (modele 1–3B) lub minikomputer.

Najczęstsze problemy i rozwiązania

Model nie chce się uruchomić

Przyczyna: Brak pamięci. Model nie mieści się w VRAM/RAM.

- Rozwiązanie: Sprawdź rozmiar modelu kwantyzacją. Wybierz Q4 zamiast Q8 lub mniejszy model (7B zamiast 30B).
- Zamknij inne programy (przeglądarka, Visual Studio Code).

Bardzo wolne odpowiedzi

Przyczyna: Model działa na CPU zamiast na GPU.

- Rozwiązanie: W LM Studio zwiększ GPU Offloading do 100%.
- W Ollamie sprawdź, czy wykryto GPU: ollama ps — kolumna PROCESSOR pokaże GPU.
- Dla Apple Silicon: upewnij się, że masz Ollamę 0.19+ z MLX.

Błąd 'CUDA error: out of memory'

Przyczyna: Model jest większy niż VRAM karty.

- Rozwiązanie: Zmniejsz kontekst (np. z 8192 do 2048).
- Użyj niższej kwantyzacji (Q4 zamiast Q8).
- Włącz GPU Offloading częściowy — część modelu na CPU.

Ollama nie widzi GPU

Sprawdź wersję sterowników NVIDIA (min. 545+). Dla AMD: zainstaluj ROCm 7.2+. Dla macOS: systemowe sterowniki Metal działają automatycznie.

Jak sprawdzić wydajność?

Użyj narzędzi do benchmarkingu:

ollama run llama5 --verbose

Na końcu odpowiedzi zobaczysz podsumowanie: eval rate w tokenach na sekundę. Dla porównania:

- Poniżej 10 t/s — bardzo wolno, model za duży na ten sprzęt.
- 10–25 t/s — użytecznie, ale widać opóźnienie.
- 25–50 t/s — komfortowa rozmowa.
- Powyżej 50 t/s — szybko, nie zauważasz opóźnień.

Bezpieczeństwo przede wszystkim:

Lokalne AI jest z natury bezpieczne — dane nie opuszczają twojego komputera.

Jeśli udostępniasz API lokalnego modelu w sieci, zabezpiecz go (VPN, firewall, lub klucz API).

Pobieraj modele tylko z zaufanych źródeł: oficjalne repozytoria Ollamy, Hugging Face od zaufanych autorów.

Co dalej? Ścieżka nauki

Gratulacje! Właśnie uruchomiłeś własne, lokalne AI. Niezależnie, czy wybrałeś Ollamę z terminalem, LM Studio z guzikami, czy Open WebUI z interfejsem webowym — twój komputer jest teraz samodzielną platformą AI. Oto sugerowana droga na najbliższe dni i tygodnie:

Dzień 1: Podstawy

- Zainstaluj Ollamę lub LM Studio (albo oba).
- Ściągnij model 3–8B i porozmawiaj z nim przez chwilę.
- Spróbuj różnych zadań: napisz e-mail, wygeneruj pomysły, poproś o wyjaśnienie koncepcji.

Dzień 2–3: Testowanie modeli

- Ściągnij 3–4 różne modele (Llama 5, Mistral Large 4, Qwen 3.6, Gemma 4).
- Porównaj ich odpowiedzi na to samo pytanie — zauważysz różnice w stylu i dokładności.
- Sprawdź wydajność: ile tokenów na sekundę osiągasz?

Dzień 4–7: Integracje

- Uruchom lokalny serwer API i podłącz go do ulubionego narzędzia (np. Continue w VS Code).
- Zainstaluj Open WebUI i wgraj PDF-a — przetestuj RAG.
- Napisz prosty skrypt w Pythonie/JS, który używa lokalnego modelu przez API.

Tydzień 2+: Głębiej

- Stwórz własny Modelfile z niestandardowym system promptem.
- Spróbuj uruchomić większy model (30B+), jeśli masz odpowiedni sprzęt.
- Poznaj kwantyzację — pobierz ten sam model w różnych wariantach (Q4, Q8, FP16) i porównaj.

- Zainteresuj się agentami AI: Cline, Open Interpreter, AutoGPT z lokalnym modelem.

I najważniejsze: pamiętaj, że lokalne AI to nie tylko oszczędność pieniędzy. To wolność. Twoje dane nie opuszczają twojego komputera. Nie ma cenzury, limitów ani ukrytych kosztów. Model działa, gdy ty tego chcesz, i milczy, gdy go wyłączysz. To twoje AI.

Rozdział 10. Zastosowania AI w praktyce

Poprzednie rozdziały pokazały Ci, jak wybrać sprzęt, zainstalować narzędzia i uruchomić pierwsze modele. Teraz nadszedł czas na praktykę. Ten rozdział to przewodnik po realnych zastosowaniach lokalnego AI w 2026 roku. Każdy podrozdział opisuje konkretne zadanie, narzędzia, których potrzebujesz, i model, który się do niego najlepiej nadaje. Niezależnie, czy jesteś programistą, pisarzem, studentem, czy przedsiębiorcą — znajdziesz tu coś dla siebie.

POZIOM 1

Dla laika

Poznasz konkretne zastosowania lokalnego AI: od asystenta programisty, przez pracę z dokumentami, po edukację i twórczość.

Asystent programisty

Programowanie to obszar, w którym lokalne AI błyszczy najbardziej. Modele kodowania w 2026 roku osiągnęły poziom, w którym potrafią generować, refaktorować i debugować kod z dokładnością bliską komercyjnym asystentom w chmurze, ale z pełną prywatnością i bez abonamentu.

