

Natural Language Explanations for Explainable AI Systems using Large Language Models with Self-correcting Architectures

Matthew Renze
Johns Hopkins University
mrenz1@jhu.edu

Abstract

In this paper, we explore the generation of Natural Language Explanations (NLEs) for eXplainable AI (XAI) systems using Large Language Models (LLMs) with self-correcting architectures.

First, a dataset was created from a modified version of the COMPAS database and SHapley Additive exPlanations (SHAP). Next, a rule-based NLE generator was hand-coded to assess optimal NLE performance. Then, a Generative Pretrained Transformer (GPT) was used with few-shot learning to explain each case record using their corresponding SHAP values. A second GPT task verified each NLE and identified factual errors. A third GPT task corrected NLEs based on any factual errors identified. Finally, a hand-coded rule-based evaluator was used to assess the final performance.

GPT-4 with self-correction outperformed all other GPT models in terms of factual accuracy. However, it underperformed relative to GPT-3.5 in terms of runtime performance and cost efficiency. This research demonstrates the potential of using GPTs to create more trustworthy AI, highlighting areas of further investigation and improvement. All code, data, and supplemental materials are available on GitHub¹.

Keywords

Explainable AI (XAI), Natural Language Explanation (NLE), Generative Pretrained Transformer (GPT)

1. Introduction

The purpose of this research is to create Natural Language Explanations (NLE) for eXplainable AI (XAI) systems using Large Language Models (LLMs), like GPT-4, using self-correcting architectures.

The system is composed of three GPT models: an explainer, a verifier, and a corrector. The explainer generates letters of explanation for end users. The verifier critically evaluates each explanation to find factual errors. The corrector uses the verifications to correct any factual errors in the explanations.

¹ <https://github.com/matthewrenze/jhu-ai-ethics-final-project>

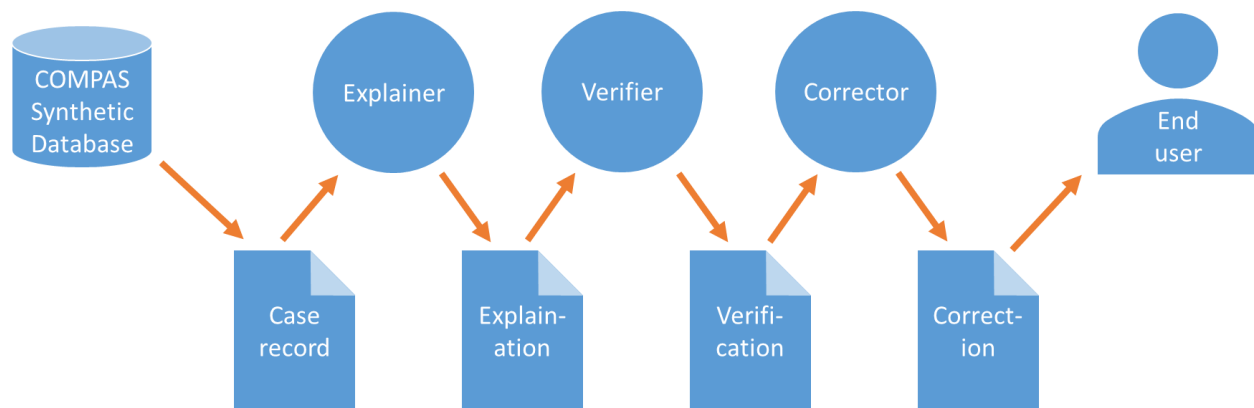


Figure 1 - System architecture, which includes an explainer, a verifier, and a corrector

1.1 Background

XAI is an active area of research. Data scientists and AI practitioners use XAI tools and techniques to better understand the decisions and predictions made by opaque Machine Learning (ML) models. This allows us to have more confidence and trust in the behavior of these ML models [1]–[6].

Unfortunately, most XAI tools and techniques require expertise to understand and use them effectively. This means that end users must rely on XAI experts to provide them with NLEs based on raw XAI outputs. This reduces end-user trust in ML systems and creates additional work for XAI experts [1], [2].

One solution to this problem is to use rule-based NLE generators. These systems use a set of hand-coded rules, typically encoded in a programming language, to generate explanations in natural language using XAI outputs [5], [7], [8]. Unfortunately, creating rule-based systems via explicit programming is difficult.

Creating rule-based NLE systems requires a static template with tags to inject dynamic content into the template. The dynamic content is then generated using a series of hand-coded grammatical rules. These rules handle pluralization, conjunctions, empty sentences, small integers written as words, etc.

In addition, a rule-based system requires strict logical operations, which makes the text they produce feel overly formal and unnatural. All of these issues make rule-based NLE systems less appealing to the developers creating them and to the end users reading their explanations.

The Generative Pre-trained Transformer (GPT) is an LLM created by OpenAI. It uses deep learning to generate human-like text. A human user provides the model with a text prompt, and the model produces text to complete the prompt as its output. Several versions of GPT exist, including GPT-3.5 and GPT-4. Each version has specific capabilities and limitations [9], [10].

GPTs have been used to automate a variety of tasks. These tasks include text summarization, language translation, style transfer, code generation, and debugging [9]. However, little research exists on the use of GPTs to generate NLEs for XAI.

Using GPTs to generate NLEs for XAI would allow non-experts to generate NLEs without programming skills. In addition, it would provide more natural-sounding explanations for end users. Other benefits might include interactive Q&A, multiple-language support, and adaptive NLEs for each end user.

1.2 Objective

My hypothesis is that we can automatically generate self-correcting NLEs using raw tabular data, corresponding SHAP values for each feature’s importance, and a GPT. Moreover, by adding a self-correcting architecture to the system, we can increase the factual accuracy of the generated NLEs.

