Akademia Tarnowska

Katedra Informatyki

Kierunek: Informatyka - Inżynieria Oprogramowania

Przedmiot: Inżynieria Oprogramowania

Semestr: 2024/2025

Kwadrat Logiczny wspomagany LLM

Autor: Szymon Pietruszka

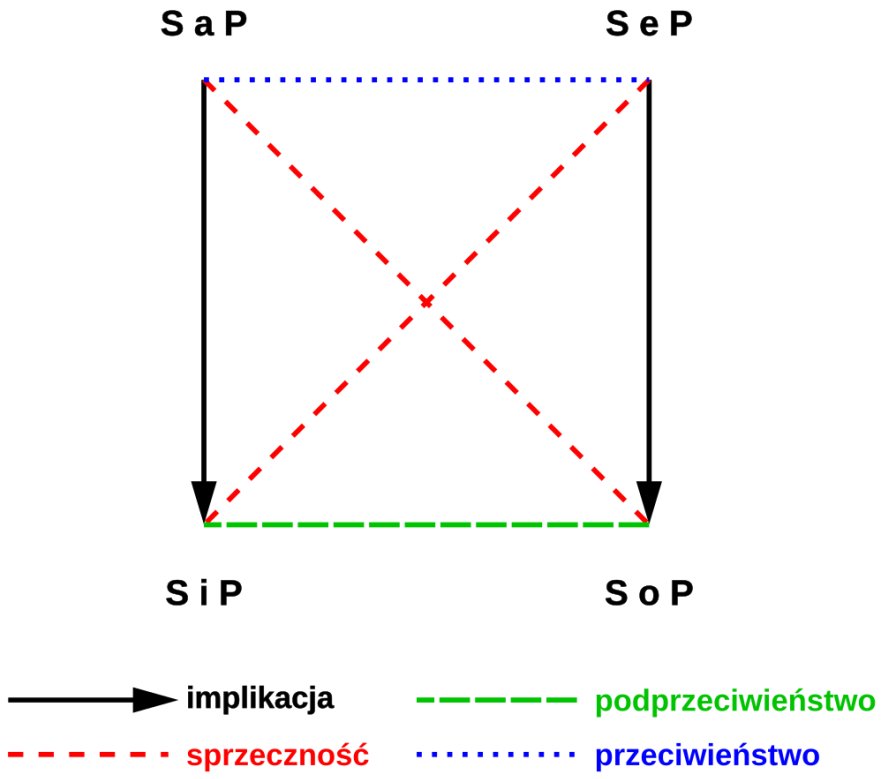
Spis Treści

Cel

Celem projektu było stworzenie kompletnego narzędzia programistycznego, wyposażonego dodatkowo w graficzny interfejs użytkownika (GUI), które umożliwiało automatyczne generowanie pełnego zestawu zdań logicznych w ramach klasycznego kwadratu logicznego (A, E, I, O) na podstawie pojedynczego zdania typu A. W projekcie zostały wykorzystane duże modele językowe (LLM) oraz różnorodne techniki promptowania, mające na celu zwiększenie trafności, spójności logicznej oraz poprawności językowej generowanych zdań.

Kwadrat logiczny

To klasyczne narzędzie używane w logice do przedstawiania relacji pomiędzy czterema typami zdań twierdzących i przeczących, znanych jako zdania kategoryczne. Został wprowadzony przez Arystotelesa i rozwijany przez średniowiecznych logików jako sposób na analizę wzajemnych powiązań logicznych między różnymi formami wypowiedzi o rzeczywistości.



Kwadrat obejmuje cztery podstawowe typy zdań:

- **A** – zdanie ogólne twierdzące, np. *„Wszyscy ludzie są śmiertelni.”*
- **E** – zdanie ogólne przeczące, np. *„Żaden człowiek nie jest nieśmiertelny.”*
- **I** – zdanie szczegółowe twierdzące, np. *„Niektórzy ludzie są mądrzy.”*
- **O** – zdanie szczegółowe przeczące, np. *„Niektórzy ludzie nie są mądrzy.”*

Te zdania są połączone czterema klasycznymi relacjami logicznymi:

1. **Kontradykcja (A–O i E–I)** – zdania wzajemnie się wykluczają; nie mogą być jednocześnie prawdziwe ani jednocześnie fałszywe.
2. **Kontrarność (A–E)** – oba nie mogą być jednocześnie prawdziwe, ale mogą być jednocześnie fałszywe.
3. **Subkontrarność (I–O)** – oba nie mogą być jednocześnie fałszywe, ale mogą być jednocześnie prawdziwe.
4. **Subalternacja (A→I i E→O)** – prawdziwość zdania ogólnego pociąga za sobą prawdziwość zdania szczegółowego, ale nie odwrotnie.

