RIXA - Explaining Artificial Intelligence in Natural Language

Maximilian Becker
Vision and Fusion Laboratory
Karlsruhe Institute of Technology
Karlsruhe, Germany

Vishwesh Vishwesh
Fraunhofer IOSB
Karlsruhe, Germany

Pascal Birnstill
Fraunhofer IOSB
Karlsruhe, Germany

Finn Schwall
Fraunhofer IOSB
Karlsruhe, Germany

Sihan Wu
Vision and Fusion Laboratory
Karlsruhe Institute of Technology
Karlsruhe, Germany

Jürgen Beyerer
Fraunhofer IOSB
Karlsruhe, Germany

***Abstract*—Natural language is the instinctive form of communication humans use among each other. Recently large language models have drastically improved and made natural language interfaces viable for all kinds of applications. We argue that the use of natural language is a great tool to make explainable artificial intelligence (XAI) accessible to end users. We present our concept and work in progress implementation of a new kind of XAI dashboard that uses a natural language chat. We specify 5 design goals for the dashboard and show the current state of our implementation. The natural language chat is the main form of interaction for our new dashboard. Through it the user should be able to control all important aspects of our dashboard. We also define success metrics we want to use to evaluate our work. Most importantly we want to conduct user studies because we deem them to be the best method of evaluation for end-user-centered applications.**

***Index Terms*—XAI, natural language explanations, interactive explanations, personalized explanations, XAI dashboard**

I. Introduction

In the age of data-driven decision-making, artificial intelligence (AI) has become an integral component of many industrial, commercial, and research sectors [1]–[3]. While the advancements of AI promise unprecedented efficiency, accuracy, and scalability, these algorithms often operate as 'black boxes', the inner workings of which remain a mystery even to the very engineers who create them. As technology gets more advanced, it is important to understand how AI makes its decisions and the reasoning behind these decisions [4], [5].

The European Union's Artificial Intelligence Act (AI Act) underscores the importance of transparency in AI applications, particularly in high-risk scenarios and specifically mentions that users of AI systems have to be able to interpret their output [6]. The proposed AI Act specifically states, "Users should be able to interpret the system output and use it appropriately" [7]. Such regulations, designed to ensure accountability and ethics in AI, mandate that AI solutions offer clarity and justification for their decisions. This could create an environment where understanding AI is not merely a luxury - it is a legal and ethical necessity [8].

Generally, humans learn better with multiple modalities [9]. For example, text and visualizations which are both potent mediums, with text offering depth and detail, while visual elements provide immediate, intuitive insights [10]. An optimal interface for AI explanations, therefore, should seamlessly integrate both [11]. However, the challenge arises when one notes that most of the XAI implementations lean too heavily on the technical side, often overwhelming end users with intricate metrics and confusing data visualizations [12]–[14].

Fortunately, in the last decade, XAI has arguably matured, providing robust tools and frameworks to interpret intricate AI models. Moreover, the advent of large language models (LLMs), such as the GPT variants, due to their transformer architecture, offer potentially unparalleled capability to generate human-like text, serving as a bridge between complex algorithms and human comprehension [15]–[18].

Our proposed XAI dashboard, RIXA (real-time interactive XAI analyser)[1] aims to place the user at its core. It allows the user to obtain all the technical details on demand. But it aims at shifting the focus away from complex technical metrics, eliminating the need for users to navigate through complicated technical graphs and plots or the need to understand difficult AI concepts. Instead, it offers an intuitive platform where users can generate the explanations they need, at the granularity they desire, leveraging the combined strength of text and visual outputs. By empowering users, we aim to demystify AI, fostering an environment of understanding and transparency.

Building on the philosophy of our proposed XAI dashboard RIXA, we outlined design goals in Section III that underscore its foundational principles. RIXA aims to be inherently end-user-centered. It should cater to individuals irrespective of their technical background. It should be extendable, adapting to the diverse needs and user preferences. The platform is designed to be conversational and interactive, fostering a dynamic engagement between the AI and its users. Rather

[1]Our implementation and documentation can be found here: https://github.com/finnschwall/RIXA

than making interpretations, RIXA's core function should be to inform, providing clear and unbiased information. Above all, RIXA demonstrates transparency, aiming at forging a pathway for genuine confidence in AI systems.

The sections that follow are structured as such: Section II delves into the foundational concepts of XAI and the intricacies of natural language processing, while Section III presents the envisioned goals for the development of our dashboard. Building on this foundation, Section IV proposes our dashboard concept, explaining its features related to the natural language chat, explanatory mechanisms, and associated technical aspects. After presenting our dashboard in detail, Section V discusses the ongoing development of RIXA and presents planned future work, in addition to stating the metrics we plan to use to evaluate the success of our dashboard. We conclude the paper in Section VI, where we summarize the key takeaways and discuss potential future directions.

II. Background

A. Explainable Artificial Intelligence

In the expansive world of AI, XAI has garnered significant interest and research [19]. At its core, XAI aims to make the often complex and opaque decision-making processes of AI models transparent, interpretable, and justifiable to human users. As the integration of AI systems into various sectors like finance and medicine increases, so does the requirement for these systems to be not just efficient, but also understandable [20].

A recurrent criticism of XAI is its overwhelmingly technical complexity [21]. Numerous techniques, algorithms, and models that fall under the XAI umbrella tend to be detailed and intricate, often riddled with deep mathematical formulations [22]. While these technicalities ensure precision and meticulousness, they may exclude a significant portion of the intended audience: the non-technical users or stakeholders. For many, the core value of XAI lies not in the intricate mathematics but in the ability to understand and effectively use AI-based systems in real-world contexts [23], [24].

Upon examining the vast literature on XAI [19], [25]–[31], one may observe an interesting trend: a multiplicity of methods that, at their essence, offer similar insights or outcomes. Whether it is the extraction of feature importance, surrogate modeling, or visualization techniques, users usually concentrate on the result and not the mode of generation of explanations [32].

B. XAI Dashboards

In general, a dashboard is a graphical user interface that provides users with easy access to important information and data visualizations. Dashboards are commonly used to present key performance indicators, metrics, and other data in a concise and understandable format. In the context of XAI, dashboards traditionally provide an interface for users to explore and interpret the outputs of machine learning models. These dashboards typically display visualizations aiming to help users understand why a model makes the decisions it does [33]. They can be the first step in promoting transparency and accountability in machine learning systems. They aim at allowing users to explore and understand the models' decision-making processes, which can help to identify biases, errors, or other issues in the model. Some dashboards related to our work are presented in the next section.