To test this hypothesis, first, I created a rule-based NLE generator and evaluator to provide me with a ceiling for optimal performance. Next, I created an explainer, verifier, and corrector using GPT-3.5 and GPT-4 with few-shot learning. Then, I evaluated the performance of both the rule-based and GPT-based explanations. Finally, I analyzed the performance of the GPT models and compared their results.

1.3 Previous Literature

There are various approaches to XAI currently in use today. These approaches can be categorized into models that are interpretable by design, model-agnostic methods, model-specific methods, example-based methods, and exploratory methods via neural representations [3], [4], [6].

Some models are inherently interpretable by design. These include decision trees, logistic regression, and linear models. With these models, we can derive explanations for their decisions by inspecting the model itself or using an interpretable model as a surrogate for an opaque model [4], [6].

Model-agnostic methods include SHapely Additive exPlanations (SHAP) [11], Local Interpretable Model-agnostic Explanations (LIME) [12], and perturbation analysis. Both SHAP and LIME are commonly used tools in XAI. Perturbation involves changing model inputs to see how they influence outputs [3], [4], [13].

Model-specific methods include gradient saliency, integrated gradients, and attention. All three methods are used specifically for deep neural networks. However, they differ in terms of their applicability to computer vision (CV) or Natural Language Processing (NLP) [4], [6].

Example-based methods include adversarial examples, influence function, and counterfactual explanations. Adversarial examples test vulnerabilities, influence function determines the impact of individual data points, and counterfactual explanations consider “what if” scenarios [4], [6].

Exploratory methods via neural representations include Testing with Concept Activation Vectors (TCAV), activation maximization, and feature visualization. These methods provide insight into the internal representations learned by ML models [3], [4].

In terms of NLE for XAI, there are attempts at using rule-based template systems to generate NLEs. For example, Template System for Natural Language Explanations (TS4NLE) [5], the *describe* function in the iBreakDown package for R [7], and IBM’s AI Explainability 360 [8].

In addition, there have been various attempts at using LLMs for NLE in CV models – for example, NLX-GPT [14], MIMIC-NLE [15], and e-ViL [16]. There have also been attempts that use LLMs for NLE in NLP models – for example, NILE [17], RexC [18], and LIREx [19].

However, after an extensive search, there do not appear to be any existing research papers on generating NLEs for tabular data classification or regression models using LLMs. So, this appears to be a novel sub-field of research.

2. Methods

In this section, we will discuss the methods used for data collection, preprocessing, model training, and evaluation in this research project.

2.1 Data Collection

The dataset for this research project was a modified version of the Correctional Offender Management Profiling for Alternative Sanctions (COMPAS) database. COMPAS is a software tool used by the criminal justice system to predict the probability of a defendant committing a future crime [20].

The COMPAS database is a collection of records from 11,757 criminal defendants in Broward County, Florida, assessed using the COMPAS algorithm between 2013 and 2014. The database was obtained through a public-records request by Propublica to perform a bias analysis on the algorithm [21].

The COMPAS Synthetic database is a modified version of the COMPAS database. It has been augmented with synthetic data to simulate additional real-world attributes of defendants. It was created by Dr. Nathan Bos, Senior Research Associate at Johns Hopkins University, and is used with permission [22].

The COMPAS synthetic dataset consists of 6,173 records from the original COMPAS database. It contains 11 input variables, including the number of prior offenses, charge degree, juvenile felony count, etc. The target variable (i.e., decile score) represents the probability of recidivism based on a scale of 1 to 10.

Four of the eleven input features were synthetically generated to correlate with the target decile score variable. However, two synthetic features (i.e., family criminality and criminal attitude) were randomly generated and, thus, do not correlate with the target variable.

2.2 Data Preprocessing

The data contained in the COMPAS synthetic database was filtered to eliminate any unnecessary columns. All features were transformed and scaled to a range of 0 to 1. This preprocessing was necessary to prepare the data for a random forest (RF) regressor algorithm.

An RF regressor model was trained using the entire dataset. SHAP values for each record were created using the trained RF model. The resulting SHAP values were then scaled from 0 to 1 across all columns to reflect each feature's relative contribution to the overall decile score.

A text label was created for each scaled feature score and scaled SHAP value. These labels encoded the level of each value as low, medium, or high. This encoding was performed to allow the GPT models to work with both numeric and categorical (i.e., textual) representations of the case record data as input.

A random sample of 100 case records was selected from the database for use in the experiment. Each case record was converted from a row of tabular data into key-value pairs that were then stored in a plain-text document. An example of a plain-text case record can be found below.

```

# Case Info
id: 142
name: richard veach
decile_score: 8
decile_score_level: high

# Features
priors_count: 9
c_charge_degree: F
juv_fel_count: 0
family_criminality: 5
criminal_attitude: 8
criminal_associates: 7
financial_problems: 3
substance_abuse: 8
noncompliance: 6
social_environment: 5
vocational: 4

# Feature Levels
priors_count_level: high
c_charge_degree_level: high
juv_fel_count_level: low
family_criminality_level: medium
criminal_attitude_level: high
criminal_associates_level: high
financial_problems_level: low
substance_abuse_level: high
noncompliance_level: high
social_environment_level: medium
vocational_level: medium

# Feature Importance Levels
noncompliance_importance_level: high
substance_abuse_importance_level: high
criminal_associates_importance_level: medium
priors_count_importance_level: medium
financial_problems_importance_level: low
vocational_importance_level: low
social_environment_importance_level: low
c_charge_degree_importance_level: low
juv_fel_count_importance_level: low
family_criminality_importance_level: low
criminal_attitude_importance_level: low

```

2.3 Models and Training

A rule-based generator and four GPT models were created and tested using the dataset described above.

The rule-based generator was hand-coded in Python. It uses a plain-text template to contain the static portions of the letter of explanation. Placeholder tags inside the template were used to inject the dynamic content. Rule-based logic was used to format all sentences grammatically.