Continue.dev — lokalny asystent w VS Code

Continue to rozszerzenie dla Visual Studio Code i JetBrains, które zamienia lokalny model w twojego asystenta kodowania. Działa z Ollamą i LM Studio: wybierasz model, konfigurujesz adres API i kodujesz. Continue oferuje autouzupełnianie, generowanie funkcji, refaktoring, dokumentację i czat z kontekstem całego projektu.

Konfiguracja zajmuje 2 minuty:

- Zainstaluj Continue z marketplace VS Code.
- W ustawieniach wybierz provider 'Ollama' lub 'LM Studio'.
- Wybierz model — Qwen3-Coder-Next lub DeepSeek V4-Flash świetnie się sprawdzają.

Continue automatycznie indeksuje twój projekt i używa kontekstu z otwartych plików, co daje odpowiedzi dopasowane do twojego kodu. Badania z 2026 roku pokazują, że lokalne modele kodowania z RAG (Continue indeksuje bazę kodu) osiągają obniżkę błędów generacji aż o 40%.

Cline — autonomiczny agent kodowania

Cline idzie o krok dalej niż Continue. To agent AI, który ma dostęp do terminala, systemu plików i przeglądarki. Może samodzielnie tworzyć pliki, uruchamiać komendy i testować kod. Działa z Ollamą i LM Studio. W 2026 roku Cline jest jednym z najpopularniejszych narzędzi typu 'coding agent' dla lokalnych modeli.

Przykładowe zadanie: powiedz Cline 'Stwórz aplikację do notatek w React z zapisem do plików JSON'. Agent napisze komponenty, skonfiguruje projekt, zainstaluje zależności i uruchomi aplikację.

Który model do kodowania?

Model	Rozmiar	Min. RAM	Mocne strony	Zastosowanie
Qwen3-Coder-Next	3B*	8 GB	Szybki, 80B MoE, Apache 2.0	Codzienne kodowanie
Qwen 3.6 Coder 7B	7B	8 GB	Python, TypeScript	Autouzupełnianie

Model	Rozmiar	Min. RAM	Mocne strony	Zastosowanie
DeepSeek V4-Flash 14B	14B	16 GB	Refaktoring, debugowanie	Agent kodowania
DeepSeek R2 14B	14B	16 GB	Rozumowanie, debug	Złożone problemy
Llama 5 70B	70B	32 GB+	Jakość GPT-4	Krytyczne projekty

* DeepSeek V4-Flash używa architektury MoE: ~284B parametrów całkowitych, ale tylko ~13B aktywowanych na zapytanie. Dzięki temu działa szybko nawet na laptopach.

Praca z dokumentami i RAG

Retrieval-Augmented Generation (RAG) to technika, która pozwala modelowi odpowiadać na pytania na podstawie twoich własnych dokumentów. Model nie musi być trenowany na tych danych — wystarczy, że ma dostęp do ich treści w momencie odpowiedzi. To najważniejsza umiejętność praktyczna lokalnego AI w 2026 roku.

Jak działa RAG?

1. Dzielisz dokument na fragmenty (chunki). 2. Każdy fragment zamieniasz na embedding (wektor liczb). 3. Zapisujesz wektory w bazie wektorowej (ChromaDB, FAISS). 4. Gdy zadajesz pytanie, system szuka najbardziej podobnych fragmentów. 5. Znalezione fragmenty dokleja do promptu jako kontekst.

Open WebUI — RAG dla każdego

Najprostsza droga do RAG: uruchom Open WebUI (patrz rozdział 9), przeciągnij PDF do okna czatu i zadaj pytanie. Open WebUI automatycznie indeksuje dokument i używa go jako kontekstu. Obsługuje PDF, Word, obrazy (OCR), kod źródłowy i pliki tekstowe.

Zastosowania RAG â€“ przykłady

Analiza kontraktów i dokumentów prawnych

Wgraj 100-stronicową umowę i zapytaj: 'Jakie są klauzule kar umownych?'. Model znajdzie odpowiednie fragmenty i podsumuje. Badania pokazują skrócenie czasu analizy kontraktów o 75%.

Badania naukowe

Wgraj 20 artykułów naukowych w PDF. Zapytaj: 'Które badania potwierdzają skuteczność metody X?'. Model przeszuka wszystkie dokumenty i poda odpowiedzi z cytowaniami.

Wewnętrzna dokumentacja firmy

Zaindeksuj polityki, procedury i instrukcje. Pracownicy mogą pytać: 'Jaka jest procedura urlopowa na B2B?' i otrzymać odpowiedź z konkretnego fragmentu dokumentacji. Zero ryzyka wycieku danych do chmury.

Analiza finansowa

Wgraj raporty kwartalne. Zapytaj: 'Jak zmieniła się marża brutto w ostatnich trzech kwartałach?'. Model znajdzie odpowiednie liczby w dokumentach.

RAG vs kontekst modelu

Modele lokalne mają ograniczoną długość kontekstu (zwykle 8K–128K tokenów). RAG rozwiązuje ten problem: indeksujesz miliony tokenów dokumentów, a do promptu trafiają tylko najbardziej istotne fragmenty. Dzięki RAG twój model „zna" więcej, niż mieści się w jego kontekście.

Edukacja i nauka

Lokalne AI to doskonałe narzędzie edukacyjne. Działa offline, nie wymaga łącza internetowej, nie śledzi postępów i może być dostosowane do indywidualnego tempa nauki. OECD w raporcie z 2026 roku podkreśla, że AI w edukacji jest najskuteczniejsze, gdy jest używane z intencją pedagogiczną — a nie jako maszynka do gotowych odpowiedzi.