Kwadrat logiczny służy nie tylko do analizy poprawności logicznej zdań, ale także do eksploracji ich wzajemnych zależności semantycznych. Jest używany w filozofii, lingwistyce, sztucznej inteligencji i teorii argumentacji jako narzędzie do rozumienia struktury i konsekwencji wypowiedzi językowych.

LLM

W projekcie zostały wykorzystane duże modele językowe, takie jak LLaMA, Gemini oraz modele OpenAI, w celu porównania ich skuteczności w generowaniu logicznych zdań uzupełniających w ramach kwadratu logicznego. Każdy z tych modeli był testowany z użyciem różnych technik promptowania. Analiza objęła zarówno jakość generowanych zdań, jak i ich zgodność z formalnymi zasadami logiki kategorycznej. Dzięki temu możliwe było określenie, które modele najlepiej rozumieją strukturę logiczną języka oraz które strategie promptowania okazały się najskuteczniejsze.

Prompt

Użytkownik wprowadza jedno zdanie (ręcznie lub poprzez plik .txt), które stanowi punkt wyjściowy – lewy górny wierzchołek kwadratu logicznego (zdanie typu A). Następnie wybierana jest technika promptowania..

Na podstawie tego zdania program generuje odpowiedni prompt zawierający:

- Dziedzinę, z której pochodzi zdanie
- Schemat konstrukcyjny czterech wierzchołków kwadratu
- Opis relacji logicznych między wierzchołkami (np. sprzeczność, przeciwieństwo, kontrarność, kontradyktoryczność)

Wybrany model językowy generuje brakujące wierzchołki (E, I, O). W zależności od zastosowanej techniki promptowania, struktura promptu może się różnić, ale powyższe elementy są wspólne dla wszystkich wariantów.

Techniki:

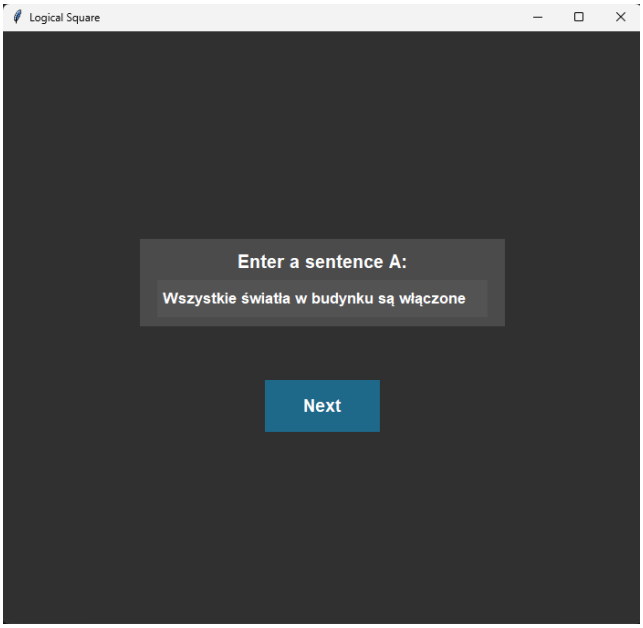
Wykorzystuje nowoczesne techniki promptowania, takie jak:

- zero-shot
- one-shot
- few-shot
- chain-of-thought
- self-consistency
- ReAct
- HyDE + RAG

GUI

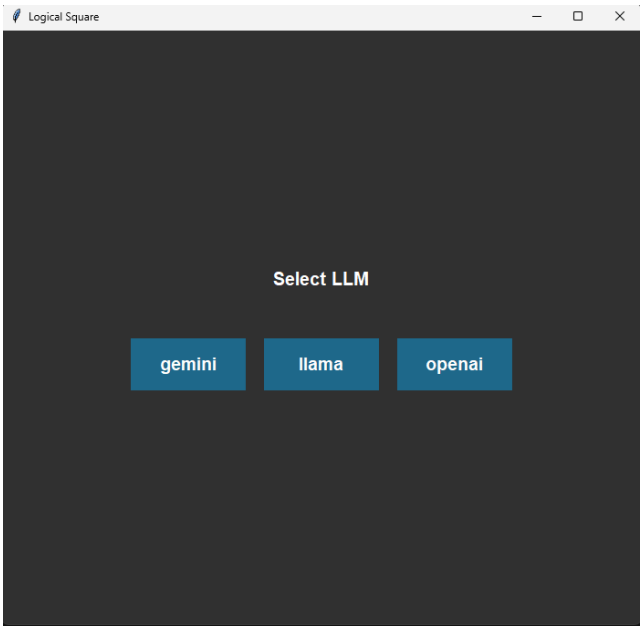
1. **Wprowadzenie zdania typu A**

Na początku użytkownik wprowadza zdanie kategoryczne typu A (np. Wszystkie światła w budynku są włączone), które będzie stanowić punkt wyjścia do wygenerowania pozostałych zdań kwadratu logicznego.