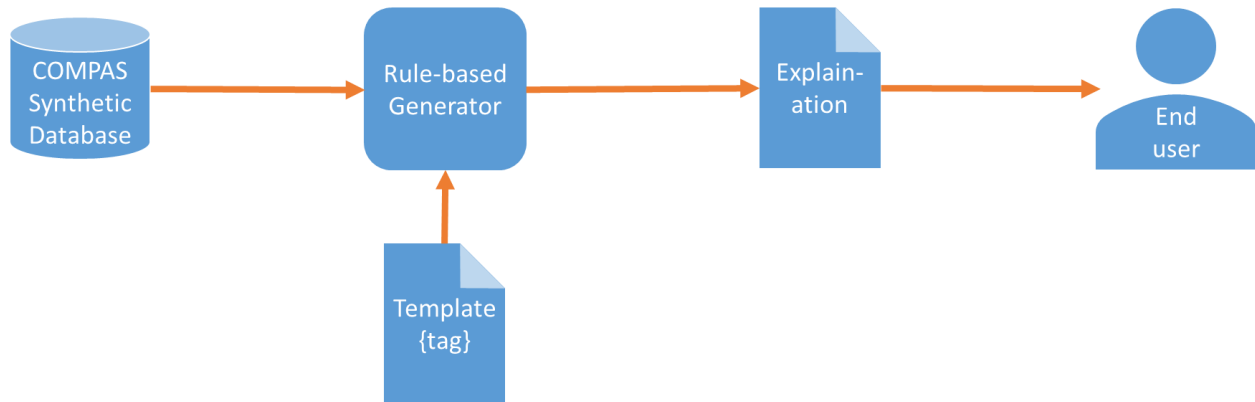


Figure 2 - Rule-based generator using a template to generate NLEs

An example of a letter of explanation created by the rule-based generator is below.

Richard,

Thank you for sharing your case record for our review.

Regarding your case record #142, you had a COMPAS risk score of 8 on a scale of 1-10. This means you were scored as a high risk for reoffending.

You received this score primarily because you had a high risk score for noncompliance (6) and a high risk score for substance abuse (8). These were the two most important factors in receiving a high overall risk score.

You also had a high risk score for criminal associates (7) and a high risk score for number of prior offenses (9). However, these factors only contributed medium importance to the COMPAS algorithm's scoring of your overall risk.

The remaining risk factors including financial problems (3), vocational risk factors (4), and social environment risk (5), etc. were of low importance to the COMPAS algorithm's score.

Demographic data including age, sex, race, etc., were not included in the algorithm's decision because they are protected by law.

Please let us know if you have any questions or concerns.

Sincerely,
Explainable AI

The GPT subsystem was composed of an explainer, a verifier, and a corrector. The explainer generates letters of explanation for end users. The verifier critically evaluates each explanation to find factual errors. The corrector uses the verifications to fix factual errors in the original explanation.

The GPT models were provided with a system prompt, few-shot examples, and a user prompt. The system prompt contained the instructions for the task. The few-shot examples contained sample input-output pairs. The user prompt contained the input data to generate the appropriate output for the task.

Two versions of GPT were evaluated using the plain-text case records described above (i.e., GPT-3.5 and GPT-4). Each version has various capabilities and limitations. GPT-3.5 allows a maximum of 4,096 tokens, while GPT-4 allows up to 8,192 tokens [10].

GPT-4's larger context window allows roughly double the number of examples for few-shot learning compared to GPT-3.5. As a result, the GPT-3.5 correction task was limited to using only two of the three few-shot learning examples. However, all three examples were used for the other two tasks.

2.4 Evaluation Methods

The performance of the models was determined using a hand-coded rule-based evaluation script. This script parsed the letters of explanation and extracted all facts contained in the explanations. It then compared each generated fact to the ground truth contained in the COMPAS Synthetic database.

Each fact contained in the letter of explanation was given a correctness score of +1, 0, or -1.

- A score of +1 indicates that a required fact was present in the explanation and was correct.
- A score of 0 indicates that an optional fact was not present in the explanation.
- A score of -1 indicates that a required fact was present in the explanation but was incorrect.

A quantitative analysis was performed to compare the three models. A factual accuracy score was used to assess each model's performance. Accuracy was computed as the number of correctly generated facts (+1) and optional facts (0) divided by the total number of possible facts {+1, 0, -1} for each case record.

The runtime performance of each model was analyzed. The duration of each inference (in seconds) was measured from the start of each inference to the model's response. A cost analysis was also performed. The cost per explanation (in US dollars) was calculated using the number of tokens and price per token.

A qualitative analysis was performed on the generated NLEs. A randomly selected subset of 10 explanations from each model was manually inspected for readability, understandability, sentence structure, grammatical errors, and misspellings.

3. Results

In this section, we will discuss the performance of the four GPT models in terms of accuracy, runtime, and cost performance.

3.1 Performance Analysis

GPT-4 with self-correction outperformed the other three GPT models with a factual accuracy of 0.996. GPT-4 without corrections performed next best with an accuracy of 0.993. GPT-3.5 with and without corrections performed worst, with an accuracy of 0.949 and 0.938, respectively. The accuracy of the rule-based model (i.e., accuracy = 1.0) was included to allow for comparison with optimal performance.