Korepetytor AI

Wyobraź sobie, że uczysz się fizyki kwantowej. Zamiast czytać podręcznik, prosisz model: 'Wytłumacz zasadę superpozycji tak, jakbym miał 12 lat. Potem jako student fizyki. Potem podaj matematyczny formalizm'. Model dostosowuje poziom do Twoich potrzeb w czasie rzeczywistym.

Nauka języków

Ustaw model z system promptem: 'Jesteś native speakerem angielskiego. Rozmawiaj ze mną na dowolny temat, ale poprawiaj moje błędy gramatyczne po każdej odpowiedzi'. Możesz też poprosić o tłumaczenie, wyjaśnienie idiomów lub konwersację w konkretnej roli (rozmowa kwalifikacyjna, negocjacje, small talk).

Pomoc w pisaniu prac

Model nie napisze pracy za Ciebie (to byłoby nieetyczne), ale może pomóc w:

- Generowaniu konspektu na podstawie tematu.
- Poprawianiu stylu i gramatyki.
- Proponowaniu źródeł i argumentów.
- Tłumaczeniu cytatów z języka obcego.

Symulacje i case studies

Poproś model: 'Symuluj rozmowę kwalifikacyjną na stanowisko młodszego programisty. Zadawaj mi pytania techniczne i komunikacyjne. Po każdej odpowiedzi daj feedback'. To bezpieczne środowisko do ćwiczeń bez presji.

Twórczość i content

Modele językowe to nie tylko kod i dokumenty. Są doskonałymi narzędziami wspomagającymi twórczość, które — w przeciwieństwie do rozwiązań chmurowych — nie analizują twojego stylu ani nie przechowują historii.

Pisanie i redakcja

- Generowanie wersji wstępnych artykułów, postów, newsletterów.
- Burza mózgów — poproś o 20 pomysłów na temat wpisu blogowego.
- Poprawa stylu: 'Przepisz ten akapit w tonie formalnym / potocznym / marketingowym'.
- Generowanie meta opisów, nagłówków, wariantów tekstów A/B.

Kreatywna współpraca

Modele językowe sprawdzają się jako partner do twórczego myślenia. Możesz poprosić o:

- Wymyślenie nazwy dla produktu lub firmy.
- Napisanie szkicu scenariusza.
- Stworzenie opisu postaci do gry RPG.
- Ułożenie wiersza lub tekstu piosenki w zadanym stylu.

POZIOM 2

Głębiej

Agenci AI, multimodalność, smart home, zastosowania biznesowe i porównanie lokalnego AI z chmurą.

Agenci AI i automatyzacja

Agenci AI to najbardziej ekscytujący obszar lokalnego AI w 2026 roku. W przeciwieństwie do prostego czatu, agenci mają dostęp do narzędzi: mogą uruchamiać kod, przeglądać pliki, szukać w internecie, a nawet kontrolować przeglądarkę. Działają autonomicznie, wykonując wieloetapowe zadania.

Open Interpreter

Open Interpreter to agent, który ma dostęp do twojego terminala. Możesz mu powiedzieć: 'Przeanalizuj plik CSV ze sprzedażą, znajdź miesiąc z największą sprzedażą, wygeneruj wykres i zapisz go jako PNG'. Agent napisze kod w Pythonie, uruchomi go, poprawi błędy i zwróci wynik. Działa z Ollamą jako backendem.

AutoGPT i BabyAGI

Te narzędzia dzielą złożone cele na podzadania i wykonują je sekwencyjnie. Przykład: 'Zbadaj rynek konkurencji dla aplikacji do notatek'. Agent wyszuka informacje, zapisze je w pliku, podsumuje i przygotuje raport. W 2026 roku te narzędzia działają stabilnie z lokalnymi modelami.

MCP — Model Context Protocol

MCP to standard łączenia modeli z zewnętrznymi narzędzami. LM Studio i Claude Desktop obsługują MCP natywnie. Przykładowe narzędzia MCP:

- Czytnik plików — model może otworzyć i przeanalizować dowolny plik.
- Kalkulator — precyzyjne obliczenia matematyczne.
- Przeglądarka — model może szukać w internecie.
- System plików — tworzenie, edycja i organizacja plików.

Lokalni agenci w 2026 roku — stan faktyczny

Agenci AI to wciąż technologia w fazie rozwoju. Modele 7–14B radzą sobie z prostymi zadaniami. Do złożonych, wieloetapowych operacji potrzebujesz modeli 30B+ lub cloud. Zaleta lokalnych agentów: pełna kontrola nad tym, co agent robi, i zero ryzyka wycieku danych.

Multimodalność: obrazy, dźwięk, wideo

Lokalne AI w 2026 roku nie ogranicza się do tekstu. Modele multimodalne (vision-language models) potrafią analizować obrazy, opisywać zdjęcia, odpowiadać na pytania o treść wizualną, a nawet transkrybować mowę.

Analiza obrazów

Modele Gemma 4, Llama 4 Scout Vision i Qwen 3.6-VL potrafią analizować obrazy. Przykładowo: wgraj zdjęcie wykresu i zapytaj 'Jaki był trend sprzedaży w Q3?'. Model odczyta dane z obrazu. Lub: 'Opisz to zdjęcie szczegółowo dla osoby niewidomej'.

Transkrypcja mowy

Whisper (OpenAI) to model rozpoznawania mowy, który działa w pełni lokalnie. Możesz transkrybować nagrania spotkań, wywiady, notatki głosowe. Whisper jest dostępny przez Ollamę (whisper:large) i jako samodzielna biblioteka Python.

Generowanie obrazów (FLUX)

Stable Diffusion to osobny ekosystem, ale warto o nim wspomnieć. ComfyUI to interfejs do generowania obrazów lokalnie. W 2026 roku modele FLUX.2 i SDXL-Ultra oferują jakość porównywalną z Midjourney, ale działają na twojej karcie graficznej. Wymagania: minimum 8 GB VRAM dla SDXL, 16 GB+ dla Flux.