C. Related Work

The following popular interactive dashboards fulfill some of our design goals listed in Section III but are not conversational: *ModelStudio*[2] [34] is a wrapper for the *Dalex*[3] library which creates a serverless site from the *Dalex* methods. The presentation is fairly technical and geared towards *Dalex* users. *OmniXAI* [4] [35] is a collection of XAI visualizations in a dashboard. The methods can also be used without the dashboard. While some code is necessary to create the dashboard, it is minimal. However, it targets users creating models or users that are very familiar with machine learning. *Google Explainable AI*[5] is a dashboard with a wide range of methods. Generally, they are presented more understandably than the other XAI dashboards. Still, it mainly addresses model creators and not end users. Additionally, it is not open source.

Other than dashboards, there are language based approaches. *XAINES* [36] and *Mediators* [37] are potentially very similar in concept to our approach, but there is no public implementation as of now. *Glass-Box* [38], [39] is out of the concept stage. It generates counterfactuals in an interactive text-based approach. However, it is limited to counterfactuals and is not extendable or open source. Although the concept of *ConvXAI* [40] seems very similar to ours, it is not extendable and it is questionable if it is end user centered. While there is a GitHub repository[6], it is inactive and undocumented which means that it lacks the much needed transparency.

TalkToModel [41] has reached a deployable stage, is open source, includes tutorials and is extendable. It is however limited to text-only interactions. The repository[7] is not actively developed anymore. Additionally, the key element, the chatbot, seems to be very limited in its ability to converse with a user and understand the user's intent. Most importantly, given the representation of the explanation, text combined with numbers, it is debatable if it is truly end-user-friendly.

Lastly, the system *ExpliClas* [42] is working and able to generate multimodal explanations. It is open source and extendable but not geared towards end users and limited to WEKA[8].

More recently *ChatGPT Plugins*[9] were introduced to provide ChatGPT access to information sources and third party

[2]https://github.com/ModelOriented/modelStudio

[3]https://github.com/ModelOriented/DALEX

[4]https://github.com/salesforce/OmniXAI

[5]https://cloud.google.com/explainable-ai

[6]https://github.com/Naviden/ConvXAI

[7]https://github.com/dylan-slack/TalkToModel

[8]"Waikato Environment for Knowledge Analysis" which is a data mining toolbox developed in Java, mainly by researchers affiliated to the University of Waikato (New Zealand).

[9]https://platform.openai.com/docs/plugins/introduction

APIs. It is in an early stage of development, exclusive to the registered customers and not open source. This functionality could however be used to create a similar application to ours.

Ultimately, to our knowledge none of the current systems can fulfill all of our requirements listed in Section III, although *TalkToModel* and *ChatGPT Plugins* come the closest. Many concepts do not seem to leave an early development or prototype stage. There is also the problem with missing transparency, either through non-disclosed or non-documented code or user studies that are hard to comprehend.

D. Natural Language Processing

Natural language processing (NLP) represents an important component in the development of XAI systems. By enabling AI systems to understand, generate, and interact with natural language, it lays the foundation of an effective human-machine communication. A specific use case are natural language chat system. The best chat systems currently are built on LLMs, to which an overview is given in the following.

The rise of pretrained language models (LMs) has revolutionized the field of modern NLP [43], [44]. Recently, LMs use transformer-based architectures (GPT, Llama). Through intensive pretraining on massive amounts of text data, the LM gains an inherent understanding of linguistic nuances, context, and semantics. Fine-tuning these pretrained LMs on task-specific domains or target tasks can use the acquired knowledge to achieve improved performance on downstream tasks. This approach proves to be very effective and establishes itself as the *pretraining, fine-tuning* paradigm [45].

With the insight that scaling the size of model and dataset has a decisive impact on the LM's performance [44], [46], [47], large pretrained LMs emerge and constantly beat the state of the art performance on various NLP tasks [46], [48], [49]. The release of Llama models and the fine-tuned Llama variants (Alpaca, Vicuna), however, demonstrates that with efficient implementation, smaller LLMs can achieve comparable performance to larger models [50]. Moreover, memory usage can be further decreased e.g., through quantization techniques [51], and methods of parameter-efficient fine-tuning [52]. An example of this is Low Rank Adaptations (LoRAs). These are able to adapt a LLM's behavior to specific use cases while substantially reducing memory and computational requirements compared to a full fine-tuning.

Using these techniques, it is feasible to build a natural language chat system by adapting a pretrained LLM to the domain of XAI and our specific use case.

III. Design Goals

In this section we present the design goals that shape the development of our dashboard. Our design goals are listed below and explained in detail in the following sections:

1) End-User-Centered
2) Extendable
3) Conversational and Interactive
4) Inform, don't Interpret
5) Transparency and Data Protection

1) End-User-Centered

Our application's most important design goal is its focus on the end user. Users can be everything, ranging from domain experts with specialized knowledge to laypeople. Using the dashboard should not require previous knowledge in XAI. The deployed dashboard, with its simple layout, aims for intuitive use without requirement of technical expertise.

2) Extendable

Another key concept is our approach's extendability and modularity. A powerful plugin system will be integrated. It will support the inclusion of a vast variety of capabilities into the server, with focused support on XAI plugins. This modularity enables us to support a wide array of use cases and target audiences. The dashboard can fulfill completely different tasks, depending on the plugin configuration.

3) Conversational and Interactive

Through the use of LLMs we aim to provide a conversational and interactive experience. We envision a natural language chat being the dashboard's main interaction method through which users can control all aspects of their experience. Users can explicitly ask for explanations in natural language or visualizations. They can ask follow-up questions for more details on what an explanation shows or how it is generated. They can also request information about the underlying machine learning model or instructions on using the dashboard.

4) Inform, don't Interpret

The next design criterion is that our dashboard should aim to minimally influence the user. The dashboard should inform the user with information grounded in factual data. The dashboard is not supposed to draw any conclusions for the user and only wants to give them the information needed to reach an informed conclusion.