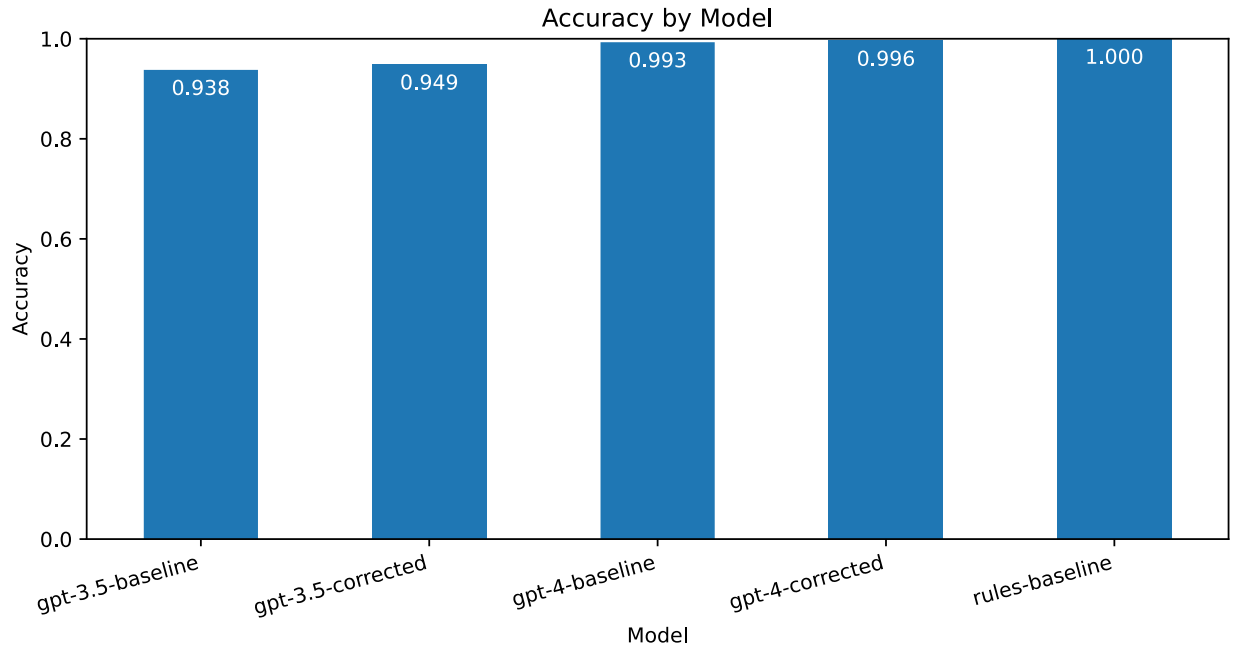


Figure 3 - Factual accuracy by model

3.2 Detailed Analysis

All four models scored perfectly in their ability to correctly generate basic case information and feature scores. The models scored worse on the feature score levels (e.g., low, medium, or high) used to describe each feature's score. The models performed worst in their ability to place features in the correct paragraph according to their feature importance.

The figures below represent the worst-performing model (i.e., GPT-3.5 without corrections) and the best-performing model (i.e., GPT-4 with corrections). Charts containing detailed errors for the other two models can be found in the supplemental materials.

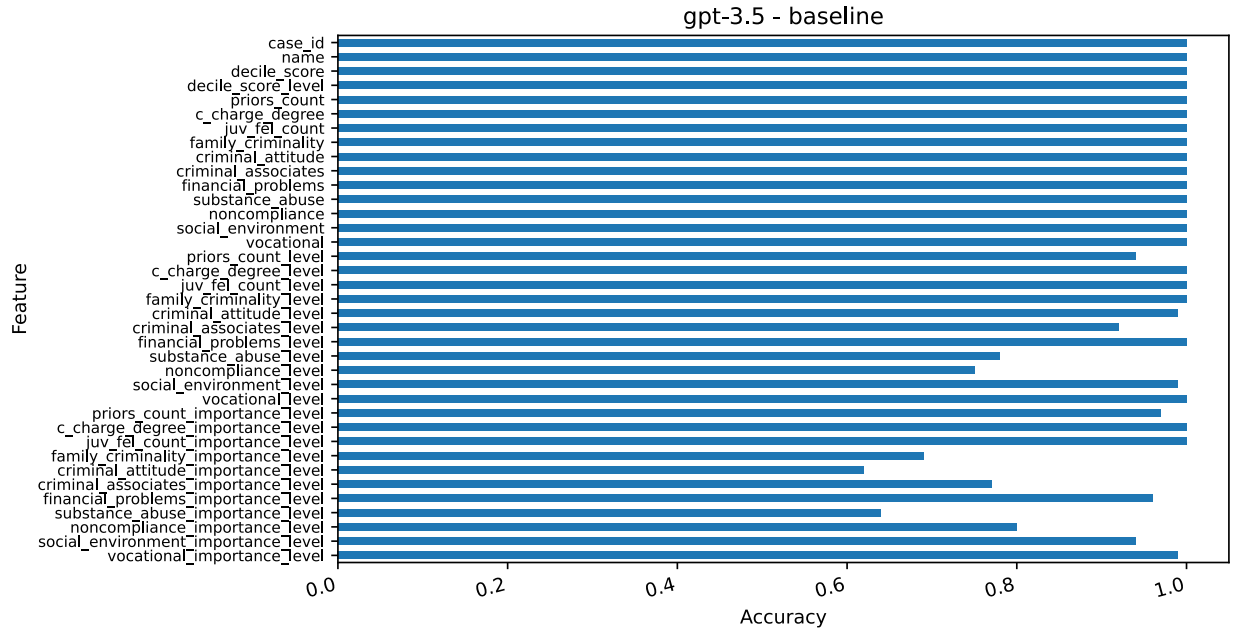


Figure 4 – Factual accuracy by feature of the worst-performing model - GPT-3.5 without corrections