Smart home i IoT

Home Assistant, najpopularniejsza platforma inteligentnego domu, w 2025 roku zintegrowała lokalne AI bezpośrednio w swoim systemie. Możesz używać lokalnego modelu językowego do sterowania domem naturalnym językiem: 'Wyłacz wszystkie światła na parterze i włącz ogrzewanie w salonie'. Home Assistant łączy się z Ollamą przez API i interpretuje komendy głosowe lub tekstowe.

Dzięki lokalnemu AI twój inteligentny dom działa w pełni offline. Sterowanie głosem, automatyzacje, analiza danych z czujników — wszystko bez wysyłania danych do chmury.

Biznes i produktywność

Lokalne AI w biznesie to przede wszystkim oszczędność kosztów i bezpieczeństwo danych. W 2026 roku coraz więcej firm decyduje się na lokalne modele do codziennej pracy, rezygnując z abonamentów ChatGPT Enterprise czy Microsoft Copilot.

Automatyzacja e-maili

- Generowanie szablonów e-maili na podstawie kontekstu.
- Klasyfikacja przychodzących wiadomości (faktura, zapytanie, reklamacja).
- Proponowanie odpowiedzi z wykorzystaniem historii korespondencji (RAG).

Analiza danych

- Zadawaj pytania w języku naturalnym o dane w plikach CSV/Excel.
- Model generuje zapytania SQL i tłumaczy wyniki na prosty język.
- Automatyczne raportowanie okresowe.

Wsparcie klienta

- Chatbot oparty na dokumentacji firmowej (RAG na politykach i procedurach).
- Model klasyfikuje zapytania i proponuje odpowiedzi.
- Pełna prywatność danych klientów — żadna rozmowa nie trafia do chmury.

Personalny asystent AI

Użyj lokalnego modelu jako swojego asystenta:

- Podsumowuj długie e-maile i dokumenty.
- Twórz notatki ze spotkań (transkrypcja + podsumowanie przez Whisper + Llama).
- Planuj dzień: 'Mam 5 zadań, uszereguj je według priorytetu i zaproponuj harmonogram'.

Lokalne AI vs chmura — kiedy które wybrać?

Lokalne AI nie jest lepsze od chmury we wszystkim. To uzupełnienie. Oto praktyczny schemat decyzyjny:

Kryterium	Lokalne AI	Chmura (ChatGPT, Claude, Gemini)
Koszt	Jednorazowy (sprzęt)	$20–200/mies. abonament
Prywatność	Pełna — dane nie opuszczają komputera	Dane trafiają na serwery
Działanie offline	Tak	Nie
Jakość odpowiedzi	70–90% cloud (zależnie od modelu)	Referencyjna
Szybkość	Zależna od sprzętu	Bardzo szybka (serwery)
Długość kontekstu	4K–128K tokenów	32K–2M tokenów
Obsługa multimediów	Ograniczona (lokalne modele vision)	Pełna (obrazy, audio, wideo)
Najlepszy dla	Prywatność, powtarzalne zadania, oszczędność	Złożone, jednorazowe zadania, maks. jakość

Optymalna strategia w 2026 roku: używaj lokalnego AI do 80% codziennych zadań (kodowanie, dokumenty, czat, analiza). Sięgaj po chmurę, gdy potrzebujesz maksymalnej jakości, bardzo długiego kontekstu, lub zaawansowanych multimediów. W ten sposób płacisz tylko za to, czego lokalnie nie da się zrobić.

Podsumowanie

Lokalne AI w 2026 roku to nie zabawka dla hobbystów. To dojrzała technologia, która w wielu obszarach dorównuje rozwiązaniom chmurowym, a w kwestiach prywatności i kosztów je przewyższa. Niezależnie od tego, czy piszesz kod, analizujesz dokumenty, uczysz się języka, czy prowadzisz firmę — znajdziesz zastosowanie, które działa lepiej lokalnie niż w chmurze.

Kluczowe wnioski z tego rozdziału:

- **Kodowanie:** Continue.dev + Qwen Coder lub Cline + DeepSeek V4-Flash to w pełni funkcjonalny lokalny asystent programisty.
- **Dokumenty:** Open WebUI z RAG zamienia twoje PDF-y w interaktywną bazę wiedzy.
- **Edukacja:** Model jako korepetytor dostosowujący się do poziomu ucznia.
- **Agenci:** Open Interpreter i MCP otwierają drzwi do autonomicznej automatyzacji.
- **Biznes:** Chatbot, analiza danych i automatyzacja e-maili w pełni lokalnie, bez comiesięcznych opłat.

W następnym, ostatnim rozdziale książki spojrzymy w przyszłość. Dokąd zmierza lokalne AI? Jakie przełomy są tuż za rogiem? I czym jest AGI, o którym wszyscy mówią?

Rozdział 11. Przyszłość AI

Doszliśmy do ostatniego rozdziału tej książki. Przeszedłeś drogę od podstaw działania AI, przez formaty, ekosystemy, wybór sprzętu, instalację narzędzi aż po praktyczne zastosowania. Teraz czas spojrzeć w przyszłość. Co przyniosą najbliższe lata? Kiedy pojawi się AGI? Czy lokalne AI zastąpi chmurę? I najważniejsze — jak się przygotować na to, co nadchodzi?

Gdzie jesteśmy w 2026 roku?

Rok 2026 to przełomowy moment w historii AI. Modelom open-source udało się zniwelować przewagę modeli zamkniętych w większości codziennych zastosowań. Qwen 3.7-Max, DeepSeek V4-Pro, Llama 5, Gemma 4 — te modele osiągają 90–98% jakości GPT-5.5 i Claude Opus 4.7 w testach, a działają lokalnie. To bez precedensu.