This design goal requires consideration of some technical and psychological problems. LLMs tend to hallucinate, meaning they can create factually false information. To maintain our end-user-centric goal, it is crucial to take measures to mitigate this. It is important to inform users when something is unclear or unknown, rather than fabricating a plausible answer. Therefore, factual data should support the LLM. Relatedly, the dashboard should alert users to technical problems or drawbacks. For instance, if an XAI technique requires a large training dataset for robust results and such data is unavailable, the user should be informed of this potential issue. In addition to these technical issues, psychological problems also need to be addressed. Users often seek confirmation of their biases, a factor the dashboard should consider.

5) Transparency and Data Protection

This summarizes several important points for our design. Firstly, we publish all our code and documentation as open source. Besides being open source, it should be usable without any closed-source dependencies. We also plan on providing a running online live demo. However, being online will not be

a necessity. The dashboard should also run locally without an internet connection. Lastly, we will prioritize privacy and data protection. Users of our dashboard will maintain sovereignty and control over their data and its processing.

Summary

Here, we presented five design goals for our work. Focusing on end users is crucial to our work. We chose Design Goals 1 and 3 because we believe that XAI needs to be more user-friendly and intuitive to achieve a broader adaptation. Design Goal 2 aims to ensure our scope is not too narrow and supports a wide variety of use cases. Design Goal 4, born from common criticism of LLMs, ensures the reliability of information provided by our dashboard. Lastly, Design Goal 5 represents the authors' beliefs that applications aiming to increase transparency should themselves be transparent and safeguard user data.

IV. RIXA - Real-time Interactive XAI Analyser

Here, we present RIXA, our design and implementation based on the design goals laid out in the previous section.

To facilitate this, we put a multimodal approach, i.e., a chat for text-based user interaction and a dashboard for visualizations in the center of the application. Through the chat, users can ask all kinds of questions and get not only natural language explanations but also dashboard elements.

To demonstrate the dashboard, this section starts with a broad overview of the dashboard with screenshots taken from a development version[10]. Following this comes Section IV-B that describes features and designs in more detail.

A. RIXA User Experience

In the following we show a first look at RIXA and demonstrate an exemplary user interaction. We chose a medical scenario with a heart disease prediction dataset[11].

RIXA's appearance and UI can be fully customized. But the default version consists of a chat on the right and room for interactive elements on the left (see Figure 1 and 2). We aim for a natural and intuitive chat interaction, more details on the chat can be found in Section IV-B1. The user can ask questions about the specific scenario. These are answered in text and, if applicable, with visualizations that are shown in the dashboard part on the left. XAI or other plugins are automatically selected if they provide relevant information. Requests are translated into function calls, more details on this can be found in Section IV-B2. The results can be visualized, integrated into the chat response or both. In the following example a SHAP[12] and a data exploration plugin are used.

Figure 1 shows the beginning of an interaction. After the user is greeted they ask for risk factors for heart disease. RIXA recognizes that this request can be answered by calculating a

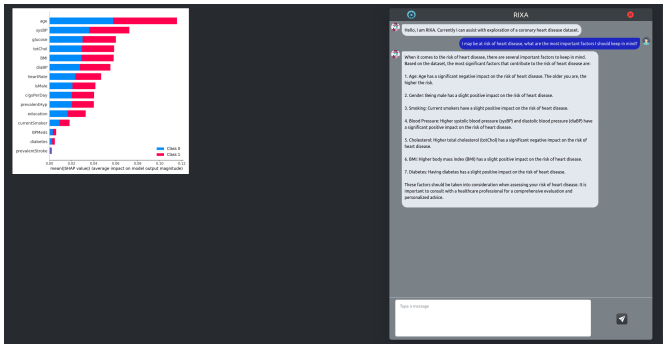

Fig. 1. Beginning of an interaction with RIXA

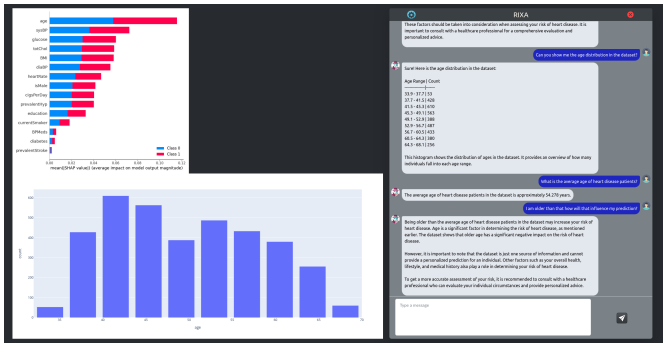

Fig. 2. Continuation of the interaction with RIXA

feature importance with SHAP. It shows the SHAP plot on the left and explains some relevant features in the chat.

The conversation continues in Figure 2. Here, the user proceeds to asks for the age distribution in the dataset which is answered with a histogram provided by the data exploration plugin. Afterwards the user asks for the average age of heart disease patients. This request is handled by the data exploration plugin as well. It automatically creates a query for patients with heart disease in the dataset and calculates the mean of their age. At the end the user asks for the influence of their age. RIXA references the feature importance from the earlier response and explains the effect of an older age but also advises the user to consult with a doctor for a comprehensive evaluation.

This example shows that RIXA can translate natural language questions into relevant function calls to answer user questions. It can also integrate the data generated by plugins into its responses and reference previous answers. Next, we show some of the technicalities.

B. Details

Here we present some details on how RIXA works and how it can be used. RIXA is implemented in Python. We do not present the full architecture here. Everyone interested in that can have a look at our GitHub repository[13].

[10]The screenshots are not altered, and come from a running version of RIXA (September 2023).

[11]https://www.kaggle.com/datasets/aasheesh200/framingham-heart-study-dataset

[12]https://github.com/shap/shap

[13]https://github.com/finnschwall/RIXA

1) Chat: The natural language chat is the key component of RIXA, serving both as the conversational user interface and the main interaction method with the dashboard (Design Goals 1, and 3). Its main functionality is achieved through integrating state of the art LLMs, such as the OpenAI models through OpenAI's API or the Llama-based models through local integration. Details on the LLM integration can be found in the following section.