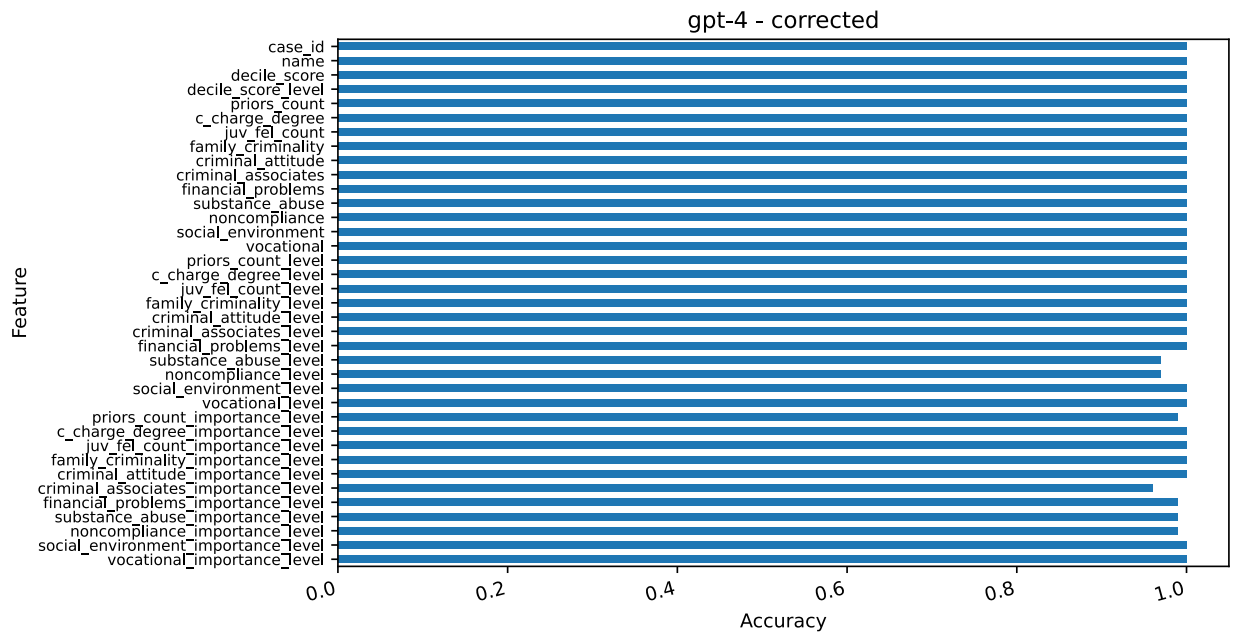


Figure 5 – Factual accuracy by feature of the best-performing model - GPT-4 with corrections

3.3 Runtime Analysis

GPT-3.5 was the fastest-performing model, with an average duration of 15.91 seconds per explanation, 18.66 seconds per verification, and 21.02 seconds per correction. GPT-4 tasks were roughly twice as slow as the equivalent GPT-3.5 tasks at 35.78 seconds, 42.51 seconds, and 37.79 seconds, respectively.

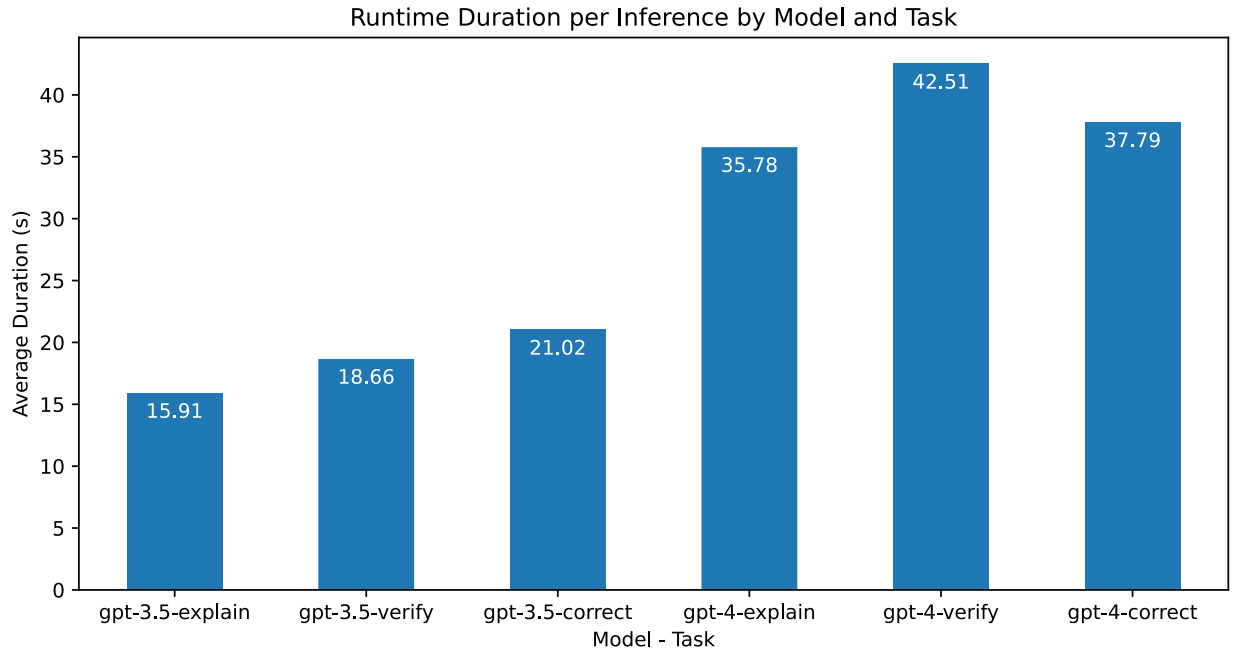


Figure 6 - Runtime performance by model and task

The runtime performance per token roughly mirrored the runtime performance per inference. Once again, the GPT-3.5 model performed best, with an average runtime duration of 5.24 to 6.36 milliseconds per token. GPT-4 performed worst, with an average runtime of 9.35 to 14.61 ms per token.