W 2026 roku możesz:

- Uruchomić model 7–8B na laptopie bez GPU.
- Uruchomić model porównywalny z GPT-5.5 na Mac Mini M4 Pro za $1 600.
- Kodować z asystentem AI, który nigdy nie wysyła kodu do chmury.
- Analizować dokumenty z RAG bez subskrypcji.
- Uruchomić agenta AI, który autonomicznie wykonuje zadania.

To wszystko było niemożliwe jeszcze dwa lata temu.

Główne trendy 2026–2028

1. Zanikanie różnicy między modelami open-source i zamkniętymi

Jeszcze w 2024 roku przepaść między GPT-4o a najlepszymi otwartymi modelami była ogromna. W 2026 roku otwarte modele dogoniły lub prawie dogoniły zamknięte w testach porównawczych. Trend jest jasny: każdego roku różnica maleje o połowę. W 2027–2028 roku modele open-source prawdopodobnie zrównają się z najlepszymi komercyjnymi pod każdym względem.

Oznacza to, że za 2–3 lata nie będzie praktycznego powodu, by płacić za ChatGPT czy Claude, chyba że potrzebujesz bardzo długiego kontekstu (ponad 128K tokenów) lub zaawansowanych multimediów. Do 95% codziennych zadań lokalny model w pełni wystarczy.

2. Era agentów AI

Microsoft, Google, OpenAI, Anthropic i niezliczone open-source'owe projekty inwestują w agentów AI. W 2026 roku agenci są w fazie wczesnego wdrażania. IBM przewiduje, że do 2028 roku 40% aplikacji korporacyjnych będzie zawierać wyspecjalizowanych agentów AI. MCP (Model Context Protocol) staje się standardem łączenia modeli z narzędziami.

Oznacza to, że model nie tylko odpowiada na pytania, ale działa: wysyła e-maile, aktualizuje CRM, analizuje dane, tworzy raporty. Twój komputer będzie miał swojego 'praktykanta', któremu zlecasz zadania.

3. Modele specjalizowane

Zamiast jednego modelu do wszystkiego, trend zmierza w kierunku ekosystemu wyspecjalizowanych modeli:

- Modele kodowania — wąsko wyspecjalizowane w generowaniu i analizie kodu (Qwen Coder, DeepSeek Coder).
- Modele matematyczne — do obliczeń i rozumowania (DeepSeek R2, GPT o4).
- Modele wielojęzyczne — lepsze w językach innych niż angielski.
- Modele multimodalne — rozumiejące obrazy, dźwięk, wideo.
- Modele małe (1–3B) — do szybkich, prostych zadań na słabym sprzęcie.

Twój lokalny zestaw będzie składać się z kilku modeli, które przełączasz w zależności od zadania — i robi to za ciebie menedżer modeli.

4. Hardware AI staje się mainstreamem

W 2026 roku kupujesz komputer i myślisz o AI. W 2027–2028 roku AI będzie domyślnym kryterium wyboru sprzętu. Procesory z NPU staną się standardem. Pamięć zunifikowana (jak w Apple Silicon) pojawi się w PC. Koszt 32 GB VRAM spadnie poniżej $1 000. Mac Studio z 256 GB UMA nie będzie wyjątkiem, ale normą dla profesjonalistów.

Rok	VRAM w GPU	Modele na GPU	Cena wejścia
2024	12–24 GB	7–14B Q8	$1 000+
2026	16–32 GB	30–70B Q4	$400–600
Prognoza 2028	32–64 GB	70–120B Q4	$500–800

Czy AGI jest blisko?

Sztuczna ogólna inteligencja (AGI) — maszyna, która dorówna człowiekowi w każdym zadaniu intelektualnym — to święty Graal AI. Pytanie 'kiedy AGI?' dzieli ekspertów. Najważniejsze głosy w 2026 roku:

Kto	Prognoza AGI	Uzasadnienie
OpenAI (Altman)	2027–2028	o4 przekroczyło ARC-AGI (92%). Skalowanie dalej działa.
Anthropic (Dario Amodei)	„Plausible do 2028"	Ryzyko i szansa są równie duże. Potrzebne regulacje.
DeepMind (Hassabis)	5–10 lat	Nadal brakuje rozumowania przyczynowo-skutkowego.
Eksperci (medi ankiety)	2029–2035	Średnia z ankiet wśród badaczy AI.
Sceptycy (LeCun, Chollet)	>2050 lub nigdy	Obecne modele to tylko dopasowywanie wzorców, brak prawdziwego rozumienia.

Nawet jeśli AGI pojawi się wcześniej, nie oznacza to końca świata ani natychmiastowego bezrobocia. AGI na poziomie człowieka będzie wchodziła stopniowo: najpierw jako narzędzie, potem jako partner, dopiero potem jako zastępstwo. Wciąż mamy czas, by się przygotować.

Czym różni się AGI od dzisiejszych modeli?

Dzisiejsze modele są wąscy specjaliści: świetnie przewidują następny token, ale nie mają świadomości, nie planują długoterminowo i nie rozumieją świata. AGI oznacza zdolność do przenoszenia wiedzy między dziedzinami, uczenia się nowych umiejętności bez fine-tuningu i rozumienia kontekstu tak, jak robi to człowiek.