If users want to go back to an earlier response and steer the chat in an alternative direction, users can manually edit the chat history, for example by deleting or altering messages. Additionally, they can check out the metadata (used embeddings, details on function calls, see Figure 7) to get detailed information on how a message was created. Since the LLMs have seen diverse formatting in the training data, the chat system is also able to support formatting languages like *Latex*, *Markdown* and *HTML*.

As already mentioned above, we design the natural language chat system as the main interaction method with the dashboard. Besides the chat history, information about the dashboard elements, including their contents and placements, are also embedded in the input of the chat system, which enables users to ask for a plot, ask questions about the dashboard elements and conduct some manipulations of the displayed elements such as to zoom in on a plot.

The chat system also has access to relevant information about the deployed XAI methods, datasets and LLMs (see Design Goal 4). Users can thus also ask for underlying information and specific context to double check or search for additional information, more details on that are provided in the next section.

2) LLM Integration: The implementation of LLMs is a central aspect of RIXA's function, although it is not exclusively reliant on LLMs for its operation (see Section IV-B3).

The LLM is a specialized plugin within the RIXA framework, enabling the addition of new forms of LLMs as they become available. This even applies to those that may not conform to the current standards. The current version of RIXA supports all OpenAI models and local LLMs with hardware acceleration. RIXA has been tested on a limited scope, currently with NVIDIA GPUs using cuBLAS[14], primarily using a local version based on llama.cpp[15]. Systems similar to llama.cpp or ones that use OpenAI's API specifications can be integrated without the use of much code.

While ensuring full control and privacy, the use of local LLMs is not without drawbacks. Performance both in terms of human perception as well as in terms of computing demands is an ongoing area of work (see Section V).

A vital objective of RIXA is to inform users, not interpret for them or provide misleading information as stated in Design Goal 4. In order to maintain the factual accuracy of the LLM's responses, it is provided with relevant text segments. The text retrieval is done by plugins. There are standard plugins for that but the details can be changed by developers. If for example a specific use case involves private data that should not be fed into an online LLM, the text retrieval has to be adapted to be aware of that.

[14]https://docs.nvidia.com/cuda/cublas/

[15]https://github.com/ggerganov/llama.cpp

```python
from rixa.remote_api import display_in_chat, display_plot
from rixa.plugin_definition import plugin_init, plugin_method


@plugin_init(name="Exemplary XAI plugin", call_architecture="functional")
def xai_plugin(ctx, config, meta_config):
    ...# do some initialization. E.g. load a large model.


@data.plugin_method(help="Generate a generic XAI plot.")
@argument('data_point')
@option('--res', default=30, help='Plot resolution', type=int)
@option('-p', "--param", default=-5, help='Some really important parameter', type=float)
def xai_plot(data_point, res=50, param=-5):
    ... #Generate stuff
    display_plot(plot_obj) #display this for the user
```

Fig. 3. Exemplary plugin code

Knowledge about the state of the dashboard is provided to the LLM by injecting it into the context, which includes information about interactive elements, their appearance and functionality. This provides the user with a more interactive way of questioning the LLM, for instance, the user can request to zoom in on a specific region.

We recognize the importance of adapting to users' needs and cognitive requirements. For instance, a medical dashboard could be accessed by both patients and doctors, who have very different needs and knowledge levels. While some of this adaptation is intrinsically done by the LLM matching the user's language style, we aim to develop a more sophisticated system. The current idea is to achieve a more reasonable result via LoRAs (see Section V).

Lastly, it is important to discuss the LLM's interaction with the server and plugin system. LLMs can utilize plugins or functions. This is accomplished by feeding all available plugin functions with information such as parameters, data types, value ranges, default values, and associated help texts into the LLM. The LLM utilizes this information to call functions. Both the format of the function call and the way the LLM attempts to call a function can differ widely. The server handles any necessary translation.

Our primary focus for future versions is to make initial LLM integration as seamless as possible. We envision a version already outfitted with a locally pre-integrated LLM.

3) Plugin System: The plugin system forms the backbone of RIXA and is our approach of fulfilling Design Goal 2. It is the mechanism that enables the inclusion of XAI components or any other additional features into the system. However, the details of its implementation are quite technical and not the primary focus of this paper. For an in-depth understanding of the plugin system, we recommend referring to our published documentation or the GitHub repository.

The fundamental idea behind the plugin system is to facilitate the seamless integration of arbitrary code, preferably with a LLM. Figure 3 shows an example for how a plugin looks like.

The server handles all other aspects, including resource management. Plugins run in a new process or thread (although usually a process). Plugin developers can specify allowed load parameters, for instance, in case of heavy AI models, where requests need to be queued. The server can manage this and more, automatically.

The server also oversees resources on the system, preventing an overload of concurrent calls. It handles per-user load to ensure that a user does not inundate the system with requests. Deadlock detection is another crucial feature managed by the server.

Plugins do not have to reside on the same computer; they are network-based. Only the plugin server code needs to be installed, which is a separate, lightweight package with minimal dependencies. The relationship among the plugin and server is many-to-many, enabling the potential for plugins as a service for private RIXA instances.

We support most Python versions, and potentially other languages too, although we have not explored this extensively. Technically, C++, C#, and Java are supported, but a local Python surrogate would be needed for the parameters and layout.

Plugins are available as locally importable packages in any other (connected) plugin, as via the normal python import statement. We have a custom import system in the background handling network imports.

The API and plugin server code is available in different forms like: *Standard*, *Functional* and *Manual*. The choice can be made per plugin, and all options can run simultaneously on the same server. The specifics boil down to a choice between a high- and low-level interface. If so desired, everything the server does can be taken care of manually by the plugin developers.

In conclusion, RIXA's plugin system is designed to be flexible and robust, catering to a variety of needs and performance requirements.

4) Configuration: The configuration of RIXA is designed with the intention to be easily setup and accessible for users, even those without a technical background. RIXA is typically installed globally, and specific instances are managed via working directories. This allows for comprehensive configuration, enabling to tailor the system to specific needs and preferences.