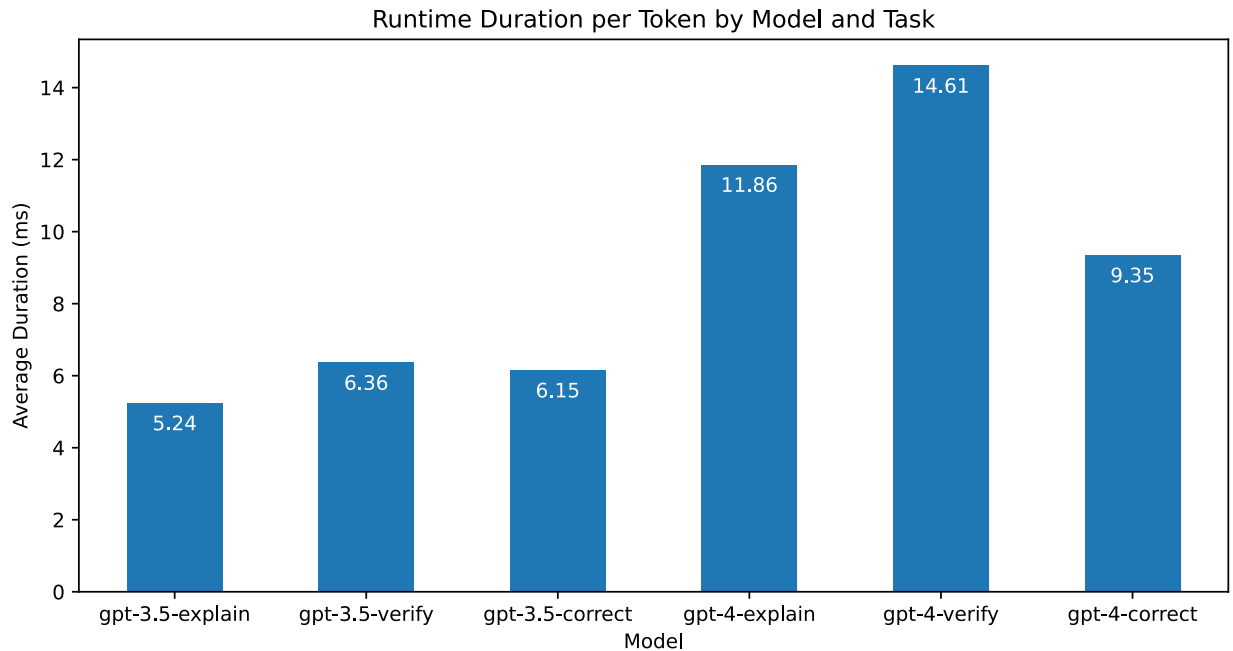
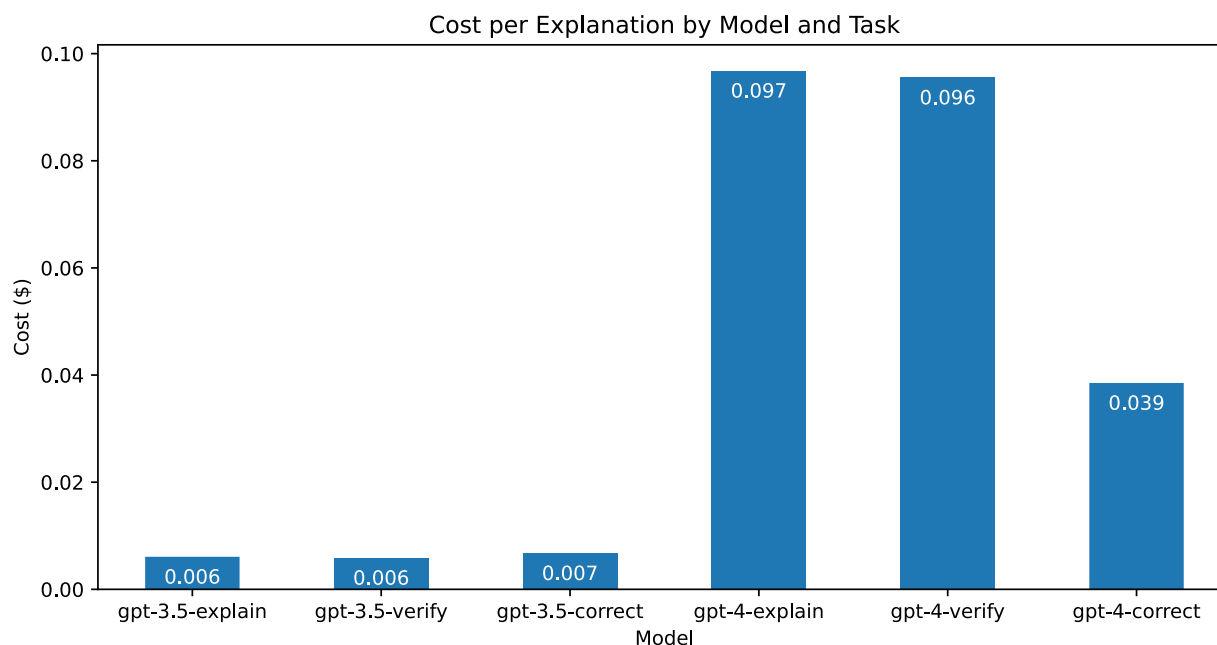


Figure 7 - Runtime performance per token by model and task

The discrepancy between the per-inference and per-token runtime performance was due to the fact that the GPT-3.5 correction task only used two of the three few-shot learning examples.

3.4 Cost Analysis

GPT-3.5 was significantly less expensive than GPT-4, with an average cost of \$0.006 per inference. GPT-4 had an average cost of \$0.097 for two of the three inference tasks. However, the cost of the correction task was significantly lower because far fewer GPT-4 generated explanations required any corrections.