Etyka, bezpieczeństwo i odpowiedzialność

Im więcej mocy dajemy modelom, tym większa odpowiedzialność spoczywa na nas, użytkownikach. Lokalne AI łagodzi jeden problem (prywatność danych), ale nie eliminuje innych:

Stronniczość i halucynacje

Modele językowe odzwierciedlają dane, na których były trenowane. Jeśli dane zawierają uprzedzenia, model je powieli. Halucynacje (generowanie fałszywych faktów z przekonaniem) to wbudowana cecha tych modeli, nie błąd. Zawsze weryfikuj informacje, zwłaszcza gdy dotyczą faktów, dat, liczb i cytatów.

Kontrola nad agentami

Gdy agent AI ma dostęp do twojego systemu plików, terminala i internetu, konsekwencje błędu mogą być poważne. Zawsze sprawdzaj, co zamierza zrobić agent, zanim mu na to pozwolisz. Uruchamiaj agentów w izolowanym środowisku (kontroler, oddzielne konto systemowe).

Uzależnienie i zanik umiejętności

OECD w raporcie z 2026 roku ostrzega przed 'metacognitive laziness' — lenistwem poznawczym, gdy zbyt mocno polegamy na AI. Regularnie sprawdzaj, czy nadal umiesz zrobić to, co zleciłeś AI. Traktuj modele jak kalkulatory: używaj ich do obliczeń, ale nie zapominaj, jak działa mnożenie.

Rynek pracy

AI nie tyle zabierze ci pracę, ile zmieni jej charakter. Osoby używające AI będą coraz bardziej produktywne od tych, które go nie używają. Kluczowa umiejętność najbliższych lat to nie programowanie, ale umiejętność formułowania celów i oceny wyników pracy AI.

Lokalne AI w 2028 roku — prognoza

Co zmieni się w ciągu najbliższych dwóch lat? Oto śmiała, ale realistyczna prognoza:

Modele

- Modele 1–3B będą miały jakość dzisiejszych 7–8B (postęp w destylacji).
- Modele 7–8B będą działać na zwykłych laptopach z jakością dzisiejszego GPT-5.5.
- Modele 30–70B staną się standardem na desktopach — każdy PC z 32 GB RAM je uruchomi.
- Modele 120B+ będą dostępne lokalnie dla entuzjastów (128–256 GB RAM w Apple Silicon).

Hardware

- Karty graficzne z 32 GB VRAM staną się standardem średniej półki.
- Pamięć zunifikowana (UMA) pojawi się w PC jako opcja dla AI.
- NPU w procesorach osiągnie 100+ TOPS, ale wciąż nie będzie używany przez mainstreamowe narzędzia AI.
- Cena wejścia do lokalnego AI spadnie do $200–400 za zestów zdolny do modeli 7B.

Oprogramowanie

- Ollama i LM Studio zintegrują się w jedno — lub pojawi się nowy, lepszy standard.
- Agenci AI staną się domyślnym interfejsem — zamiast czatu będziesz rozmawiać z agentem.
- RAG będzie wbudowany w system operacyjny, nie w aplikacje.
- Modele będą współdzielone między aplikacjami (jeden załadowany model obsługuje wiele programów).

Jak się przygotować na przyszłość?

1. Zbuduj nawyk używania AI

Nie czekaj, aż AI stanie się idealne. Używaj go codziennie już teraz. Każdego dnia znajdź jedno zadanie, które możesz zlecić modelowi: napisanie e-maila, burza mózgów, analiza dokumentu. Im więcej używasz, tym lepiej rozumiesz, gdzie AI pomaga, a gdzie przeszkadza.

2. Ucz się podstaw, nie konkretnych narzędzi

Konkretne narzędzia (Ollama, LM Studio, Continue) zmienią się w ciągu roku. Podstawy się nie zmienią:

- Co to jest kwantyzacja i jak wpływa na wydajność.
- Jak działa RAG i kiedy go używać.
- Różnica między modelem, silnikiem inferencyjnym i interfejsem.
- Jak oceniać jakość odpowiedzi modelu.

Te umiejętności będą aktualne za 5 i 10 lat.

3. Inwestuj w sprzęt rozsądnie

Nie kupuj najdroższego sprzętu, bo 'AI tego wymaga'. Kup to, co potrzebujesz dzisiaj, z zapasem na 1–2 lata. Ceny spadają, możliwości rosną. Mac Mini M4 Pro 48 GB to bezpieczna inwestycja na 3–4 lata. Drogie GPU kupuj tylko, jeśli wiesz, że będziesz trenować własne modele.

4. Zachowaj krytyczne myślenie

Największym zagrożeniem AI nie jest to, że model się pomyli — ale że my przestaniemy sprawdzać. Nie ufaj odpowiedziom modelu bez weryfikacji. Traktuj go jak stażystę: entuzjastycznego, szybkiego, ale często mylącego się w szczegółach. Twoja rola to być mądrzejszym przełożonym.

5. Buduj społeczność

Lokalne AI rozwija się dzięki społeczności. Dołącz do forów (r/LocalLLaMA na Reddicie, discord Ollamy), czytaj blogi, subskrybuj newslettery. Dziel się swoimi doświadczeniami — to, co odkryjesz dzisiaj, jutro może pomóc komuś innemu.

Słowo na koniec

Książkę tę napisałem, ponieważ wierzę, że AI nie powinno być zarezerwowane dla gigantów technologicznych z centrami danych wielkości boiska. AI należy do każdego, kto ma komputer. Lokalne AI to nie tylko oszczędność pieniędzy — to wolność, prywatność i kontrola.

W 2026 roku po raz pierwszy w historii każda osoba może mieć na własnym komputerze inteligencję porównywalną z najlepszymi światowymi systemami. Nie musisz mieszkać w Dolinie Krzemowej. Nie musisz mieć konta w OpenAI. Nie musisz płacić abonamentu. Wystarczy, że otworzysz terminal i wpiszesz:

```
ollama run llama5
```

To wszystko. Właśnie uruchomiłeś AI na swoim komputerze. Witaj w przyszłości.

