**Porównanie potencjału generatywnych sieci neuronowych do generowania kodu na przykładzie aplikacji do przeprowadzania ankiet**

**Motywacja**

Sieci neuronowe oraz różnej maści oparte na nich chatboty stały się w ciągu ostatnich kilku lat istotną częścią naszego życia. Znalazły swoje zastosowanie w niemal każdej dziedzinie – od sztuki aż po nauki ekonomiczne i społeczne. Jedną z grup zawodowych wymienianych jako najbardziej prawdopodobne do zastąpienia przez „sztuczną inteligencję”, jak zwykło się zbiorczo nazywać te narzędzia, są programiści. Przyczyn takich podejrzeń jest wiele – kod programu jest dość mocno ustrukturyzowany oraz stosunkowo powtarzalny, a jego twórca osobą drogą do opłacenia. Wielu studentów kierunków powiązanych z informatyką oraz docenionych programistów używa narzędzi opartych o sieci neuronowe niemal codziennie w swojej pracy, a coraz więcej firm zachęca swoich pracowników do korzystania z nich. W związku z tym postanowiłem sprawdzić, jak popularne chatboty, czasami nawet reklamowane jako specjalnie dostosowane do potrzeb piszących kod, poradzą sobie z napisaniem stosunkowo prostej aplikacji, z rolą człowieka ograniczającą się do prostego stwierdzenia czy dany program działa zgodnie z założeniami, czy nie.

**Cel pracy**

Celem moich działań było porównanie potencjału trzech popularnych chatbotów do napisania kodu prostej aplikacji. Jedyną moją ingerencją było napisanie promptu dla sieci neuronowej, uruchomienie otrzymanego programu i ewentualne wytknięcie błędów. Chatboty, które poddałem próbie to ChatGPT firmy OpenAI, DeepSeek firmy o tej samej nazwie oraz Claude, stworzony przez Anthropic. Mierzona była przede wszystkim konieczna liczba interakcji niezbędna do osiągnięcia zakładanego celu i parametry kodu, a wszystkie badania były przeprowadzane na darmowych wersjach chatbotów.

**Przeanalizowane metody i narzędzia**

**Wykorzystane metody i narzędzia**

**Sposób rozwiązania problemu**

Zdecydowałem się przeprowadzić trzy badania na każdej z sieci:

1. Pierwsze badanie dotyczyło zdolności sieci do wygenerowania kodu od podstaw na bazie pojedynczego promptu. W tym badaniu wszystkie sieci dostawały za każdym razem dokładnie ten sam prompt w nowej konwersacji.
2. Kolejne dotyczyło zdolności sieci do wprowadzania modyfikacji do już istniejącego kodu oraz ulepszania stworzonego przez siebie programu. Tutaj prompty były tworzone na bieżąco w zależności od aktualnych problemów konkretnego kodu.
3. Ostatnie sprawdzało, czy zachęcone sieci będą zadawały pytania na temat otrzymanego promptu w celu jak najlepszego spełnienia założeń.

W każdym badaniu celem chatbotów było stworzenie prostej aplikacji do zbierania ankiet w frameworku streamlit w wersji 1.44.0. Aplikacja ta miała być wyposażona w dwa tryby, w których pokazywane byłyby dokładnie te same pytania. Pierwszy z trybów („tryb timera”), miał być wyposażony w widoczny licznik czasu, po upłynięciu którego program powinien przejść bezwzględnie do kolejnego pytania. Drugi z nich („tryb zrelaksowany”) miał być pozbawiony jakichkolwiek ograniczeń czasowych i umożliwiać swobodne poruszanie się między pytaniami, podczas gdy aplikacja miała mierzyć czas spędzony nad każdym z nich.

W pierwszym badaniu największym problemem okazało się poprawne opisanie timera, który miał przyjąć postać skracającego się stopniowo paska. Timer ten często wykazywał błędy w działaniu, do których należały zazwyczaj brak automatycznego odświeżania (często odświeżanie następowało po interakcji z elementem aplikacji) oraz restart po zmianie odpowiedzi. Po każdej z nieudanych prób modyfikowałem odpowiednio dostarczany prompt i spisywałem zalety oraz wady każdego z otrzymanych kodów.

Po trzech modyfikacjach promptów żadnej z sieci nie udało się spełnić całkowicie postawionych przed nimi założeń. Jedyną siecią, która poprawnie zaimplementowała pasek timera był DeepSeek; niestety przygotowany przez niego kod nie pokazywał odpowiedzi do pytań w pierwszej części. Zarówno ChatGPT, jak Cladue spotkały się z tym samym problemem – ich timery restartowały się po zaznaczeniu dowolnej z odpowiedzi. Dodatkowo, aplikacja przygotowana przez ChatGPT nie dokonywała pomiarów czasu spędzonego na każdym pytaniu w części „zrelaksowanej”, a aplikacja Claude potrafiła zwrócić błąd przy próbie powrotu do wcześniejszych pytań w drugiej części ankiety.

W drugim badaniu przekazałem sieciom ich kod napisany w ostatniej próbie pierwszego badania, jako że w każdym przypadku był najbardziej udany ze wszystkich. Czasami podczas badania zdarzało się, że pomimo prośby o modyfikację, sieć zwracała program cierpiący na te same problemy, jak ten dostarczony – w takiej sytuacji przede wszystkim starałem się użyć innych sformułowań. Ponieważ żadnemu z chatbotów nie udało się w pełni napisać aplikacji przy pomocy pojedynczego promptu, pierwsze modyfikacje kodu skupiały się na poprawnej implementacji podstawowej wersji. Najszybciej udało się to ChatowiGPT – całkowicie działającą aplikację otrzymałem już po drugiej modyfikacji kodu. Nieco dłużej zadanie to zajęło DeepSeekowi, który potrzebował czterech interakcji. Dużym zawodem