3.5 – Qualitative Analysis

A review of randomly selected explanations identified no errors in spelling, grammar, or sentence structure. The explanations produced were often perfect mirrors of the letters generated using a rule-based approach. So, they are roughly equivalent in terms of readability and understandability.

For comparison, a sample NLE produced by GPT-4 with self-corrections is provided below. This corresponds to the rule-based NLE provided in section 2.3. Only two minor differences (highlighted below) can be found: the addition of “the” and the swapped order of two low-importance features.

Richard,

Thank you for sharing your case record for our review.

Regarding your case record #142, you had a COMPAS risk score of 8 on a scale of 1-10. This means you were scored as a high risk for reoffending.

You received this score primarily because you had a high risk score for noncompliance (6) and a high risk score for substance abuse (8). These were the two most important factors in receiving a high overall risk score.

You also had a high risk score for criminal associates (7) and a high risk score for the number of prior offenses (9). However, these factors only contributed medium importance to the COMPAS algorithm's scoring of your overall risk.

The remaining risk factors including financial problems (3), social environment risk (5), and vocational risk factors (4), etc. were of low importance to the COMPAS algorithm's score.

Demographic data including age, sex, race, etc., were not included in the algorithm's decision because they are protected by law.

Please let us know if you have any questions or concerns.

Sincerely,

Explainable AI

4. Discussion

In this section, we will discuss the limitations of this research project, the implications of this work, and potential opportunities for future research.

4.1 Limitations

This research project was limited by the use of only a single data source to generate NLEs. In addition, the data source was not a benchmark dataset used for XAI tasks. Moreover, the data were synthetically generated, so a naturally occurring database would have provided better real-world conditions.

The max token limit for both GPT-3.5 and GPT-4 was a significant limitation. It constrained the instructions, few-shot examples, and input data for each prompt, which likely impacted performance. In addition, only two of the three few-shot examples could be used for the GPT-3.5 correction task.

As a result, few-shot examples had to be hand-selected and hand-crafting to cover a wide range of scenarios that occur with NLEs. Paragraphs containing zero to four feature scores were common. So, hand-picked examples had to be diverse enough to allow the model to generalize to these cases.

There were also several failures in early iterations of the system. Originally, the explanation task used raw feature scores and SHAP values to generate letters of explanation. However, this approach was abandoned because the models performed better on leveled values (e.g., low, medium, and high).

In addition, early attempts to use a single verifier-and-corrector model also failed. Separating the verifier and corrector into separate models improved the overall correction score. However, it would have been beneficial in terms of runtime and cost to only perform a single inference for each correction.

4.2 Implications

This research project demonstrates the feasibility of automatically generating letters of explanation in natural language using self-correcting GPT models. This work lays a foundation for future research on XAI, NLE, and self-correcting LLMs. As a result, further research in this area of study is warranted.

Automating this task has the potential to provide end users with NLEs for opaque ML models. This could increase end-user trust in ML systems. In addition, automating this task frees data scientists and ML engineers from the task of translating technical XAI data (e.g., SHAP, LIME, etc.) into NLEs for end users.

This research also builds upon existing research in NLE for XAI. It moves beyond template-based and rule-based NLE systems to use LLMs like GPT with self-correction architectures to perform the same task more naturally and flexibly. It also opens the door for interactive Q&A sessions with GPTs for NLEs.

4.3 Future Research

To improve upon this research, I propose the following follow-up experiments:

First, a new experiment should be performed with a commonly used XAI benchmark dataset. This would allow the resulting rule-based and GPT-generated NLEs to be used as a new benchmark dataset for future experiments using LLMs to create NLEs for XAI.

Second, the experiment should be repeated with raw feature scores and raw SHAP values. Prompt engineering or data representation improvements could lead to better performance than the previously abandoned attempts in this experiment.

Third, the experiment should be repeated using the rank order of feature importance to present features in the correct order. This is the only way to guarantee a required feature isn't omitted in an NLE. In addition, it would ensure that all features are listed by their order of importance within each paragraph.

Fourth, the verification task should be modified to provide a score for perceived factual accuracy (i.e., "groundedness"). This score could be used to rate NLEs and set a threshold of acceptability for potential errors. This addition would allow the system to reject NLEs that fall below a specified confidence level.

Fifth, an experiment involving best-n generated explanations and dynamic adjustments on GPT model temperature (i.e., "creativity") should be performed. This enhancement would allow the system to produce multiple NLEs and pick the NLE with the highest perceived factual accuracy.

Sixth, the experiment should be repeated using data formatted in either Comma Separated Values (CSV) or markdown table notation – rather than key-value pairs. There is significant duplication of text when using key-value pairs. However, condensed tabular formats would eliminate most of this duplication.

Seventh, this experiment should be repeated with GPT-4 32k once it has been released. Having access to 32,000 tokens would provide a context window four times larger than that of GPT-4 8k. Additional tokens would allow more text in prompts for instructions, few-shot examples, and input data.

Finally, a more in-depth qualitative analysis should be performed. Letters of explanation should be evaluated for readability, understandability, narrative, and trustworthiness. This experiment could be performed using a survey with A/B testing to compare a set of generated NLEs for perceived quality.

5. Conclusion

This research paper has demonstrated the potential of automatically generating NLEs for XAI using LLMs. GPT-4 outperformed GPT-3.5 in terms of accuracy. However, it underperformed in terms of runtime performance and cost efficiency.

These findings indicate that GPTs have the potential to provide end users with NLEs to explain the decisions of opaque ML models. Automating this task could create more trustworthy AI and free up data scientists from the task of translating technical XAI explanations into NLEs for end users.

However, there are several limitations to this study. For example, the explainer model had to be helped by converting feature scores and SHAP values into three levels (i.e., low, medium, and high). In addition, the verifier and corrector could not be combined into a single model without a loss of accuracy.