There are many options for customization. The server's behavior i.e. technicalities, appearance etc. can be changed via the primary configuration file. An excerpt from such a configuration can be seen in Figure 4.

```
[server]
DEBUG = True
MAINTENANCE_MODE = False
TIME_ZONE = Europe/Berlin
LOG_LOC = log/main

[security]
FULL_PASSWORD_VALIDATION = False
CSRF_COOKIE_SECURE = False
SESSION_COOKIE_SECURE = False
CSRF_COOKIE_SAMESITE = Lax
SESSION_COOKIE_SAMESITE = Lax

[plugins]
MANAGED_VENVS=False
PLUGIN_PATHS='plugins'
SHOW_ALL_PLUGIN_EXCEPTIONS = True
GENERATE_INTERMEDIARY_FILES = False
NLP_BACKEND = llama
NLP_GREETING = "RIXA here. How can I help?"
```

Fig. 4. An exemplary configuration file for RIXA

In addition to the main configuration file, RIXA's plugin system (see Section IV-B3) supports a system, where plugins can be easily added via drag-and-drop (see Figure 8. The system automatically parses and loads new plugins. Remote plugins can also be added via configuration files, but the server will not accept connections that have not been configured, as a security measure.

Plugins can be configured individually via plugin configuration files. These files always contain common settings such as virtual environment, resource management, etc. However, each plugin can also define individual settings, providing further customization options for users.

Beyond the configuration files, RIXA also offers an administrative GUI for a visual representation of the system's configuration. This GUI allows for live loading, unloading, starting, and restarting of plugins. It also supports live plugin configuration changes and provides status information. An exemplary admin interface can be seen in the appendix (Figure 8).

5) Transparency and Data Protection: Many of the components relevant to this point are mentioned in previous sections. However, the fulfillment of Design Goal 5 does not rely on one implementation alone. RIXA and all its components are open source and distributed under the Apache 2.0 license[16]. This means that anyone can have a look at them, try them out and change them. To encourage this, we aim to document all our code and provide examples for easy entry.

RIXA also aims to be truly local, platform independent[17] and use as little resources as possible[18]. Therefore, it should run on most consumer hardware without an internet connection. RIXA also supports SSL encryption, not only between the user and server but also between the plugins. This means that data processed by it is protected. Depending on the criticality of the data, different usage modes are possible from an online version using for example ChatGPT to a completely offline and fully encrypted version using a local LLM.

[16]https://www.apache.org/licenses/LICENSE-2.0

[17]Tested so far under Ubuntu, Debian, Windows 10 native and WSL2.

[18]In the current version RIXA's memory footprint is less than 200 MiB.

V. Development Road Map

RIXA is a project in development. Here we try to lay out how the project is developing and will develop in the future. However, this is just a snapshot. Further, we hope for an exchange with those developing plugins and XAI.

Currently ongoing development is listed in V-A. Planned but not yet started research and development is listed in V-B. The section ends with how we want to evaluate our work in V-C.

A. Active development

Under the umbrella of active development, we put everything that is scheduled to be mostly finished by the time the paper is published. While most of this concerns areas currently being worked on, others are scheduled for the near future.

One of the significant shortcomings of RIXA is its lack of testing and external feedback. So far RIXA has only been used by less than 20 people on the user side and just 4 on the developer side. While RIXA is very stable in our usage, this stability may not extend to all use cases that other developers may have. We also expect that our envisioned and implemented API very likely will not fulfill requirements for everybody.

To mitigate this, we plan on implementing user-centered development cycles within our institution.

At the very core of RIXA's development is the plugin system which still contains bugs and problems; however, most of them are minor. A major ongoing point is the development of many-to-many plugins and standalone plugins. The first one will enable private RIXA instances to connect to hosted plugins, e.g., to enable computationally extensive plugins as a service. The second could allow for ease of use e.g., an embeddings plugin could then provide a bash interface to generate embeddings from various sources.

While the plugin system has the main share in terms of development, the linchpin of RIXA is the LLM integration. As the development of LLMs is rapidly evolving, development in this area is never ending. New models and abilities appear regularly, and we stride to keep up with this evolution in respect to the core functionalities of RIXA. Due to the statistical nature and the opaqueness of LLMs, a lot of insights and potential improvements come from usage therefore ongoing changes are expected. Despite that, we currently have key areas in the LLM integration that get dedicated focus, the main one being spatial awareness of the models. In our experience, even the most state of the art LLMs are having great difficulties with this. The current pure text indication of where elements are is in our experience insufficient. Currently we are looking into better descriptions and/or other methods like grid systems or vision-based models. Additionally, the knowledge and state injection is of great importance. These involve relatively minor changes such as improved knowledge retrieval, better insight of the LLM into called functions, etc.

B. Future Work

Here we list ideas we will try to implement in the future. Not all of our ideas are listed. Development has already begun on some of them, but they are not integrated into RIXA yet.

A high-priority task, whose development should begin very soon, is a MoE (mixture of experts) approach to the LLMs. We aim to provide a dashboard that can meet the user's need for cognition. E.g. a doctor and a patient or a banker and a credit applicant may very well use the same dashboard with the same underlying models. Their requirement in terms of used language, usage of sources, amount of information etc. can be very different. There are also potential use cases where some form of interpretation on the LLM side is desired (e.g. banking). To achieve this, we want to use LoRAs to steer the models. We strive for a stacked LoRA design where the multiple dimensions like subject matter, cognition need or similar are covered. We also hope that this research will help to lead the LLMs to more factual answers (i.e. less imagination).

Long term we envision usage outside a user-centered dashboard. RIXA could be extended with a standardized and human-centered evaluation platform for XAI. Many approaches are evaluated using technical metrics to compare to other approaches [53]. To offer a method of mitigating these problems, we envision a test platform that gives developers the ability to easily evaluate their methods. We could offer incorporated standardized tests and questionnaires, which would simplify the process for developers, requiring them only to implement a plugin. After that, they would already have a study system that is ready to be used. The standardization of tests would make the results easily comparable and replicable. The user studies could even be conducted centrally and online, so developers submit their plugins and interested users could participate in the studies and try out different methods to evaluate them. The results should also be publicly accessible, so different methods can be compared.

Another idea is to make RIXA accessible and inclusive by incorporating text-to-speech and speech-to-text. Screen readers often rely on information given in a format without inclusivity in mind. With RIXA we have access to the origin of all information. Therefore, we would be able to facilitate a much deeper form of integration and interaction. We could not only recite what can be seen on the screen, but also use, for example, the data used to generate plots directly to convey information.

