Explicarea pe bază de prompt engineering a explicațiilor date de SHAP, LIME sau alte astfel de metode

Partea I - Context. Metode Existente.

Arădoaie Ioana-Maria, Toader Ana-Maria, Mereu Ioan-Flaviu

1 Introducere

Acest proiect urmărește dezvoltarea unui framework unificat care transformă explicațiile generate prin diverse metode de explicabilitate, precum SHAP [1] sau LIME [2], în explicații în limbaj natural. În acest sens, ne propunem a folosi modele lingvistice mari preantrenate și principii de prompt engineering, astfel încât rezultatele să fie utile publicului țintă, ușor de înțeles și relevante. Abordarea urmărește să reducă erorile și ambiguitățile prin șabloane structurale, instrucțiuni de fidelitate și tehnici de prompt engineering, menținând trasabilitatea dintre artefactele XAI și textul generat. Urmărim explorarea diverselor tehnici precum Few-Shot Prompting [3], Chain-of-Thought [4] și Self-Consistency [5] pentru a implementa un mecanism de trasabilitate a răspunsului, prezentare și validare a explicațiilor, fără a necesita reantrenarea modelelor.

2 Context

Metodele XAI uzuale includ abordări agnostice de model, precum SHAP [1], LIME [2], tehnici bazate pe perturbare [6], și metode specifice arhitecturii, adică cele pe gradient [7]. Aceste metode produc contribuții locale (pe o predicție) și contribuții globale (pe întregul model), precum și hărți de salienă, greu de înțeles fără contextualizare. În paralel, LLM-urile pot transforma aceste semnale în explicații narative accesibile, însă cu riscuri de pierdere a fidelității sau introducerea de halucinații. Prin prompt engineering (Few-Shot [3], Chain-of-Thought [4], Self-Consistency [5], șabloane), putem controla stilul și conținutul, păstrând trasabilitatea între text și valorile XAI și reducând ambiguitățile. Lipsesc însă cadre practice care să combine sistematic XAI + LLM cu constrângeri de format, validări post-generare și metrice (claritate, acoperire top-k, consistență). Proiectul propune tocmai acest cadru unificat: convertirea artefactelor SHAP/LIME în explicații concise, aliniate la rezultate și verificabile, fără reantrenarea modelelor.

3 Fundamente: SHAP și LIME

SHAP (SHapely Additive exPlanation) [1] este o metodă independentă de model pentru interpretarea predicțiilor, care atribuie fiecărei caracteristici o valoare care indică cât de mult influențează acea caracteristică predicția finală a modelului. Bazată pe valorile Shapeley din

teoria jocurilor, metoda SHAP descompune predicția modelului în valoare de bază plus suma contribuțiilor fiecărei caracteristici.

O altă metodă independentă de model, frecvent utilizată pentru explicabilitate, este LIME (Local Interpretable Model-agnostic Explanations) [2], care explică o predicție indicând caracteristicile cu cel mai mare impact asupra ieșirii modelului. LIME aproximează local comportamentul unui model complex printr-un model simplu, generând variații ale intrării, observând cum se modifică predicția și antrenând un model mai mic, ai cărui parametri indică importanța fiecărei caracteristici.

Dincolo de SHAP și LIME, există numeroase metode de explicabilitate (de exemplu tehnici bazate pe perturbare [6] sau pe gradient [7]), fiecare având avantaje și limitări în funcție de tipul datelor și al modelului. În acest proiect ne propunem să începem cu SHAP și LIME, urmând să extindem pe parcurs explorarea altor metode.

4 Abordarea propusă

Transformăm artefactele XAI în explicații în limbaj natural cu ajutorul LLM-urilor, controlate prin prompt engineering [3, 4, 5]. Printre obiective se enumeră: claritate pentru utilizator, fidelitate față de rezultate, și trasabilitate.

Pipeline scurt:

- Intrare: valori SHAP/LIME + metadate
- NLG: șabloane few-shot, CoT și Self-Consistency $\rightarrow$ text concis cu semn și magnitudine
- Verificare: conservarea sumei pentru arbori și notarea abaterilor/stabilității

La pasul de verificare, urmărim ca explicațiile generate să respecte proprietățile matematice și logice ale artefactelor XAI, iar orice abateri sau instabilități relevante pentru interpretarea rezultatelor să fie identificate.

Pentru valorile SHAP, urmărim conservarea sumei pentru arbori, adică suma contribuțiilor tuturor feature-urilor plus valoarea de bază trebuie să fie egală cu predicția modelului pentru un exemplu dat. Acest principiu asigură că explicația este completă. Modelul verifică, pentru fiecare instanță, dacă suma valorilor SHAP corespunde cu predicția.

Pentru metoda LIME, folosim notarea abaterilor și a stabilității, care presupune că sistemul urmărește dacă există discrepanțe majore între contribuții, instabilități de la o rulare la alta. Astfel, se pot marca automat zonele de incertitudine, caracteristicile cu aport foarte mic sau instanțele unde interpretabilitatea este compromisă.

5 Arhitectura sistemului

Sistemul are patru straturi principale, cu trasabilitate cap-coadă (artefact XAI $\leftrightarrow$ enunț textual):

- Explainer: calculează artefacte XAI (SHAP/LIME) + metadate (semn, magnitudine, top-k).