Future research should explore the use of benchmark XAI datasets. Additional experiments should attempt to improve accuracy through better prompt engineering, model augmentations, and GPT-4 32k. Finally, the subjective quality of explanations should be measured through a qualitative assessment.

Overall, the use of GPT models to automate NLEs for XAI shows great promise. This research paves the way to build more explainable, interpretable, and trustworthy AI systems.

6. References

- [1] A. F. Markus, J. A. Kors, and P. R. Rijnbeek, "The role of explainability in creating trustworthy artificial intelligence for health care: A comprehensive survey of the terminology, design choices, and evaluation strategies," *J Biomed Inform*, vol. 113, p. 103655, Jan. 2021, doi: 10.1016/J.JBI.2020.103655.
- [2] W. Saeed and C. Omlin, "Explainable AI (XAI): A systematic meta-survey of current challenges and future opportunities," *Knowl Based Syst*, vol. 263, p. 110273, Mar. 2023, doi: 10.1016/j.knosys.2023.110273.
- [3] S. R. Islam, W. Eberle, S. K. Ghafoor, and M. Ahmed, "Explainable Artificial Intelligence Approaches: A Survey," *ArXiv*, vol. abs/2101.09429, 2021.
- [4] M. Nagahisarchoghaei *et al.*, "An Empirical Survey on Explainable AI Technologies: Recent Trends, Use-Cases, and Categories from Technical and Application Perspectives," *Electronics (Basel)*, vol. 12, no. 5, p. 1092, Feb. 2023, doi: 10.3390/electronics12051092.
- [5] I. Donadello and M. Dragoni, "Bridging Signals to Natural Language Explanations with Explanation Graphs," in *Proceedings of the 2nd Italian Workshop on Explainable Artificial Intelligence*, Online Event: CEUR, 2021, pp. 1–13.
- [6] S. Nazir, D. M. Dickson, and M. U. Akram, "Survey of explainable artificial intelligence techniques for biomedical imaging with deep neural networks," *Comput Biol Med*, vol. 156, p. 106668, Apr. 2023, doi: 10.1016/J.COMPBIOMED.2023.106668.
- [7] P. Biecek, "Explanations in natural language," *The Comprehensive R Archive Network (CRAN)*, 2022. https://cran.r-project.org/web/packages/iBreakDown/vignettes/vignette_iBreakDown_description.html (accessed Apr. 26, 2023).
- [8] IBM, "AI Explainability 360," *IBM Research Trusted AI*. <https://aix360.mybluemix.net/> (accessed Apr. 26, 2023).
- [9] Open AI, "GPT-4," 2023. <https://openai.com/product/gpt-4> (accessed Apr. 05, 2023).

- [10] Open AI, “Models.” <https://platform.openai.com/docs/models/overview> (accessed Apr. 05, 2023).
- [11] S. Lundberg and S.-I. Lee, “A Unified Approach to Interpreting Model Predictions,” in *31st Conference on Neural Information Processing Systems (NIPS 2017)*, Long Beach, CA: Curran Associates, Inc., 2017.
- [12] M. T. Ribeiro, S. Singh, and C. Guestrin, “‘Why Should I Trust You?’: Explaining the Predictions of Any Classifier,” in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, New York, NY, USA: ACM, Aug. 2016, pp. 1135–1144. doi: 10.1145/2939672.2939778.
- [13] M. Ivanovs, R. Kadikis, and K. Ozols, “Perturbation-based methods for explaining deep neural networks: A survey,” *Pattern Recognit Lett*, vol. 150, pp. 228–234, Oct. 2021, doi: 10.1016/j.patrec.2021.06.030.
- [14] F. Sammani, T. Mukherjee, and N. Deligiannis, “NLX-GPT: A Model for Natural Language Explanations in Vision and Vision-Language Tasks,” Mar. 2022.
- [15] M. Kayser, C. Emde, O.-M. Camburu, G. Parsons, B. Papiez, and T. Lukasiewicz, “Explaining Chest X-Ray Pathologies in Natural Language,” 2022, pp. 701–713. doi: 10.1007/978-3-031-16443-9_67.
- [16] M. Kayser *et al.*, “e-ViL: A Dataset and Benchmark for Natural Language Explanations in Vision-Language Tasks,” May 2021.
- [17] S. Kumar and P. Talukdar, “NILE : Natural Language Inference with Faithful Natural Language Explanations,” in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Stroudsburg, PA, USA: Association for Computational Linguistics, 2020, pp. 8730–8742. doi: 10.18653/v1/2020.acl-main.771.
- [18] B. P. Majumder, O.-M. Camburu, T. Lukasiewicz, and J. McAuley, “Knowledge-Grounded Self-Rationalization via Extractive and Natural Language Explanations,” Jun. 2021.
- [19] X. Zhao and V. G. V. Vydiswaran, “LIREx: Augmenting Language Inference with Relevant Explanation,” in *The Thirty-Fifth AAAI Conference on Artificial Intelligence (AAAI-21)*, 2021, pp. 14532–14539.
- [20] Northpointe, “Practitioner’s Guide to COMPAS Core,” *Northpointe*. Mar. 19, 2015. Accessed: Apr. 26, 2023. [Online]. Available: <https://assets.documentcloud.org/documents/2840784/Practitioner-s-Guide-to-COMPAS-Core.pdf>
- [21] J. Larson, S. Mattu, L. Kirchner, and J. Angwin, “How We Analyzed the COMPAS Recidivism Algorithm,” *Propublica*, May 23, 2016. <https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm> (accessed Apr. 26, 2023).
- [22] Johns Hopkins University, “Nathan Bos | Hopkins EP Online.” <https://ep.jhu.edu/faculty/nathan-bos/> (accessed Apr. 27, 2023).