Dziękuję, że przeczytałeś tę książkę. Mam nadzieję, że była dla Ciebie przydatna tak samo, jak dla mnie była przyjemnością w pisaniu. Lokalne AI to najlepsza rzecz, jaka przydarzyła się technologii od czasów otwartego oprogramowania. Ciesz się nią, eksperymentuj, ucz się i — przede wszystkim — używaj jej odpowiedzialnie.

Dodatek A: Słowniczek pojęć

Pojęcie	Wyjaśnienie
Agent AI	Program, który używa modelu językowego do autonomicznego wykonywania zadań z dostępem do narzędzi (plików, internetu, terminala).
API (OpenAI-compatible)	Standardowy interfejs programistyczny, który pozwala aplikacjom łączyć się z modelem. Lokalne narzędzia (Ollama, LM Studio) oferują API zgodne z OpenAI.
Context window (okno kontekstu)	Maksymalna liczba tokenów, jaką model może 'widzieć' przy generowaniu odpowiedzi. Większy kontekst = więcej miejsca na dokumenty i historię rozmowy.
CUDA	Platforma obliczeniowa NVIDIA do akceleracji GPU. Większość narzędzi AI na PC wymaga CUDA do działania na kartach NVIDIA.
Embedding (osadzenie)	Liczbowa reprezentacja tekstu (wektor), która pozwala porównywać znaczenie fragmentów. Używane w RAG do wyszukiwania podobnych dokumentów.
Fine-tuning	Dokszkolenie istniejącego modelu na własnych danych, by lepiej działał w konkretnej dziedzinie. Wymaga dużej mocy obliczeniowej.
FP16 / FP32	Formaty liczb zmiennoprzecinkowych. FP32 (32-bit) to pełna precyzja, FP16 (16-bit) to połowa pamięci przy minimalnej utracie jakości.
GGUF	Format pliku modelu dla llama.cpp. Jeden plik .gguf zawiera wagi, tokenizer i metadane. Standard lokalnego AI.
GPU offloading	Przeniesienie części lub całości obliczeń modelu na GPU. Im więcej warstw na GPU, tym szybciej działa model.
Hallucynacja	Sytuacja, w której model generuje fałszywe informacje z przekonaniem. Wbudowana cecha modeli językowych.

Pojęcie	Wyjaśnienie
Inferencja	Proces generowania odpowiedzi przez model. Różni się od trenowania — model już jest wytrenowany, tylko odpowiada.
Kwantyzacja	Kompresja wag modelu do niższej precyzji (np. Q4, Q8). Zmniejsza zapotrzebowanie na pamięć kosztem minimalnej utraty jakości.
llama.cpp	Silnik inferencyjny napisany w C++. Podstawa działania Ollamy i LM Studio. Umożliwia uruchamianie modeli na CPU i GPU.
LLM (Large Language Model)	Duży model językowy — sieć neuronowa trenowana na ogromnych ilościach tekstu, potrafiąca generować i rozumieć język naturalny.
MCP (Model Context Protocol)	Standard łączenia modeli językowych z narzędziami zewnętrznymi (plikami, API, przeglądarką). Opracowany przez Anthropic.
MLX	Framework Apple do uczenia maszynowego na Apple Silicon. Od 2026 roku używany przez Ollamę do przyspieszenia inferencji na Mac.
MoE (Mixture of Experts)	Architektura modelu, w której tylko część parametrów jest aktywowana na zapytanie. Większa wydajność przy niższym koszcie obliczeniowym.
NPU (Neural Processing Unit)	Dedykowany układ w procesorze do akceleracji AI. W 2026 roku używany głównie przez system operacyjny, nie przez narzędzia AI.
Parametr	Waga w sieci neuronowej — liczba określająca rozmiar modelu. Więcej parametrów = większe możliwości, ale więcej pamięci.
Prompt	Wejściowy tekst (polecenie, pytanie), który wysyłasz do modelu. Jakość promptu ma ogromny wpływ na jakość odpowiedzi.

Pojęcie	Wyjaśnienie
RAG (Retrieval-Augmented Generation)	Technika łączenia modelu z bazą dokumentów. Model odpowiada na podstawie znalezionych fragmentów, a nie tylko swojej wiedzy.
ROCm	Platforma obliczeniowa AMD do akceleracji GPU. Alternatywa dla CUDA, działa głównie na Linuksie.
System prompt	Początkowa instrukcja dla modelu, która określa jego zachowanie, ton i role. Niewidoczna dla użytkownika.
Token	Podstawowa jednostka tekstu dla modelu. Token to nie słowo, ale fragment (np. 'kot' to jeden token, 'kotem' to dwa). Model przewiduje następny token.
TensorRT-LLM	Silnik inferencyjny NVIDIA do maksymalnej wydajności na kartach NVIDIA. Używany w produkcji, nie na desktopie.
Tokens per second (t/s)	Podstawowa miara wydajności modelu. Liczba tokenów wygenerowanych na sekundę. 25+ t/s to komfortowa rozmowa.
UMA (Unified Memory Architecture)	Współdzielona pamięć między CPU i GPU w Apple Silicon. Pozwala uruchamiać większe modele niż na PC z osobną VRAM.
vLLM	Silnik inferencyjny do dużego obciążenia. Używany w produkcji, optymalizuje wykorzystanie pamięci przy wielu równoległych zapytaniach.
VRAM	Pamięć karty graficznej. Kluczowy zasób dla lokalnego AI — model musi się w niej zmieścić, by działać w pełni na GPU.
Whisper	Model rozpoznawania mowy OpenAI. Działa lokalnie, transkrybuje mowę na tekst z wysoką dokładnością.