- Normalizer & Mapper: aliniază scale/limbi, grupează trăsături corelate, mapează (feature → afirmație).
- NLG: șabloane few-shot + CoT + Self-Consistency generează text concis cu marcaje de fidelitate.
- Validator & evidență: verifică alinierea textului cu valorile XAI și păstrează o tabelă de evidențe (enunț ↔ metodă, trăsătură, contribuție).

6 Strategii de prompt engineering

Prompt engineering este esențial pentru a controla comportamentul LLM-urilor și a obține răspunsuri orientate către obiective, existând numeroase astfel de tehnici pentru care au demonstrat o creștere a acurateții, factualității și consistenței ieșirilor generate [8]. Printre tehnicile de prompt engineering pe care urmărim să le folosim pentru a stabiliza răspunsurile, a reduce ambiguitățile și a păstra alinierea față de explicațiile date de metodele de XAI se numără:

- Chain-of-Thought (CoT) [4], care presupune ghidarea LLM-ului spre a oferi explicații pas cu pas pentru raționamentul ales, și încurajează descompunerea în etape intermediare înainte de a genera un răspuns final. Prin această metodă se urmărește reducerea erorilor prin clarificarea modului în care a fost obținut răspunsul, îmbunătățirea acurateței, dar și a transparenței decizionale.
  Pentru implementare, vom separa două secțiuni: ”Analiză” (inaccesibilă utilizatorului, în care modelul parcurge pașii intermediari din raționament) și ”Răspuns final” (vizibil utilizatorului, care conține explicația și concluzia).
- Self-Consistency [5], o tehnică prin care modelul este instruit să producă mai multe răspunsuri independente, iar rezultatul ales este cel care pare a fi cel mai consistent dintre ele. Spre deosebire de Chain-of-Thought, care încurajează un singur lanț de gândire, Self-Consistency urmărește explorarea unor rute variate, cu scopul de a crește robustețea inferenței.
  Pentru implementare, urmărim producerea a mai multor lanțuri de gândire independente prin CoT și generarea unui rezumat coerent care să agrege răspunsul majoritar într-un răspuns final.
- Few-Shot Prompting [3], prin care LLM-ul primește exemple reprezentative integrate direct în prompt, cu scopul de a ”învăța” din context, fără a fi reantrenat, aplicând apoi tiparul la intrări similare pentru a genera un răspuns.
  Pentru implementare, vom include câteva exemple din același domeniu în prompt, prin care să expunem mapări de tipul ”caracteristică” - ”afirmație”, cu scopul de a ghida modelul spre formatul și raționamentul dorit.

7 Plan de evaluare

Evaluăm pe patru axe, cu verificări simple și măsuri cuantificabile:

- Corectitudinea conținutului: textul reflectă semnul și ordinea top-k trăsături (potrivire 80%); la arbori, verificăm $\sum \text{SHAP} \approx F(x) - \mathbb{E}[F]$.
- Robustețe: rulări multiple (seed-uri/setări diferite) și Jaccard@k ≥ 0.6 între listele de top-k; marcăm variații mari ale contribuțiilor. [9]
- Utilitate: timp de parcurgere scurt, claritate/incredere auto-raportate (scor mediu 4/5), capacitatea de a identifica corect „primele 3" trăsături și direcția lor.
- Runtime: timp median/p95 per instanță sub bugetul stabilit; raportăm separat SHAP pe arbori vs. KernelSHAP/LIME pe modele generale.

8 Seturi de date

Pentru validarea și testarea sistemului propus, am ales două seturi de date bine frecvent întâlnite în comunitatea de ML:

- Breast Cancer Wisconsin (Diagnostic) [1]
  Este un set de date ușor de utilizat, cu caracteristici numerice clare folosit pentru clasificare binară (tumori maligne vs. benigne). Am ales acest dataset pentru a testa componentele noastre de explicare deoarece oferă o problemă clară și bine definită, fiind de dimensiuni reduse.
- Adult Income [2]
  Acest set conține date demografice și socio-economice despre adulți și peste 10 atribute numerice și categorice. Problema țintă este o clasificare binară destinată predicției dacă venitul anual este peste 50.000$. Se utilizează frecvent pentru teste de explicabilitate datorită complexității variabilelor și relevanței sociale clar definite.

Vom folosi aceste seturi de date pentru a antrena diverse modele (ex. Random Forest, XGBoost), vom aplica apoi SHAP și LIME pentru explicarea predicțiilor pentru instanțe relevante, urmând ca apoi să evaluăm rezultatele obținute.

Bibliografie

[1] Scott Lundberg and Su-In Lee. A unified approach to interpreting model predictions, 2017.

[2] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. "why should i trust you?": Explaining the predictions of any classifier, 2016.

[3] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner,

[1] https://www.kaggle.com/datasets/uciml/breast-cancer-wisconsin-data/data
[2] https://www.kaggle.com/datasets/wenruliu/adult-income-dataset

Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[4] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

[5] Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. Self-consistency improves chain of thought reasoning in language models, 2023.

[6] Maksims Ivanovs, Roberts Kadikis, and Kaspars Ozols. Perturbation-based methods for explaining deep neural networks: A survey. *Pattern Recognition Letters*, 150:228–234, 2021.

[7] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps, 2014.

[8] Shubham Vatsal and Harsh Dubey. A survey of prompt engineering methods in large language models for different nlp tasks, 2024.

[9] Christopher Burger, Charles Walter, Thai Le, and Lingwei Chen. Towards robust and accurate stability estimation of local surrogate models in text-based explainable ai, 2025.