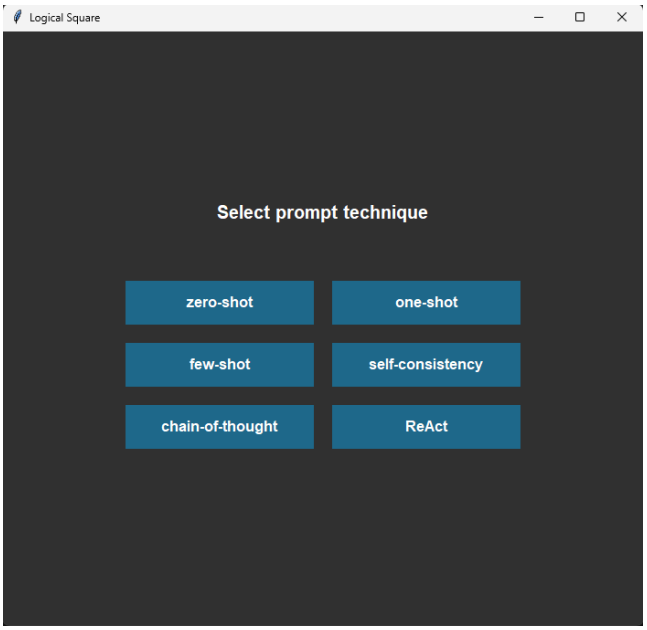
2. **Wybór modelu językowego (LLM)**

W kolejnym kroku użytkownik wybiera model językowy, który zostanie wykorzystany do generacji zdań – do dyspozycji są LLaMA, Gemini oraz OpenAI.

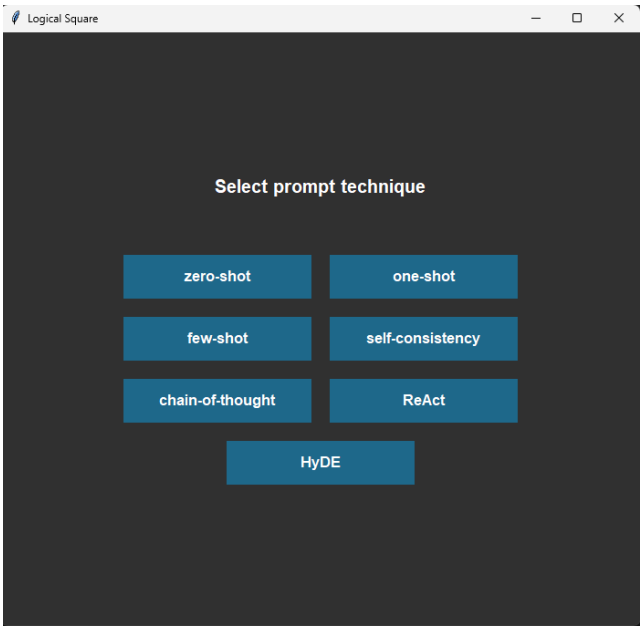


3. **Wybór techniki promptowania**

Następnie użytkownik wybiera technikę promptowania, która zostanie zastosowana do sformułowania zapytania do modelu.

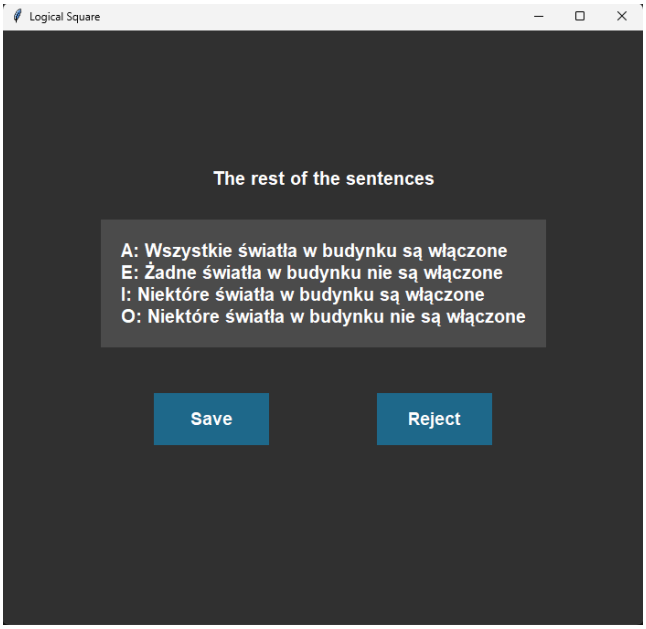


W przypadku modelu LLaMA dostępna jest dodatkowa opcja wykorzystania techniki RAG.