Dodatek B: Porównanie modeli AI (maj 2026)

Tabela aktualna na: maj 2026. Modele i ich wersje zmieniają się szybko – przed zakupem sprawdź aktualne benchmarki na artificialanalysis.ai.

Model	Param.	Min. RAM	Kodowanie	Ogólny	Najlepszy do
Llama 5 8B	8B	8 GB	★★★	★★★	Uniwersalny, sprawdzony
Gemma 4 26B A4B	26B (4B act.)	8 GB	★★★	★★★★	Najlepszy model średniej wielk.
Mistral Large 4 7B	7B	8 GB	★★★	★★★	Szybki, niezawodny, kompatybilny
Qwen 3.6 7B	7B	8 GB	★★★	★★★	Wielojęzyczny, tani
Phi-4 14B	14B	16 GB	★★★★	★★★	Kodowanie, logika
Codestral 2	22B	16 GB	★★★★	★★☆	Kodowanie (dedykowany)
DeepSeek R2 14B	14B	16 GB	★★★	★★☆	Debugowanie, rozumowanie
DeepSeek V4-Flash	~300B*	16 GB	★★★★	★★★	Kodowanie (MoE, bardzo szybki)
Qwen 3.6 14B	14B	16 GB	★★★★	★★☆	Kodowanie (dedykowany)
GLM-5.1 24B	24B	24 GB	★★★	★★★★	Bardzo dobry ogólny
Qwen 3.7-Max	~30B act.*	24 GB	★★★	★★★★	Jakość bliska GPT-5.5 (API)

Model	Param.	Min. RAM	Kodowanie	Ogólny	Najlepszy do
Gemma 4 31B	31B	24 GB	★★★	★★★★	Wielojęzyczny, mocny
Llama 5 70B	70B	48 GB	★★★★	★★★★	Jakość GPT-5.5 na Mac
Qwen 3.7-Max	~500B MoE*	API	★★★★	★★★★★	Topowy model (API-only)
DeepSeek V4-Pro	1.6T*	128 GB+	★★★★	★★★★★	Stan wiedzy open-source
Llama 5 ~500B	~500B	256 GB+	★★★★★	★★★★★	Największy open-weight

* Codestral 2: 22B dense. DeepSeek V4-Flash: ~300B całkowitych, ~13B aktywnych (MoE). DeepSeek V4-Pro: 1.6T całkowitych, ~49B aktywnych (MoE). Llama 5: ~500B dense.

Dodatek C: Linki i zasoby

Poniższe linki były aktualne w momencie pisania książki (maj 2026). Zalecam samodzielne sprawdzenie aktualności przed skorzystaniem.

Narzędzia i aplikacje

Narzędzie	URL	Opis
Ollama	ollama.com	Najpopularniejsze narzędzie CLI do modeli językowych
LM Studio	lmstudio.ai	Graficzny interfejs do lokalnych modeli
Open WebUI	github.com/open-webui/open-webui	Interfejs webowy dla Ollamy (jak ChatGPT)
Continue.dev	continue.dev	Asystent kodowania AI do VS Code / JetBrains
Cline	github.com/cline/cline	Autonomiczny agent kodowania
Jan	jan.ai	Alternatywny interfejs desktopowy dla modeli
ComfyUI	github.com/comfyanonymous/ComfyUI	Interfejs do generowania obrazów (Stable Diffusion)
Home Assistant	home-assistant.io	Platforma smart home z integracją Ollamy
Llama.cpp	github.com/ggerganov/llama.cpp	Silnik inferencyjny (fundament)
vLLM	github.com/vllm-project/vllm	Serwer inferencyjny do dużego obciążenia

Modele i huby

Zasób	URL	Opis
Ollama Library	ollama.com/library	Oficjalna biblioteka modeli dla Ollamy
Hugging Face	huggingface.co	Największe repozytorium modeli AI
Hugging Face GGUF	huggingface.co/models?library=gguf	Modele w formacie GGUF
Artificial Analysis	artificialanalysis.ai	Porównanie wydajności modeli i hardware'u
Can I Run AI?	canirun.ai	Sprawdź, czy twój sprzęt uruchomi dany model

Społeczności i fora

Społeczność	URL / platforma	Opis
r/LocalLLaMA	reddit.com/r/LocalLLaMA	Największe forum o lokalnym AI
Discord Ollamy	discord.gg/ollama	Oficjalny serwer społeczności Ollamy
Discord LM Studio	discord.gg/lmstudio	Oficjalny serwer społeczności LM Studio
Hugging Face Forums	discuss.huggingface.co	Forum społeczności Hugging Face
Ollama GitHub	github.com/ollama/ollama	Kod źródłowy i zgłaszanie problemów

Nauka i dokumentacja

Zasób	URL	Opis
Ollama Docs	docs.ollama.com	Oficjalna dokumentacja Ollamy
LM Studio Docs	lmstudio.ai/docs	Dokumentacja LM Studio
llama.cpp Docs	github.com/ggerganov/llama.cpp	README z instrukcjami
Learn LLM	learnllm.dev	Kurs lokalnego AI
OpenAI API Docs	platform.openai.com/docs	Referencja API OpenAI (kompatybilne)
MIT SMR AI Trends	sloanreview.mit.edu	Raporty o trendach AI

Sztuczna inteligencja bez tajemnic – poradnik dla każdego

Książka napisana i wygenerowana w maju 2026 roku.

Autor: Mzdrowy

Narzędzia: Node.js, biblioteka docx, Ollama, DeepSeek V4.

Koniec książki

Sztuczna inteligencja bez tajemnic

Napisana i wygenerowana w maju 2026 roku.