RIXA is work in progress and we do not have solutions for all our goals mentioned in Section III yet. We have to improve on end user friendliness in the usage and installation. Bias is another issue that our dashboard should address, but it is more of a psychological issue than a technical one. Lastly, we can still improve on data protection for example by incorporating ways to handle encrypted data or exploring ideas around differential privacy. We plan to provide different operating modes in the future for different criticality levels. In the next section we list our metrics for evaluating whether RIXA is a success.

C. Success Metrics

We think that a multimodal approach is superior to just one modality but we need to back this claim up. Also, we would like to give a more complete, quantifiable picture. How much better? Is that the same in all cases? Does the user prefer a multimodal approach? Does the user gain higher cognition? To answer these and other questions, we plan on conducting user studies to evaluate our work.

The first user study would preferably be conducted with participants who have little technical expertise. There we would compare RIXA to similar XAI approaches like dashboard only, chat only and combined approaches. We would compare them with two metrics: a subjective and an objective user experience. The subjective experience would measure how users perceive the explanation systems using for example the explanation satisfaction scale [54]. To measure the subjective experience, we would need a glass box model or fake classifier. We could then ask the participants questions with definitive answers about the model's behavior and see which systems enable the users to achieve better results.

We would also like to conduct a more technical experiment where we ask programmers to integrate an existing XAI method into RIXA and a selection of other common dashboard-like-tools. We will then evaluate how they perceived the process for each tool. With this we can quantify how RIXA compares to other development tools. It will also help us gain valuable feedback and suggestions for improvement on the programmer-facing side of RIXA.

VI. Conclusion

We presented our idea for a novel, user-centric XAI dashboard. It should follow our 5 design goals: 1) End-User-Centered, 2) Extendable, 3) Conversational and interactive, 4) Inform, don't interpret and 5) Data Protection and Transparency. We also showed the current state of RIXA, our implementation based on these design goals that is currently in progress. It consists of a natural language chat through which all important aspects of the dashboard can be controlled and a dashboard for visualizations. Users can ask all kinds of XAI or other questions in the chat. These can be answered directly in text or with visualizations displayed in the dashboard. XAI, at the moment, is a really technical field and directed to people with technical knowledge. We think that RIXA could make XAI more accessible to end users. With the integration of a natural language chat we want to make the dashboard usable by laypeople with no previous expertise in XAI or machine learning.

To confirm our stated goals, we will conduct user studies that evaluate how users perceive RIXA and how well they are able to understand a model with RIXA compared to similar approaches.

Acknowledgment

This work was supported by funding from the topic Engineering Secure Systems of the Helmholtz Association (HGF) and by KASTEL Security Research Labs.

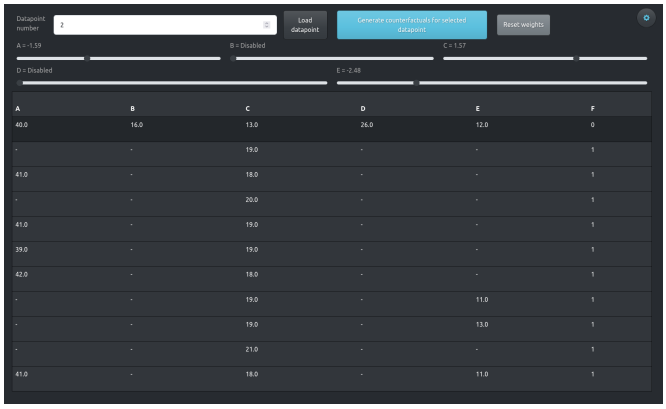

Fig. 5. A configuration of RIXA without the chat that is used in a user study

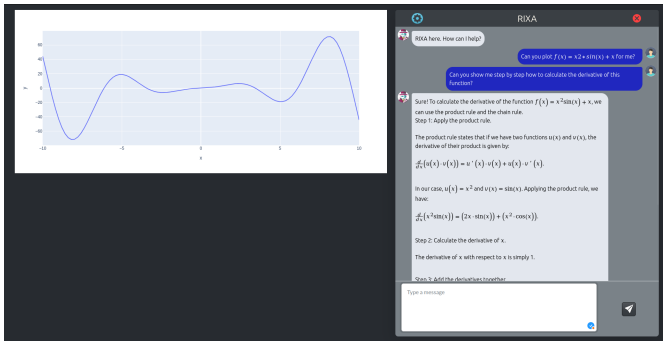

Fig. 6. Plotting a function and calculating its derivative

Appendix

A. Counterfactual User Study

RIXA is currently in use for a user study. The study does not focus on evaluating RIXA itself but a method for calculating weighted counterfactuals and their benefits for users. The natural language chat is disabled for the study. A screenshot of the study setup can be seen in Figure 5. RIXA was used because it runs stably, is lightweight and it was easy to add a plugin for the evaluated method.

B. PhysicsGPT

RIXA can not only be used for XAI applications. Another application realized with RIXA is jokingly called PhysicsGPT. As the name suggests it is a chatbot that can help with physics questions. It uses the same idea of a natural language chat which can be combined with a dashboard for plots or visualizations. The chat gets additional physics knowledge from scraped Wikipedia articles which are added as context based on embeddings.

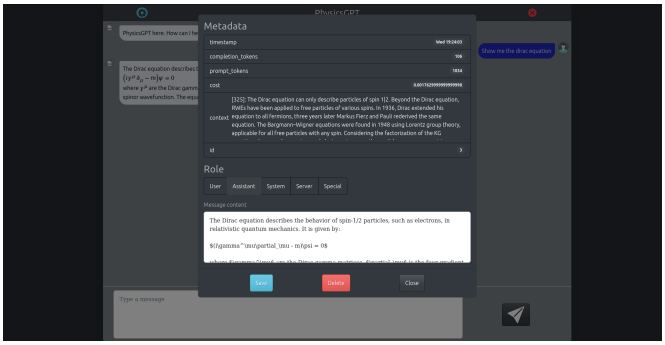

Fig. 7. Additional information about a generated chat message

C. Additional Screenshots

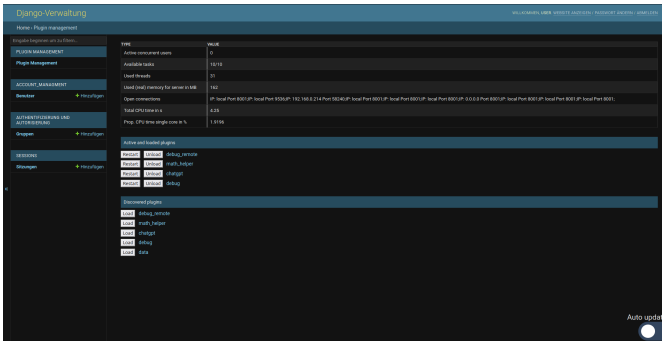

Fig. 8. RIXA's plugin manager

REFERENCES

[1] S. Tan, R. Caruana, G. Hooker, and Y. Lou, "Distill-and-compare," in *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*. ACM, dec 2018. [Online]. Available: https://doi.org/10.1145%2F3278721.3278725