4. **Generacja i ocena wyników**

Po wygenerowaniu wyników przez wybrany model, aplikacja prezentuje komplet zdań w układzie kwadratu logicznego. Użytkownik może zaakceptować wynik (co powoduje zapis do pliku `.txt`) lub odrzucić go i ponowić generację wyników.



Main

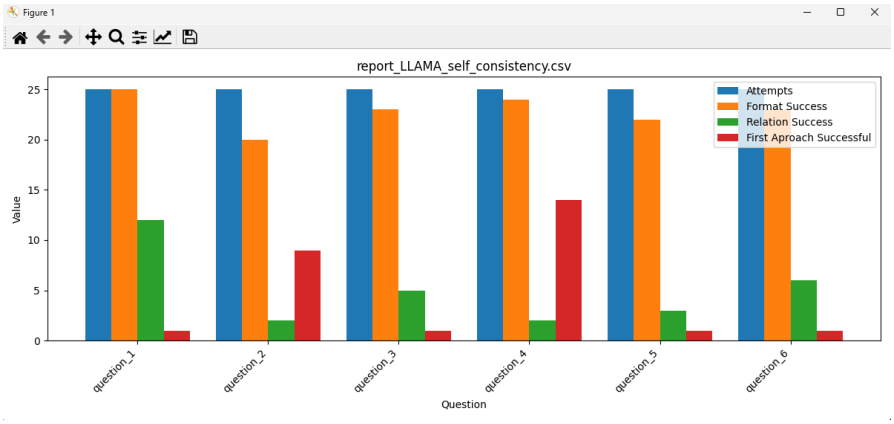
W projekcie oprócz GUI dostępna jest także wersja konsolowa w pliku **`main`**. Dzięki temu możliwe jest korzystanie z pełnej funkcjonalności aplikacji również w środowiskach bez graficznego interfejsu.

Testy i raporty

Po uruchomieniu pliku **`run.py`** znajdującego się w folderze **tests**, użytkownik ma możliwość przeprowadzania testów działania systemu generowania zdań logicznych. Proces ten przebiega w kilku etapach:

A. **Przegląd istniejących raportów:**
   - Użytkownik może zdecydować, czy chce zobaczyć podsumowanie wszystkich dotychczas zapisanych raportów znajdujących się w folderze **reports**

- Wyniki są prezentowane w formie wykresów słupkowych opisanych jako:
  – łączną liczbę przeprowadzonych testów
  – liczbę poprawnie wygenerowanych zdań (A, E, I, O)
  – liczbę testów zakończonych sukcesem
  – średnią liczbę prób do uzyskania pierwszego poprawnego wyniku



B. Nowy test:

- Użytkownik wybiera model językowy lub RAG, który ma zostać przetestowany
- Następnie wybierana jest technika promptowania

C. Testy:

- Użytkownik decyduje czy chce przeprowadzić nowe testy
- W przypadku potwierdzenia podaje liczbę testów, które mają zostać wykonane
- Zdania typu A są pobierane z plików **question_nr.txt** w folderze **questiuons**
- Po zakończeniu testowania, w konsoli wyświetlane są szczegółowe wyniki nowo utworzonego raportu
- Wyniki są zapisywane w pliku **csv** oraz w **pdf**

D. Wykonanie testów:

- Cały proces można powtarzać wielokrotnie, testując różne modele i techniki
- Zakończenie działania programu następuje w momencie, gdy użytkownik przy wyborze modelu LLM wpisze 0

Podsumowanie

Projekt ten pokazuje, w jaki sposób sztuczna inteligencja może wspierać klasyczne obszary wiedzy, takie jak logika formalna.

Dzięki wykorzystaniu dużych modeli językowych oraz różnych technik promptowania udało się stworzyć narzędzie, które automatycznie generuje komplet zdań kwadratu logicznego na podstawie jednego zdania typu A. Aplikacja została wyposażona zarówno w graficzny interfejs użytkownika (GUI), jak i wersję konsolową, co zwiększa jej dostępność i elastyczność.

Dodatkowo system testujący umożliwia ocenę skuteczności różnych modeli i metod generowania, wspierając dalszy rozwój i optymalizację rozwiązania.