[2] C. Howell, "A framework for addressing fairness in consequential machine learning," in *Proceedings of the FAT Conference Tuts, New York, NY, USA*, 2018. [Online]. Available: https://api.semanticscholar.org/CorpusID:211201589

[3] H. Wu, W. Ruan, J. Wang, D. Zheng, B. Liu, Y. Gen, X. Chai, J. Chen, K. Li, S. Li, and S. Helal, "Interpretable machine learning for covid-19: An empirical study on severity prediction task," 2021. [Online]. Available: https://europepmc.org/article/PPR/PPR318926

[4] B. Goodman and S. Flaxman, "European union regulations on algorithmic decision making and a "right to explanation"," *AI Magazine*, vol. 38, no. 3, pp. 50–57, sep 2017. [Online]. Available: https://doi.org/10.1609%2Faimag.v38i3.2741

[5] S. M. Lundberg, G. Erion, H. Chen, A. DeGrave, J. M. Prutkin, B. Nair, ..., and S.-I. Lee, "From local explanations to global understanding with explainable ai for trees," *Nature Machine Intelligence*, vol. 2, no. 1, pp. 56–67, 2020.

[6] E. Commission, "Proposal for a regulation of the european parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts." [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=CELEX:52021PC0206

[7] ——, "Proposal for a regulation of the european parliament and of the council laying down harmonised rules on artificial intelligence (artificial intelligence act) and amending certain union legislative acts." [Online]. Available: https://eur-lex.europa.eu/legal-content/EN/TXT/?uri=celex%3A52021PC0206

[8] R. Hamon, H. Junklewitz, and I. Sanchez, "Robustness and explainability of artificial intelligence," *Publications Office of the European Union*, vol. 207, 2020.

[9] N.-Y. Kim, "The more, the better? effects of multiple modalities on efl listening and reading comprehension," *STEM Journal*, vol. 22, pp. 29–45, 08 2021.

[10] K. Aslaksen, M. Haga, H. Sigmundsson, and H. Lorås, "Evidence for a common multi-modal learning style in young adults? a psychometric investigation of two modality-specific learning style inventories," *Frontiers in Education*, vol. 5, 2020. [Online]. Available: https://www.frontiersin.org/articles/10.3389/feduc.2020.00040

[11] S. Ali, T. Abuhmed, S. El-Sappagh, K. Muhammad, J. M. Alonso-Moral, R. Confalonieri, R. Guidotti, J. Del Ser, N. Díaz-Rodríguez, and F. Herrera, "Explainable artificial intelligence (xai): What we know and what is left to attain trustworthy artificial intelligence," *Information Fusion*, vol. 99, p. 101805, 2023. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1566253523001148

[12] J. Zhu, A. Liapis, S. Risi, R. Bidarra, and G. Youngblood, "Explainable ai for designers: A human-centered perspective on mixed-initiative co-creation," in *2018 IEEE conference on computational intelligence and games (CIG)*, 08 2018, pp. 1–8.

[13] S. Anjomshoae, A. Najjar, D. Calvaresi, and K. Främling, "Explainable agents and robots: Results from a systematic literature review," in *18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019), Montreal, Canada, May 13–17, 2019*. International Foundation for Autonomous Agents and Multiagent Systems, 2019, pp. 1078–1088.

[14] Y. Du, A. M. Antoniadi, C. McNestry, F. M. McAuliffe, and C. Mooney, "The role of xai in advice-taking from a clinical decision support system: A comparative user study of feature contribution-based and example-based explanations," *Applied Sciences*, vol. 12, no. 20, 2022. [Online]. Available: https://www.mdpi.com/2076-3417/12/20/10323

[15] J. Ye, X. Chen, N. Xu, C. Zu, Z. Shao, S. Liu, Y. Cui, Z. Zhou, C. Gong, Y. Shen, J. Zhou, S. Chen, T. Gui, Q. Zhang, and X. Huang, "A comprehensive capability analysis of gpt-3 and gpt-3.5 series models," 2023.

[16] K. Thakkar and N. Jagdishbhai, "Exploring the capabilities and limitations of gpt and chat gpt in natural language processing," *Journal of Management Research and Analysis*, vol. 10, pp. 18–20, 04 2023.

[17] H. Nori, N. King, S. M. McKinney, D. Carignan, and E. Horvitz, "Capabilities of gpt-4 on medical challenge problems," 2023.

[18] E. Chang, "Examining gpt-4: Capabilities, implications, and future directions," 07 2023.

[19] M. Bellucci, N. Delestre, N. Malandain, and C. Zanni-Merk, "Towards a terminology for a fully contextualized xai," *Procedia Computer Science*, vol. 192, pp. 241–250, 2021, knowledge-Based and Intelligent Information & Engineering Systems: Proceedings of the 25th International Conference KES2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S187705092101512X

[20] M. T. Ribeiro, S. Singh, and C. Guestrin, ""why should i trust you?": Explaining the predictions of any classifier," 2016.

[21] U. Ehsan, K. Saha, M. Choudhury, and M. Riedl, "Charting the sociotechnical gap in explainable ai: A framework to address the gap in xai," 02 2023.

[22] Y. Mualla, I. Tchappi, T. Kampik, A. Najjar, D. Calvaresi, A. Abbas-Turki, S. Galland, and C. Nicolle, "The quest of parsimonious xai: a human-agent architecture for explanation formulation," *Artificial Intelligence*, vol. 302, p. 103573, 08 2021.

[23] M. Ribera and À. Lapedriza García, "Can we do better explanations? a proposal of user-centered explainable ai." CEUR Workshop Proceedings, 2019.

[24] L. Yang, H. Wang, and L. A. Deleris, "What does it mean to explain? a user-centered study on ai explainability," in *International Conference on Human-Computer Interaction*. Springer, 2021, pp. 107–121.

[25] T. Miller, P. Howe, and L. Sonenberg, "Explainable ai: Beware of inmates running the asylum or: How i learnt to stop worrying and love the social and behavioural sciences," 2017.

[26] M. Eiband, H. Schneider, M. Bilandzic, J. Fazekas-Con, M. Haug, and H. Hussmann, "Bringing transparency design into practice," in *23rd international conference on intelligent user interfaces*, 03 2018, pp. 211–223.

[27] Y. Rong, T. Leemann, T.-t. Nguyen, L. Fiedler, T. Seidel, G. Kasneci, and E. Kasneci, "Towards human-centered explainable ai: User studies for model explanations," *arXiv preprint arXiv:2210.11584*, 2022.

[28] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and regression trees*. CRC press, 1984.

[29] N. Burkart and M. F. Huber, "A survey on the explainability of supervised machine learning," *Journal of Artificial Intelligence Research*, vol. 70, pp. 245–317, 2021.

[30] S. M. Lundberg and S.-I. Lee, "A unified approach to interpreting model predictions," in *Advances in neural information processing systems*, 2017, pp. 4765–4774.

[31] P. Linardatos, V. Papastefanopoulos, and S. Kotsiantis, "Explainable ai: A review of machine learning interpretability methods," *Entropy*, vol. 23, no. 1, 2021. [Online]. Available: https://www.mdpi.com/1099-4300/23/1/18

[32] A. Holzinger, A. Saranti, C. Molnar, P. Biecek, and W. Samek, *Explainable AI Methods - A Brief Overview*. Cham: Springer International Publishing, 2022, pp. 13–38. [Online]. Available: https://doi.org/10.1007/978-3-031-04083-2_2

[33] T. Clement, N. Kemmerzell, M. Abdelaal, and M. Amberg, "Xair: A systematic metareview of explainable ai (xai) aligned to the software development process," *Machine Learning and Knowledge Extraction*, vol. 5, no. 1, pp. 78–108, 2023.

[34] H. Baniecki and P. Biecek, "modelStudio: Interactive Studio with Explanations for ML Predictive Models," *Journal of Open Source Software*, vol. 4, no. 43, p. 1798, 2019. [Online]. Available: https://doi.org/10.21105/joss.01798

[35] W. Yang, H. Le, T. Laud, S. Savarese, and S. C. H. Hoi, "Omnixai: A library for explainable ai," *arXiv preprint arXiv:2206.01612*, 2022. [Online]. Available: https://arxiv.org/abs/2206.01612

[36] M. Hartmann, H. Du, N. Feldhus, I. Kruijff-Korbayová, and D. Sonntag, "XAINES: Explaining AI with narratives," *KI - Künstliche Intelligenz*, vol. 36, no. 3-4, pp. 287–296, Nov. 2022. [Online]. Available: https://doi.org/10.1007/s13218-022-00780-8

[37] N. Feldhus, A. M. Ravichandran, and S. Möller, "Mediators: Conversational agents explaining nlp model behavior," 2022.

[38] K. Sokol and P. Flach, "Conversational explanations of machine learning predictions through class-contrastive counterfactual statements," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, 7 2018, pp. 5785–5786. [Online]. Available: https://doi.org/10.24963/ijcai.2018/836

[39] ——, "Glass-box: Explaining ai decisions with counterfactual statements through conversation with a voice-enabled virtual assistant," in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, 7 2018, pp. 5868–5870. [Online]. Available: https://doi.org/10.24963/ijcai.2018/865

[40] L. Malandri, F. Mercorio, M. Mezzanzanica, and N. Nobani, "ConvXAI: a system for multimodal interaction with any black-box explainer," *Cognitive Computation*, Nov. 2022. [Online]. Available: https://doi.org/10.1007/s12559-022-10067-7

[41] D. Slack, S. Krishna, H. Lakkaraju, and S. Singh, "Talktomodel: Explaining machine learning models with interactive natural language conversations," 2023.

[42] J. M. Alonso and A. Bugarín, "Expliclas: automatic generation of explanations in natural language for weka classifiers," in *2019 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)*. IEEE, 2019, pp. 1–6.

[43] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019.

[44] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever, "Language models are unsupervised multitask learners," 2019.

[45] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever, "Improving language understanding by generative pre-training," 2018.

[46] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *Advances in neural information processing systems*, vol. 33, pp. 1877–1901, 2020.

[47] J. W. Rae, S. Borgeaud, T. Cai, K. Millican, J. Hoffmann, F. Song, J. Aslanides, S. Henderson, R. Ring, S. Young *et al.*, "Scaling language models: Methods, analysis & insights from training gopher," *arXiv preprint arXiv:2112.11446*, 2021.

[48] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann *et al.*, "Palm: Scaling language modeling with pathways," *arXiv preprint arXiv:2204.02311*, 2022.

[49] J. Hoffmann, S. Borgeaud, A. Mensch, E. Buchatskaya, T. Cai, E. Rutherford, D. de Las Casas, L. A. Hendricks, J. Welbl, A. Clark, T. Hennigan, E. Noland, K. Millican, G. van den Driessche, B. Damoc, A. Guy, S. Osindero, K. Simonyan, E. Elsen, J. W. Rae, O. Vinyals, and L. Sifre, "Training compute-optimal large language models," 2022.

[50] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample, "Llama: Open and efficient foundation language models," 2023.

[51] E. Frantar, S. Ashkboos, T. Hoefler, and D. Alistarh, "Gptq: Accurate post-training quantization for generative pre-trained transformers," 2023.

[52] E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, and W. Chen, "Lora: Low-rank adaptation of large language models," 2021.

[53] S. McCarthy, "Development of an explainability scale to evaluate explainable artificial intelligence (xai) methods," Ph.D. dissertation, Technological University Dublin, 2022.

[54] R. R. Hoffman, S. T. Mueller, G. Klein, and J. Litman, "Measures for explainable ai: Explanation goodness, user satisfaction, mental models, curiosity, trust, and human-ai performance," *Frontiers in Computer Science*, vol. 5, p. 1096257, 2023.